3
A Web Semântica

A *web semântica* é uma extensão da *web* atual na qual a informação é publicada conjuntamente com meta-informações explicitando sua semântica, o que é essencial para permitir a interoperabilidade e cooperação entre diferentes computadores (*machine-to-machine cooperation*) sem que isso demande um esforço computacional tão grande como hoje em dia e, ao mesmo tempo, mantendo a capacidade já existente de fácil utilização por seres humanos (*man-to-man cooperation*) [9].

Há quem afirme que a *web semântica* será a *web* do conhecimento [10]. Do ponto de vista da geração de novo conhecimento, esta afirmação não está totalmente correta – a *web semântica* não terá conhecimento novo em relação a *web* atual, mas sim uma nova forma de estrutura que permitirá que o conhecimento (explícito) nela presente seja mais bem aproveitado, não só por pessoas, mas principalmente, por máquinas. Podemos comparar a *web* atual como uma biblioteca cheia de livros, porém sem nenhuma forma de organização – todos os livros estão espalhados, e quem quiser procurar por algo deve ler todos até encontrar o que quer. Já a *web semântica* é uma biblioteca estruturada, onde cada livro está separado de acordo com seu assunto, seu autor, editora, etc., ou seja, cada livro tem vinculado a si um conjunto de informações extras que não dizem respeito a conteúdo novo, mas sim ao próprio conteúdo do livro (as chamadas *meta-informações*). Além disso, tais informações estão estruturadas segundo um padrão formal e bem definido. A existência desse padrão formal e bem definido torna possível atingir o que é o grande diferencial da *web semântica*: a possibilidade de que máquinas sejam capazes de processar seu conteúdo de forma muito mais eficaz e eficiente [2, 9].

Na *web* atual, as informações estão escondidas atrás de arquivos escritos em formato HTML, o que torna difícil sua utilização em alguns contextos, pois HTML não provê nenhuma forma de adicionar informações de semântica aos

documentos, apenas indicações de formatação de como o texto deve ser exibido. Por exemplo, imagine o seguinte trecho de documento em HTML:

```
Livia tem um cão da <i>raça Labrador</i>
```

Um navegador baseado em texto iria exibir o trecho "`raça Labrador`" em itálico, mas qual seria o sentido disso? Por que esse trecho está destacado? E se o navegador não fosse baseado em texto, mas sim em voz, o que ele faria? Um ser humano, ao ler esse trecho, poderia interpretá-lo facilmente – significa que a ênfase da frase, a principal informação contida, é que o cão é um Labrador, mas isso é totalmente inapropriado para máquinas, pois não há informações suficientes para elas "interpretarem" o sentido de alguma coisa.

Vamos considerar agora uma pequena mudança: imagine que existisse a *tag* de marcação *emphasized*, cujo significado é o de indicar que os trechos por ela englobados são mais importantes que os demais. Assim, a nova frase seria:

```
Livia tem um cão da <emphasized>raça Labrador</emphasized>
```

Note que a nova *tag* não carrega mais consigo informação de formatação, mas sim informação semântica – seu objetivo não é indicar que "o trecho deve ser exibido de uma determinada maneira", mas sim que "o trecho possui relevância maior". Um navegador baseado em texto poderia continuar a exibir as palavras destacadas em itálico, ao passo que um navegador baseado em voz poderia dar uma maior ênfase nesse trecho da frase. Uma máquina, ao processar esse texto, teria como reconhecer quais os trechos que contêm as informações mais relevantes.

Vamos estender ainda mais esse exemplo, adicionando as seguintes *tags*:

- person: indica que o trecho envolvido é o nome de uma pessoa. Possui um atributo que indica um *hyperlink* para um local onde podem ser obtidas maiores informações sobre a pessoa.
- animal: indica que o trecho envolvido é um animal. Possui dois atributos: um atributo que indica um *hyperlink* para um local onde

podem ser obtidas maiores informações sobre o animal e um contendo a espécie do animal em questão.

- Dessa forma, a nova estrutura seria:

<person href = “http://www.bol.com.br/Livia”>Livia</person> tem um <animal href = “http://bol.com.br/Livia/cao” type=”cachorro”>cão</animal> da <emphasized>raça Labrador</emphasized>

Note que agora uma máquina saberia facilmente que Livia é uma pessoa, que cão é um animal da espécie cachorro, que maiores informações sobre Livia podem ser encontradas em “http://www.bol.com.br/Livia”, que maiores informações sobre cão podem ser encontradas em “http://www.bol.com.br/Livia/cao” e que a principal informação da frase está no trecho raça Labrador.

Essa é a idéia da web semântica – encontrar formas de associar aos documentos informações que possibilitem que máquinas possam processá-los de forma computacionalmente eficiente.

Assim sendo, há quem pregue que a web semântica nada mais é do que uma imensa solução de reengenharia em cima da web já existente de forma a torná-la ainda mais útil [11]. Porém, essa reestruturação de conteúdo abrirá um imenso leque de novas possibilidades de utilizações da web, impulsionando o desenvolvimento de novas aplicações, inicialmente modulares (verdadeiros processadores de informação) e, posteriormente, aplicações que façam uso desses processadores, baseadas em composição de serviços (web services [11, 12]).

3.1
Interoperabilidade Semântica na WWW

A interoperabilidade semântica pode ser definida como o conjunto de convenções adotadas que são utilizadas para dar significado à informação. A idéia é que, uma vez que seja criado um vocabulário comum e com semântica muito bem definida, todas as aplicações que compreendam esse vocabulário serão capazes de se comunicar. Dessa forma, fica claro que a interoperabilidade

semântica é um requisito essencial para que máquinas possam utilizar as informações na *web*.

Uma forma de definir conceitos e relacionamentos entre eles é utilizando ontologias. Uma ontologia pode ser definida como uma especificação formal e explícita de um conjunto de conceitos compartilhado entre diferentes partes, quer sejam pessoas, organizações ou mesmo aplicações [13]. Uma ontologia é criada com o objetivo de dividir conhecimento de um domínio de interesse comum, e provê um entendimento unificado de definições de termos de um domínio, além de especificar relações entre estes termos. Ontologias são consideradas um instrumento decisivo na obtenção da interoperabilidade semântica na *web* [2, 13].

A idéia de permitir que grupos definam seus padrões para troca de conhecimento é boa, porém não é suficiente para que haja total interoperabilidade semântica. Para que isso ocorra, é necessário que haja somente um grupo de conceitos, caso contrário ficaremos com um cenário parecido com o que existe hoje em dia com a comunicação oral no mundo: há diversas línguas faladas por diferentes grupos, sendo que algumas são bastante parecidas. Entre um mesmo grupo, a comunicação se dá sem problemas. Porém, entre diferentes grupos, é necessário um intérprete, que é alguém que fala ambas as línguas. Devido a isso, boa parte da pesquisa na área de ontologias é voltada para o desenvolvimento de tecnologias para o suporte a reuso em larga escala de ontologias em escala global. A fim de possibilitar o máximo de reuso possível, ontologias devem ser pequenos módulos com uma forte coesão interna e o mínimo de interação com outros módulos. Assumindo que o mundo é repleto de pequenos "módulos ontológicos" bem projetados, a construção de uma nova ontologia (mais complexa) é somente questão de como combinar os módulos já existentes [13]. Ainda que a interoperabilidade semântica não seja atingida em escala global, a arquitetura de "intérpretes" é uma boa solução como um meio-termo [2].

Uma outra forma de atingir a interoperabilidade semântica é, ao invés de tentar unificar as ontologias, tentar unificar o processo de publicação e compartilhamento de ontologias [2]. Esforços como a *Resource Description Language* (RDF) [14] e RDF *Schema* [15, 16] procuram fornecer um *framework* unificado baseado em meta-dados para semântica na *web*. RDF é uma forma de

representar uma ontologia, enquanto que RDF Schema provê algumas primitivas básicas de modelagem de ontologias, como as primitivas "subClassOf" e "subPropertyOf". Porém, segundo [17] e [18] a expressividade de RDF Schema é insuficiente para a representação de ontologias mais gerais. Por exemplo, não há suporte a conectivos lógicos como conjunção, disjunção e negação, não há como definir uma propriedade (atributo) de uma propriedade, não há como definir simetria ou transitividade. Outra grande crítica ao RDF Schema é a falta de semântica formal – não há mapeamento dessa linguagem para nenhum tipo de lógica formal, o que elimina qualquer possibilidade de suporte a deduções e inferências (que são fatores cruciais para processamentos automatizados) [2].

Os problemas encontrados com RDF e RDF Schema levaram a novos esforços para o desenvolvimento de uma linguagem de representação de conhecimento que seja mais formal e que possua um maior poder de expressão para criação e compartilhamento de ontologias na *web*.

Uma boa linguagem para representação de ontologias deve possibilitar não só a criação e representação de uma ontologia, mas também deve ser eficientemente processável por máquinas. São requisitos de uma boa linguagem [2]:

- Ser expressiva o suficiente para capturar muitas ontologias;
- Ter uma sintaxe única;
- Ser facilmente integrável com ontologias que foram criadas em outras linguagens;
- Ter uma semântica formal, para que máquinas possam entendê-la e realizar inferências sobre a informação por ela representada [19].

3.2
Linguagens para Representação de Ontologias para a Web

3.2.1
Simple HTML Ontology Extension (SHOE)

SHOE é uma linguagem para representação de conhecimento que possibilita inserir em páginas na *web* anotações com semântica baseada em algum tipo de ontologia [9, 20]. SHOE foi proposta como uma extensão de HTML e desenvolvida na Universidade de Maryland em 1996, ou seja, antes do desenvolvimento de RDF e XML [21]. A sintaxe de SHOE está definida numa DTD (inicialmente uma DTD em SGML e, posteriormente, uma em XML [22]).

SHOE separa descrições de termos, denominada de parte ontológica, das assertivas, conhecidas como a parte de instância. A parte ontológica de SHOE permite a definição de categorias, que podem ser comparadas às classes no paradigma de orientação a objetos, e aos conceitos na lógica. É possível também definir relações, que são análogas aos papéis na lógica, e aos atributos na orientação a objetos. Uma relação em SHOE é um predicado n-ário, enquanto que um papel na lógica é um predicado binário. SHOE também permite que regras de inferência sejam definidas numa especificação de ontologia na forma de cláusulas lógicas, por exemplo, $a \wedge b \wedge c \Rightarrow d$.

As principais vantagens de SHOE em relação a RDF são:

- A possibilidade de escrever regras de inferência na ontologia, o que é suporte suficiente para permitir que máquinas realizem deduções sobre o conteúdo de uma ontologia (raciocínio);
- SHOE pode ser utilizada em documentos HTML, o que facilita a sua adaptação aos documentos já existentes hoje em dia na *web* [9, 20].
- SHOE pode ser utilizada conjuntamente com documentos XML, o que possibilita a utilização das várias ferramentas já disponíveis para processamento de XML. Além disso, todo software que já está preparado para processar documentos XML, mas não preparado para

processar SHOE, ainda poderá utilizar parcialmente a informação contida num documento que tenha trechos em SHOE [9].

A principal desvantagem é que SHOE possui uma expressividade ainda menor que RDF Schema [2].

3.2.2
Ontology Inference Layer (OIL)

A *Ontology Inference Layer* (que também é conhecida como *Ontology Inference Language*) é uma linguagem de representação e inferência para ontologias na *web*. O desenvolvimento de OIL é patrocinado pela comunidade européia e faz parte do projeto On-to-knowledge [23]. Os principais integrantes do projeto OIL são a University of Manchester (Inglaterra), Vrije Universiteit Amsterdam (Holanda), Stanford University (EUA), University of Karlsruhe (Alemanha), AIdministrator Nederland (Holanda), Research Bell Labs (EUA) e o MIT (EUA).

A forma como OIL elimina o problema da falta de expressividade e semântica formal de RDF é através da extensão do padrão RDF Schema. É proposta uma camada adicional logo acima da camada de RDF Schema, camada esta que provê semântica formal e suporte a raciocínio [18, 24].

OIL foi projetada para ser um padrão extensível. Para tanto, OIL é estruturada em camadas:

- O nível mais baixo, chamado *Core OIL*, é compatível com RDF Schema, exceto pelo *reification mechanism* presente em RDF e que não tem suporte em OIL. Ontologias definidas pelo *Core OIL* podem ser interpretadas por qualquer aplicação que dê suporte a RDF Schema.
- O próximo nível, denominado *Standard OIL*, adiciona funcionalidades, tornando OIL apenas parcialmente compatível com RDF Schema. Essa camada é desenvolvida para prover expressividade e formalismo suficiente para permitir raciocínio e dedução.

- O nível final, chamado *Instance OIL*, é uma camada que provê integração de indivíduos (instâncias) na linguagem.

A Figura 1 ilustra a estrutura em camadas de OIL. No tipo da hierarquia, há uma quarta camada, denominada *Heavy OIL*, que não existe de fato, servindo apenas como ilustração de como poderia ser feita uma extensão à linguagem [25].

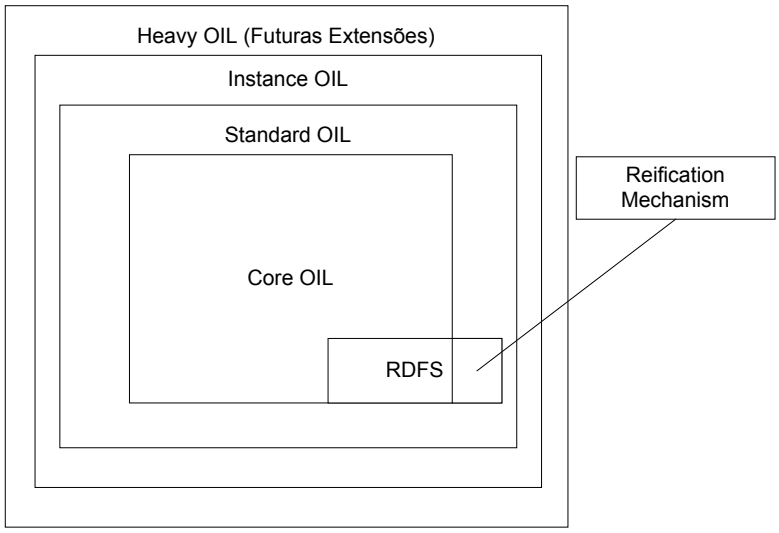

Figura 1: Estrutura das camadas de OIL

Uma ontologia escrita em OIL consiste de duas partes principais: o *ontology container* e a *ontology definition*. O container provê meta-dados sobre a ontologia, como seu título, autor, etc. OIL faz uso do *Dublin Core Metadata Element Set, Version 1.1 standard* [26] para definição dos metadados. A definição de ontologia consiste de um conjunto de expressões que descrevem classes e *slots* (que são as propriedades, equivalentes as *properties* de RDF Schema), e provê três tipos expressões: *class definition, slot constraints* e *slot definition*.

- *Class Definition*: Associa o nome de uma classe à sua descrição. Podem ser primitivas (*primitive*) ou definidas (*defined*). Uma classe primitiva pode ser definida em termos de outras classes, mas não o contrário.
- *Slot Constraint*: Podem ser definidas por um dos seguintes tipos:
- *has-value*: equivalente ao quantificador existencial na lógica (∃);
- *value-type*: equivalente ao quantificador universal na lógica (∀);

- *max-cardinality*: número máximo de instâncias que um *slot* pode ter;
- *min-cardinality*: número mínimo de instâncias que um *slot* pode ter.
- Quando *min-cardinality* = *max-cardinality*, pode-se usar apenas *cardinality*. As expressões de cardinalidade de OIL são similares às restrições numéricas na lógica. Atualmente, por motivos de eficiência computacional, OIL dá suporte apenas a dois tipos de dados primitivos: inteiro (*integer*) e string.
- *Slot Definition*: Associa o nome de um *slot* à sua descrição. Pode incluir os seguintes componentes:
- *subslot-of*: indica que esse *slot* é um *subslot* de outro. Equivalente ao *subPropertyOf* de RDF Schema.
- *domain* e *subrange*: para definir domínios válidos de valores dos *slots*, equivalentes ao *domain* e *subrange* de RDF Schema.
- *inverse*: permite a definição de um *slot* como tendo uma relação inversa com outros *slots*. Não há análogo no RDF Schema.
- *properties*: permite a definição de propriedades nos *slots*, como transitividade e simetria.

OIL provê uma semântica clara e formal para a linguagem de representação de ontologias através do mapeamento de suas expressões em lógica.

OIL possui algumas vantagens sobre outras linguagens para representação de ontologias para a web. Entre elas, está o alto acoplamento a RDF e XML. Outra grande vantagem de OIL é ter sua semântica formal definida em lógica, o que garante suporte a raciocínio e dedução.

3.2.3 Darpa Agent Markup Language (DAML)

A DARPA Agent Markup Language (DAML) [15, 27] foi construída levando-se em conta os esforços empregados e a experiência ganha em XML e RDF, OIL, SHOE, entre outros [28]. Para tanto, foi criado um comitê envolvendo

Estados Unidos e Europa, que inclui as pessoas-chave de todos estes projetos, com o objetivo de definir um *framework* unificado para uma linguagem de representação de ontologias para a *web*.

A primeira especificação de DAML, conhecida como DAML-ONT, foi lançada em Outubro de 2000 [29], mas dois meses após, em Dezembro de 2000, foi lançada uma nova especificação, chamada DAML+OIL [30], com o objetivo de substituir DAML-ONT. DAML+OIL tem uma semântica mais clara ao mesmo tempo em que tornou a linguagem mais consistente com o projeto OIL. Esta especificação também foi rapidamente substituída, em Março de 2001, por uma nova versão de DAML+OIL, denominada DAML+OIL (Março 2001) [31], cujo maior avanço foi possibilitar o uso dos tipos de dados de XML Schema [32] – nenhuma linguagem antes tinha conseguido prover suporte a tipos de dados arbitrários, o que é possível em XML Schema e, conseqüentemente, por DAML+OIL (Março 2001).

DAML+OIL (Março 2001) é dividida em duas partes. A primeira, denominada domínio de objetos (*object domain*), consiste de objetos que são membros de classes definidas na ontologia de DAML. A segunda parte é chamada de domínio de tipos (*datatype domain*), que consiste de valores que pertencem a tipos de dados oriundos de XML Schema. Por exemplo, em DAML+OIL (Março 2001), instâncias de classes, como uma pessoa chamada Matheus Leite, seriam interpretadas separadamente de instâncias de tipos, como o inteiro 6. Horrocks et al. [33] sugeriu que a separação entre tipos de dados e classes implica em que os primeiros acabam por ser modelados fora da ontologia, o que facilita não só a manutenção da simplicidade e controle de tamanho da linguagem de representação da ontologia, mas também facilita a implementação de seu suporte ao raciocínio.

Uma referência completa sobre todos os elementos de representação de ontologias providos por DAML+OIL pode ser encontrada em [34]. Até maio de 2002, havia 178 ontologias escritas em DAML e submetidas à biblioteca de ontologias em DAML (http://www.daml.org/ontologies). Ferramentas para DAML também estão sendo desenvolvidas (editores, navegadores, validadores, motores de inferência, etc. - uma lista completa pode ser achada em

http://www.daml.org/tools). A larga adoção de ontologias escritas em DAML e também a ampla disponibilização de ferramentas de suporte tornam essa linguagem uma forte candidata a ser a provedora da interoperabilidade semântica na *web*.