2
Conceitos Básicos

A *World Wide Web* surgiu nos anos 90 baseada em três componentes básicos: HTTP – *HyperText Transfer Protocol*, URLs – *Universal Resource Locators* – e HTML – *HyperText Markup Language*. Esses elementos se tornaram essenciais ao compartilhamento e acesso à informação, levando a WWW a um crescimento explosivo. Grande parte desse crescimento deve-se à simplicidade do HTML, que inicialmente servia apenas para mostrar informações. Entretanto, o HTML não era extensível, pelo contrário, continha marcações específicas que requeriam o entendimento dos desenvolvedores antes que mudanças pudessem ser feitas.

O advento do *eXtensible Markup Language*, ou simplesmente XML, representou uma grande mudança. Proposta em 1996 pelo *World Wide Web Consortium* - W3C, o XML oferecia uma maneira de manipular e estruturar informação similar ao HTML, mas que poderia ser organizado com diferentes marcações, o que simplificava o processo de definição e uso de meta-informação, fornecendo extensibilidade, hierarquia e formatação.

Entretanto, problemas como usabilidade ainda persistiam: humanos ainda encontravam dificuldade em acessar o conteúdo Web, o que se agravava em processos automatizados. Estes problemas eram gerados porque a Web foi concebida para ser utilizada por humanos, o HTML foi concebido para o layout, tamanho, cor e outros requisitos de apresentação apenas.

Além disso, é notório o crescimento no uso de imagens na apresentação de informação. Usuários humanos podem facilmente interpretar esta informação, mas não se pode dizer o mesmo de processos automatizados, realizados por meio de agentes e *crawlers*, sem falar em usuários detentores de restrições cognitivas.

Nesta seção abordaremos alguns conceitos que fornecem um conhecimento para o entendimento do trabalho aqui proposto.

The Semantic *Web* is a collaborative movement led by the World Wide *Web* Consortium (W3C) that promotes common formats for data on the World Wide *Web*.

2.1 A Web Semântica

O final dos anos 90 determinou o início da mudança na forma de publicar a informação na Web. Os esforços começaram a se concentrar na busca da sua compreensão, o que se tornou um desafio conhecido hoje como Web Semântica.

"The goal of Semantic Web research is to transform the Web from a linked document repository into a distributed knowledge base and application platform, thus allowing the vast range of available information and services to be more effectively exploited." [27]

Os principais responsáveis pela sua popularização: Tim Beners-Lee, Hendler e Ora Lassila, descreveram sua visão da Web Semântica em um artigo científico em 2001 [28], a qual é reproduzida a seguir:

"To date, the World Wide Web has developed most rapidly as a medium of documents for people rather than of information that can be manipulated automatically. By augmenting Web pages with data targeted at computers and by adding documents solely for computers, we will transform the Web into the Semantic Web. Computers will find the meaning of semantic data by following hyperlinks to definitions of key terms and rules for reasoning about them logically. The resulting infrastructure will spur the development of automated Web services such as highly functional agents. Ordinary users will compose Semantic Web pages and add new definitions and rules using off-the-shelf software that will assist with semantic markup."

Para a concretização dessa visão, foi necessário adicionar novas camadas de linguagens de marcação adequadas à arquitetura Web. Na Figura 2, reproduzimos o diagrama proposto por Beners-Lee, onde não só é possível

visualizar as tecnologias, como também os desafios por trás da Web Semântica 2000 [29].

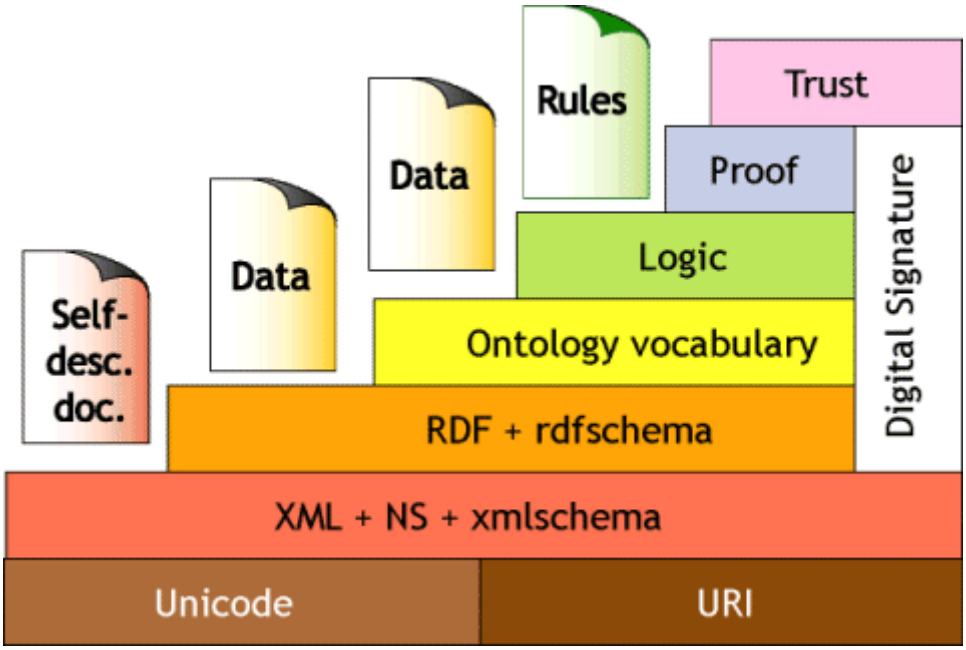

Figura 2. O "*bolo de aniversário*" da Web Semântica proposta por Berners-Lee.

A camada base é formada pelo Unicode (um padrão utilizado pela indústria na representação digital da linguagem humana, símbolos e scripts) e a URI (*Uniform Resource Identifier)* que provê um meio de localizar e identificar recursos na Web. A seguir, na próxima camada, encontra-se o XML, *Name Space* e o XML *Schema*, mecanismos criados para capacitar a validação do documento XML. O uso de XML em uma arquitetura Web é de imensa importância, uma vez que ele esta presente em muitos padrões como, por exemplo, no SOA - *Service Oriented Archictecture*.

Em seqüência, a camada composta pelo RDF e o RDF Schema, que possibilita a representação semântica da informação. O RDF - Resource Description Framework - é uma notação baseada em lógica de descrição que descreve a informação por meio de meta-informação através de triplas, onde a informação é representada através do sujeitos, predicados e o objetos. O sujeito e o objeto funcionam como adjetivos, ou "coisas" que precisam ser descritas através de URIs. O predicado tem a função de um verbo, que descreve o relacionamento entre o sujeito e o objeto, e é geralmente expresso em sintaxes como *sameAs* ou

isPartof. Em termos da Teoria dos Grafos, podemos representar um conjunto de triplas em RDF em um grafo dirigido, onde os sujeitos e objetos são nós e os predicados são arestas. O RDF *Schema*[7] estende o RDF, adicionando maior semântica à informação como domínio, subclasses e subpropriedades.

À medida que camadas vão sendo adicionadas na arquitetura ilustrada na Figura 2, mais descrição e formalismo lógico são adicionados. A camada de Ontologia, por exemplo, provê mais "meta" informação, como transitividade, unicidade, ambigüidade, cardinalidade dentre outras. Finalmente, nas últimas camadas da arquitetura, a lógica é utilizada para mediar a heterogeneidade, e o difícil desafio de determinar a confiabilidade da informação.

2.2 RDF

Como dito anteriormente, a WWW foi originalmente concebida para a utilização humana. Embora toda informação contida nela possa ser lida por máquinas, esta informação não é interpretada automaticamente. Isso ocorre por que na Web existe uma infinidade de informações originadas de diferentes fontes, modeladas de diferentes formas, e publicadas com diferentes protocolos. O RDF [30-33], em particular, permite descrever esta informação através da adição de meta-informação, o que facilita a interoperabilidade entre as aplicações que realizam troca de informações e o processamento automatizado de dados e recursos, sendo empregado de diversas formas, como por exemplo: para encontrar recursos, melhorar as máquinas de busca; na catalogação, para descrever conteúdos, disponível em Sites ou páginas; por agentes de software, para facilitar o compartilhamento e troca de conhecimento; dentre outras.

O objetivo amplo do RDF é definir mecanismos para descrever recursos de maneira que não façam menção sobre um domínio em particular, nem que definam, em um primeiro momento, a semântica de uma aplicação. Em outras palavras, a definição do mecanismo deve ser neutra de domínio, de forma que o mecanismo seja adequado para descrever a informação de qualquer domínio.

A base do RDF consiste em um modelo para representação de propriedades e seus valores. O modelo RDF foi desenhado em princípios bem estabelecidos na

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

representação da informação de várias comunidades. Uma propriedade RDF pode ser entendida como um atributo de um recurso, o que, neste contexto, corresponde ao tradicional par atributo-valor. Propriedades RDF também podem representar o relacionamento entre recursos, dessa forma, um modelo RDF pode se assemelhar a um diagrama de entidade-relacionamento.

O Framework de Descrição de Recursos – RDF – é uma forma neutra de sintaxe, utilizado para estimar a equivalência de significado entre expressões, o que determina que: duas expressões RDF só serão equivalentes se e somente se seus modelos de representação forem os mesmos. Esta definição de equivalência permite a ocorrência de variação de sintaxe sem que ocorra a alteração do significado. Dessa forma, RDF pode ser escrito de várias maneiras, a exemplo do RDF/XML, triplas e RDFa.

Intrinsecamente, uma declaração RDF representa um grafo rotulado direcionado. Assim sendo, podemos representar a sentença *"Existe uma pessoa cujo nome é Edgard"* na forma de grafo, como ilustrado na Figura 3.

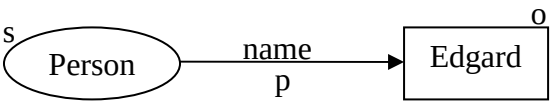

Figura 3. Representação simples de uma declaração RDF na forma de grafo.

Como podemos perceber, na Figura 3, o sujeito é representado pela classe *Person* e o predicado é um atributo da classe, e o objeto o valor (Edgard). Em RDF, o sujeito e o predicado são recursos; o objeto pode ser um literal ou outro recurso. No caso específico ilustrado pela Figura 3, o valor "Edgard" trata-se de um literal do tipo *String*.

A forma mais comum de se escrever RDF é através da *Extensible Markup Language*, ou simplesmente XML, que assim como o RDF, trata-se de uma recomendação da W3C para se escrever e estruturar a informação [31]. Além disso, XML é usado em vários protocolos de comunicação, como: SOAP [8]–*Simple Object Acess Protocol*; SPARQL [34]; ou na publicação da informação como em

[8] http://www.w3.org/TR/soap/

(X)HTML[9] [10]e RSS[11]. Como visto na seção anterior, XML consiste na camada base da Web Semântica. Existem várias razões para se utilizar XML na publicação de informação, mas talvez a mais significativa, seja porque permite tornar a informação compreensível tanto pra homens quanto para máquinas. A seguir, na Listagem 1, apresentamos um RDF correspondente à declaração contida

```
<Person namespace="http://xmlns.com/foaf/0.1/">
      <name>Edgard</name>
<Person/>
```

no grafo da Figura 3, serializado na sintaxe RDF/XML.

Listagem 1. Exemplo simples de RDF em XML utilizando o vocabulário FOAF[12].

Outra maneira de se representar RDF é através de triplas. Chamamos de triplas a forma de representar os nós na forma *{p, s, o}*, onde *p* é o predicado, *s* o sujeito e *o* o objeto. Dessa forma, a representação do grafo da Figura 3 no formato de triplas seria: *{name, Person, Edgard}*. Triplas são comumente encontradas nos chamados *triplestores*, também conhecidos como banco de dados nativos RDF, que nada mais são que banco de dados que armazenam RDF no formato de triplas: N3 visto anteriormente na forma sujeito, predicado e objeto; e no formato N4, que adiciona mais uma informação à tripla, o contexto [35].

Por fim, RDF também pode ser embutido em páginas (X)HTML através de RDFa. RDFa define um sintaxe de mapeamento RDF para um número de atributos (X)HTML, mas pode ser facilmente importada para outras linguagens baseadas em XML. A seguir, na Listagem 2, apresentamos um exemplo da declaração da Figura 3 escrita em RDFa.

[9] http://www.w3.org/TR/xhtml1/

[10] http://www.w3.org/html/

[11] http://feed2.w3.org/docs/rss2.html

[12] http://xmlns.com/foaf/spec/

```
<div xmlns:foaf="http://xmlns.com/foaf/0.1/">
   <h2 property="foaf:name">Edgard</h2>
</div>
```

Listagem 2. Exemplo simples de RDFa embutido em uma página utilizando o vocabulário FOAF.

2.3 R2RML

R2RML [36-38] é uma linguagem desenvolvida para criar mapeamentos customizados e foi proposta pelo grupo de trabalho RDB2RDF, como proposta de padronização da linguagem de mapeamento das ferramentas de conversão de banco de dados relacionais para RDF. Com R2RML, é possível materializar dados relacionais na forma RDF, estruturados e mapeados para vocabulários definidos pelo usuário. R2RML é escrito através da sintaxe Turtle [39] e, é por si só um grafo RDF, o que possibilita escrever uma sentença de diferentes formas.

Baseado no *Survey of Current Approaches for Mapping of Relational Databases to RDF*, preparado pelo grupo RDB2RDF [40], o escopo do R2RML é fundamentado nos seguintes princípios: definir o mapeamento de dados relacionais e esquemas relacionais para RDF e OWL; possuir uma sintaxe legível, bem como possuir representação em RDF e XML para possibilitar a geração e leitura por máquina; possibilitar o suporte a tipos específicos de dados SQL de diferentes fornecedores, e permitir o mapeamento de um mecanismo para criar identificadores de entidades de banco de dados.

Um documento de mapeamento R2RML [41] consiste em uma ou mais estruturas chamadas *TriplesMaps*. Cada *TriplesMap* contém uma referência a uma tabela lógica do banco de dados relacional de entrada. A tabela lógica pode ser definida por uma tabela, uma *view*, ou simplesmente uma consulta.

Além disso, um *TriplesMap* contém as regras para o mapeamento das entradas da Tabela lógica para um conjunto de triplas RDF. Essas regras são constituídas por uma estrutura *SubjectMap* e uma ou mais estruturas *PredicateObjectMap* (s).

As triplas RDF geradas a partir de uma tupla compartilham o mesmo sujeito. A estrutura *SubjectMap* em um *TriplesMap* contém as regras para gerar o sujeito de uma tupla.

Cada estrutura *PredicateObjectMap* em um *TriplesMap* contém as regras para a geração de um par – objeto, predicado – a partir dos valores na linha da Tabela. Ele consiste de uma estrutura *PredicateMap* e uma estrutura *ObjectMap*.

Um *TriplesMap* é usado para gerar triplas RDF através das entradas da Tabela lógica de um banco de dados relacional, combinando o sujeito, gerado usando o *SubjectMap*, com o(s) par(es) – objeto, predicado(s) – gerado(s) usando o(s) *PredicateObjectMap*(s). A seguir, na Listagem 3, apresentamos um exemplo de mapeamento simples.

```
<#TriplesMap1>
    a rr:TriplesMapClass;
    rr:SQLQuery """
       Select "name"
         from person
       """;

    rr:subjectMap [ rr:class foaf:person;];

    rr:predicateObjectMap
    [
      rr:predicateMap [ rr:predicate foaf:name ];
      rr:objectMap    [ rr:column "name"; rr:datatype
xsd:String];
.
```

Listagem 3. Exemplo de um mapeamento do atributo *name* de uma Tabela ralacional *person* para o vocabulário FOAF utilizando R2RML.

2.4 Templates

O termo template, em Ciências da Computação, é usado para designar os documentos que têm uma estrutura pré-definida e são utilizados como ponto de partida para a criação de novos documentos, de modo que a estrutura não necessite ser recriada quando for reutilizada. Outra definição encontrada nos dicionários Cambridge[13] e Oxford[14] define um template como algo que serve como um modelo para produção de artefatos similares.

Templates vêm sendo usados há muito tempo, nas mais diversas áreas do conhecimento: Matemática, Biologia, Química, dentre outras. O uso de templates

[13] http://dictionary.cambridge.org

é notoriamente interessante por dois aspectos: primeiro, são genéricos e reutilizáveis. Templates não são uma linguagem específica e sim um tipo de estrutura, em geral um documento, com modelos e padrões que podem ser transcritos alterando-se apenas os valores, característica essa, especialmente útil na criação de novos documentos; segundo, templates possibilitam a divisão de expertise entre dois grupos: aqueles que criam o template, e aqueles que inserem os dados, uma vez que o conhecimento utilizado na criação do template é dispensável na hora de realizar o mapeamento dos dados e vice-versa, possibilitando tanto a sintetização do conhecimento utilizado na criação do documento quanto em seu preenchimento.

Infelizmente, a importância prática e teórica de se usar templates como vantagem competitiva, bem como evidências empíricas e sistemáticas de sua utilização são escassas na literatura, talvez pela ausência de métricas aceitas para validação do uso dos mesmo.

Em um estudo publicado em 2006, Gabriel Szulanski e Robert J. Jansen [42] avaliaram o uso de templates aplicado a rotinas organizacionais. Nelson e Winter [43] usaram o termo template para se referir aos exemplos de rotinas organizacionais, que em sua concepção, contêm aspectos críticos e não críticos da rotina, fornecendo os detalhes e nuanças do trabalho, em que seqüência, e de como vários componentes e sub-rotinas são interligadas. Segundo os autores, alavancar ativos de conhecimento através da replicação de rotinas da envolve recriar conhecimento produtivo do local de origem e facilita a transferência de conhecimento. O estudo realizado por Gabriel Szulanski e Robert J. Jansen foi aplicado em 15 países europeus, durante oito anos na Xerox Europa, e revelou que a adoção de templates leva a uma transferência eficaz de conhecimento.

2.5 XML

XML [44] foi desenvolvido por um Grupo de Trabalho XML (originalmente conhecido como o Conselho de Revisão Editorial SGML), formado sob os auspícios da *World Wide Web Consortium* (W3C) em 1996.

[14] http://oxforddictionaries.com

A eXtensible Markup Language (XML) está se tornando rapidamente o padrão de fato para troca de informações na Web, sua adoção está levando ao surgimento de um novo conjunto de requisitos à manipulação da informação, tais como a necessidade de armazenar e consultar documentos XML. XML é uma forma restrita de SGML ou Standard Generalized Markup Language [ISO8879], composta de unidades de armazenamento chamadas entidades - *Entities*. As entidades contêm ambos os dados, interpretáveis ou não. Dados interpretáveis são compostos de caracteres, alguns dos quais formam caracteres de informação, e alguns dos quais formam a estrutura como a marcação do formulário. A marcação codifica uma descrição da organização e a estrutura lógica de armazenamento do documento, que pode possuir restrições através de mecanismos (DTD, XMLSCHEMA).

O XML apresenta algumas vantagens. Por exemplo, é auto-descritivo – a marcação descreve a estrutura e nomes de tipo de dados, embora não a semântica, é portátil – Unicode, e pode descrever os dados em estruturas de árvore ou Gráfico. Ele também possui desvantagens, por exemplo, é prolixa e o acesso aos dados é lento devido à conversão e análise de texto.

A definição de um conjunto de interfaces de programação independente da linguagem, que possibilita o acesso e a manipulação de documentos, tornaram a manipulação do XML mais fácil para os programadores. O XML não só explora a necessidade de uma codificação de informações e formato padrão de armazenamento, mas também permite aos programadores escolher uma forma de manipulá-la. Atualmente, exitem duas APIs – Application *Programming Interface* – principais que definem maneiras distintas de se manipular os documentos XML: SAX [45] e DOM [46].

A especificação SAX – *Simple API for XML*, por outro lado, define uma abordagem baseada em eventos em que *parsers* navegam através das informações contidas no XML, chamando funções de manipulação sempre que determinadas partes do documento - por exemplo, nós de texto ou instruções de processamento - são encontradas.

O sistema SAX – Simple API for XML – é baseado em eventos, onde o interpretador não cria uma representação interna do documento. Em vez disso, o interpretador chama funções de manipulação quando determinados eventos, definidos pela especificação SAX, ocorrererem. Estes eventos incluem a

identificação do início e do final do documento; encontrar um nó de texto; encontrar elementos filhos ou um elemento mal formado, o que possibilita ao usuário personalizar ou criar seu próprio modelo de tratamento de eventos.

A especificação DOM – *Document Object Model* – define uma abordagem de navegação do documento XML baseada em árvore. Em outras palavras, um *parser* DOM processa dados XML e cria uma representação orientada a objetos de forma hierárquica, que pode ser percorrida em tempo de execução.

Diferentemente da API SAX, a API DOM cria uma árvore interna baseada na estrutura hierárquica dos dados contidos no arquivo XML, que permanece na memória até que seja liberada. DOM usa funções que retornam os nós, pai e filho, dando-lhes pleno acesso aos dados – estrutura, hirarquia, informações – contidos no XML. A manipulação do documento XML através de DOM é simples, e sua a API é de fácil entendimento.