Capítulo

1

Representação de Conhecimento na Semantic Web

Giovani Rubert Librelotto, José Carlos Ramalho, Pedro Rangel Henriques

Resumo

A Semantic Web é uma área emergente que tem como objectivo tornar a World Wide Web (Web) mais útil e fácil de usar. O presente mini-curso visa descrever o problema da anotação para a Semantic Web, um dos actuais desafios desta área. A ideia de ter na Web dados definidos e ligados para serem usados por máquinas, não apenas para apresentação a humanos, com vista à sua integração e partilha entre aplicações, ainda é um anseio. Para isto, os dados devem ser descritos de forma que o homem e a máquina entendam seu significado.

Este mini-curso apresenta as linguagens, baseadas no padrão de anotação de texto XML, que se enquadram como tecnologia para promover a ideia da Semantic Web.

Basicamente, os mecanismos a serem desenvolvidos para o estabelecimento da Semantic Web compreendem duas vertentes: a disponibilização de uma colecção de dados estruturados; e regras de inferência associadas a essa colecção que permitirão a criação de ferramentas capazes de percorrer a Web realizando tarefas complexas com base nessas estruturas de conhecimento. Estas duas vertentes serão aqui abordadas, sendo assumido que recorre-se a ontologias para estruturar as fontes de informação e a linguagens de anotação para as descrever.

Este mini-curso consiste de uma apresentação dos formatos de representação de conhecimento (estruturas de facetas, dicionários, índices, taxonomias, thesauri, ontologias) e das linguagens para Semantic Web (RDF, RDF Schema, XOL, SHOE, OWL, Topic Maps e TMCL), assim como uma visão geral das ferramentas que suportam o desenvolvimento de aplicações e ontologias para a Semantic Web. Por fim, serão apresentados as actuais e futuras tendências para o desenvolvimento da Semantic Web.

1.1 Introdução

Considerando a maneira usual em que os utilizadores, que consultam um hiperdocumento, manuseando os navegadores da *World Wide Web* (Web), o processo baseia-se em saltar de uma página para outra a partir de palavras-chaves inseridas no corpo do documento ou em índices externos que estão associados a ligações Web (*links*).

O utilizador lê e interpreta o significado do documento encontrado – o qual foi estruturado de uma forma qualquer pelo seu criador – decidindo se tal documento é o que ele buscava; caso negativo, uma nova procura deverá ser realizada.

Todavia, o conhecimento encenado no documento não estará necessariamente disponível de forma explícita ao utilizador: pode ser requerido uma base sólida sobre o assunto para entender a informação que o documento apresenta. Por exemplo, um documento sobre *desnaturação das proteínas*, em biologia, contém um conhecimento que a maioria das pessoas não consegue extrair. O conhecimento não pode ser capturado e extraído porque estas pessoas não conseguem interpretar as palavras contido no documento, ou seja, elas não conseguem decifrar o significado implícito na informação que lhes é mostrada. E por quê? Porque tais pessoas não possuem um modelo conceptual suficientemente rico sobre este determinado domínio.

Por outro lado, os computadores são úteis para organização e processamento de dados, tipicamente mantendo as informações em hierarquias rígidas, enquanto a mente humana tem a habilidade especial de ligar pequenas unidades de informação de forma aleatória. Com base nesta constatação, a segunda geração da Web, cunhada como *Semantic Web* [Berners-Lee et al., 2001], envolve a disposição dos conceitos e de seus relacionamentos dentro de um universo de discurso de forma que não apenas a sintaxe determine uma busca: a semântica seria fundamental. Assim, um computador poderia representar associações entre coisas que poderiam parecer não relacionadas, mas que de fato, compartilham algum relacionamento.

Cada vez mais, novos recursos de informação estão sendo conectados à Web. Portanto, o sistema documental de hipermédia Web está crescendo muito rapidamente, tornando cada vez mais difícil a tarefa dos motores de busca.

Tim Berners-Lee vê a necessidade de evolução da Web, até que os recursos de informação possuem formato tal que os navegadores possam fazer associações entre as informações que se relacionam. Quando isso ocorrer de facto, terá sido implementada a *Semantic Web*.

Este mini-curso discute formas de solucionar esta situação apresentando inicialmente, na Secção 1.2, o que se entende no âmbito deste mini-curso por conhecimento distinguido de informação e de dado. A Secção 1.3 mostra algumas metodologias para representar abstractamente o conhecimento. Uma visão geral sobre *Semantic Web* encontra-se na Secção 1.4. Por fim, serão introduzidas algumas das linguagens para representação de conhecimento mais utilizadas para obter-se a *Semantic Web* (Secção 1.6). Porque as linguagens apresentadas são todas baseadas em XML, a Secção 1.5 dedica-se à introdução dessa linguagem de anotação, conceitos básicos e tecnologias associadas. A Secção 1.7 descreve ferramentas que auxiliam o projecto de ontologias para Semantic Web. Por fim, o mini-curso é concluído na Secção 1.8.

1.2 Dado, Informação e Conhecimento

Os termos *dado*, *informação* e *conhecimento* são usualmente utilizados de formas mal-definidas, o que é aceitável em conversas coloquiais. Contudo, tal uso pode resultar em uma destruição de distinções técnicas críticas, a ponto que esses termos possam ser vistos como sinónimos. Esta secção pretende delinear seus significados, de forma a evitar ambiguidades na sequência do mini-curso.

Dados e conhecimento são simplesmente pontos finais em uma linha contínua, como sugere a Figura 1.1. No lado esquerdo, encontra-se o *dado*; do lado direito, o *conhecimento* [Park and Hunting, 2003].

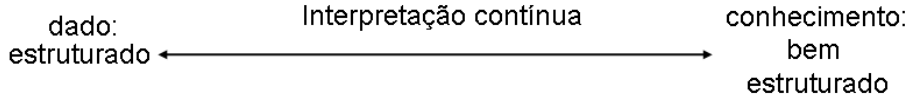

Figura 1.1: A linha contínua da interpretação. Fonte: Jack Park (2003)

O conhecimento é uma modelação simbólica complexa (uma representação) de algum aspecto de um universo de discurso (isto é, o que uma pessoa pode falar sobre seres humanos); dado é uma modelação simbólica simples. A seguinte equação expressa o relacionamento entre conhecimento e dados:

Conhecimento = Dado + Interpretação

O termo *informação* é usado nesta tese no sentido mais técnico, como Frost [Frost, 1986] derivou de Shannon [Shannon, 1974]:

> Information is a measure of the extent to which a piece of knowledge tells you something which you did not previously know.

Portanto, a informação contida em algum dado depende do que uma pessoa sabe sobre o tema e, em geral, isto pode variar de indivíduo para indivíduo. A informação representa mudança no conhecimento, pois ela altera o conhecimento sobre um determinado tema:

Novo Conhecimento = Velho Conhecimento + Informação

A partir disto, pode-se definir *interpretação* como o mapeamento entre um conjunto estruturado de dados e um modelo de algum conjunto de objectos em um universo de discurso com respeito ao significado pretendido para estes objectos e os relacionamentos entre os objectos.

Interpretação, por consequência, é o mapeamento entre notações: por exemplo, sequências de caracteres de algum alfabeto (para textos) ou algum conjunto de códigos binários definidos (para gráficos, vídeo, etc.), e o que tais notações pretendem significar em um universo de discurso definido para humanos. As notações são símbolos sem sentido a menos que elas tenham uma interpretação, isto é, mapeados para um objecto em um modelo. Interpretação é semântica: é interpretar as notações sintácticas de acordo com sua semântica pretendida.

Tipicamente, o modelo existe na mente de um humano. Quando os humanos *entendem*, eles simbolicamente representam os objectos de um domínio em algum modelo,

assim como os relacionamentos entre esses objectos. Os humanos tem a semântica do domínio em suas mentes, a qual é bem estruturada e interpretada.

Na incapacidade de abarcar e manipular a complexidade do todo, o humano necessita de arranjar modelos (representação reduzida/parcial e normativa) para entender e trabalhar a realidade.

Quando um humano lê um documento textual, ele lê as notações nas páginas e interpreta-as de acordo com o seu modelo mental; isto é, ele fornece a semântica (o significado). Se há um desejo em difundir o conhecimento contido em um documento, é necessário torná-lo disponível a outros humanos, esperando que eles forneçam um interpretador semântico (seus modelos mentais). Contudo, não há conhecimento neste documento sem interpretação; a interpretação extrai o conhecimento a partir das notações contidas em uma página.

Se o desejo é que um computador possa auxiliar na difusão do conhecimento contido em um documento, é necessário automatizar parcialmente o processo de interpretação, o que significa que é necessário construir e representar em uma forma utilizável por computadores alguma porção de nosso modelo mental.

1.3 Estruturando o Conhecimento

Actualmente, diferentes soluções têm sido apresentadas para o problema de "como estruturar o conhecimento existente em um domínio"; porém, muitas têm sido rejeitadas. Há, contudo, algumas que têm prevalecido e causado uma boa impressão em todas as áreas onde o conhecimento é abundante.

Esses *sistemas de organização de conhecimento* são mecanismos que permitem uma estruturação da informação e podem ser empregados em casos onde exista uma grande colecção de dados, como museus, bibliotecas e arquivos.

Nesta secção serão apresentados as principais formas de organização de conhecimento, os quais podem ser divididos nas seguintes classes:

Tipo universal para representação interna da informação :

- Estruturas de Facetas (*Feature Structure*);

Sistemas baseados em listas de termos: normalmente este tipo de lista tem uma estrutura simples para a representação de conhecimento. Os exemplos mais conhecidos são:

- Dicionários;
- Índices;

Sistemas baseados em grafos: determinam associações entre os termos através de um conjunto de relações semânticas. Como exemplos, tem-se:

- Taxonomias;
- Thesaurus;
- Ontologias;
  - Redes Semânticas (*Semantic Network*).

Para progredir da Web para a *Semantic Web*, tem-se que passar do nível de informação para o nível de conhecimento. Portanto, as próximas subsecções são dedicadas à apresentação das estruturas de representação de conhecimento mais utilizadas no âmbito da Semantic Web.

1.3.1 Estruturas de Facetas (*Feature Structure*)

Em sua tese de doutoramento, José João Almeida [de Almeida, 2003] disse que a utilização de um tipo universal para representação interna da informação a manipular tem sido uma constante de quase todos os ambientes.

Para determinar qual seria esse tipo universal, vários pontos devem ser levados em consideração, como por exemplo o uso de um sistema tipado ou não [Lamport and Paulson, 1999], assim como estudar as consequências a nível do poder expressivo, da complexidade de implementação e a nível da facilidade de ligação com outros ambientes.

Com base nestes pontos, as Estruturas de Facetas (EF) – ou *Feature Structures* – apresentam-se como uma boa aproximação para a representação interna da informação. As EF foram introduzidas por Martin Kay [Kay, 1979] e rapidamente se tornaram populares como modo de representar informação em áreas de processamento de linguagem natural.

Ao descrever informação usando EF, há uma necessidade de arrumar a informação em compartimentos que associam um nome de atributo a um valor. Esse valor pode ser atómico ou uma outra EF. Considere-se o seguinte exemplo, correspondente à informação ligada à palavra *comprador* [de Almeida, 2003]:

```
[ name -> comprador
  género -> masculino
  cat -> adjectivo_nomeComum
  número -> singular
  radical -> [
     name -> comprar
     Cla -> v. tr.
     Conjugação -> 1
     Fonética -> kõprár
     FrasesExemplo -> <
         O Miguel comprou um carro novo.
         A Filipa comprou mais um livro.
         O advogado comprou as testemunhas.
         ...>
     Derivadas -> [
     ...
  ...
```

A informação contida nesta FS está compartimentada em vários atributos. Certos atributos têm valores atómicos (exemplo *género*) outros têm valores complexos (exemplo *radical*).

1.3.2 Dicionários

Um dicionário[1] é uma colecção alfabetada dos vocábulos de uma língua ou de qualquer ramo do saber, com a respectiva significação, carácter fonético, mórfico e sintáctico (léxico). Pode ser visto também como um mapeamento de termos para a sua descrição ou definição.

As principais características de dicionários são:

- é um documento simples para definição de termos que requerem esclarecimentos, de modo a melhorar a comunicação e a reduzir o risco de mal-entendidos;

[1] Para uma definição do termo *dictionary*, veja *The American Heritage Dictionary of the English Language: Fourth Edition*, em http://www.bartleby.com/61/88/D0208800.html – Acedido em 15/11/2004

- é criado durante o levantamento de requisitos, e é continuamente aperfeiçoado em cada ciclo de desenvolvimento, a medida que novos termos são encontrados;
- é feito em paralelo às especificações de requisitos, aos casos de uso e ao modelo conceptual.

Um dicionário pode ser visto como um vocabulário explicativo dos termos, conceitos, palavras, expressões, frases utilizadas no domínio de aplicação em questão e que podem dar margens a interpretações erradas ou que sejam desconhecidas do público alvo e não tenham sido explicados no texto.

Um tipo de dicionário é o glossário. Glossário é um vocabulário em que se dá a explicação de certas palavras antigas ou pouco conhecidas sobre um determinado tema. Também pode ser visto como um dicionário de termos técnicos de uma arte ou ciência[2].

Em vez de um apontador para uma ocorrência de um conceito, como faz um índice, um dicionário apenas fornece a definição do conceito. Ele pode conter informações adicionais, tais como as referências *veja* (*see*) e *veja também* (*see also*), ou fornecer um guia sobre o uso ou pronúncia da linguagem em questão.

1.3.3 Índices

Um índice[3] é uma lista em ordem alfabética dos termos relevantes que surgem em um documento, ou conjunto de documentos, mapeados para a sua localização nos referidos documentos, ou seja, o índice mapeia os termos para os locais onde aparecem a informação sobre eles.

Por exemplo, em um livro indica a página (ou páginas) onde cada termo é mencionado; numa biblioteca, indica a posição (sala, estante, prateleira) de cada obra descrita pelo seu título, autor, ou assunto. Já numa base de dados, associa a chave de cada registo com a sua localização física no disco. No caso de um índice remissivo em um livro, cada conceito é associado com a lista de páginas onde tal conceito foi definido.

Ao contrário dos 3 modelos de representação de conhecimento que serão apresentados a seguir – os quais descrevem relações entre termos – o índice mapeia termos em posições (o que também facilita a busca).

1.3.4 Taxonomias

Taxonomia (do Grego: *taxis* significa organização ou divisão e *nomos* significa lei) é a ciência de classificação. Uma taxonomia[4] é um sistema de classificação, organizando entidades/conceitos. Enquanto a ontologia descreve completamente o domínio e os diferentes conceitos que compõem este domínio, uma taxonomia descreve o relacionamento hierárquico entre os conceitos, identificando os membros das classes e subclasses.

Uma boa taxonomia apresenta somente uma dimensão. Em taxonomia, as categorias devem ser mutuamente exclusivas, assim como um conceito individual deve ser

[2]Para uma definição do termo *glossary*, veja *The American Heritage Dictionary of the English Language: Fourth Edition*, em http://www.bartleby.com/61/66/G0156600.html – Acedido em 15/11/2004

[3]Para uma definição do termo *index*, veja *The American Heritage Dictionary of the English Language: Fourth Edition*, em http://www.bartleby.com/61/7/I0100700.html – Acedido em 15/11/2004.

[4]Para uma definição do termo *taxonomy*, veja *The American Heritage Dictionary of the English Language: Fourth Edition*, em http://www.bartleby.com/61/54/T0065400.html – Acedido em 15/11/2004.

encontrado em um local somente. A taxonomia deve ser exaustiva; todas as possibilidades devem ser incluídas.

Uma das mais conhecidas taxonomias é a inventada por um cientista sueco, Carolus Linnaeus (1707-1778), na qual ele classifica os organismos vivos [Prescott et al., 1996]. Considera-se que o seu trabalho está na origem da botânica moderna e da nomenclatura zoológica.

Uma classificação divide os conceitos que compõem um domínio em diferentes classes, assim como define os relacionamentos entre estes diferentes conceitos. Esses relacionamentos dizem muito sobre o significado dos conceitos individuais.

Um exemplo de uma taxonomia é apresentado na Figura 1.2. Esta taxonomia representa uma hierarquia dos conceitos encontrados no domínio do planeta Terra. As relações estão definidas através de ligações entre os conceitos: os conceitos inferiores são sub-classes dos conceitos acima deles. As sub-classes herdam as características de suas classes. Assim sendo, a partir da taxonomia da Figura 1.2 pode-se dizer que um humano é um mamífero, que por sua vez, é um animal[5].

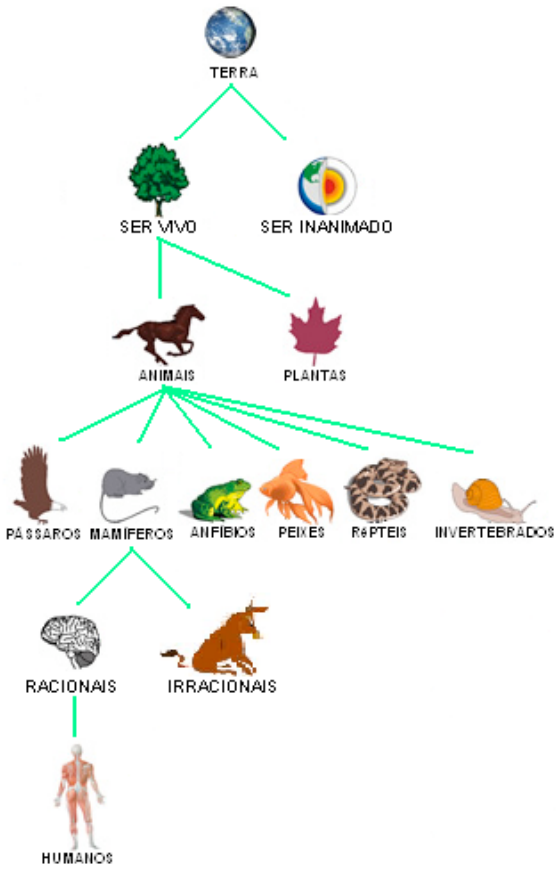

Figura 1.2: Uma taxonomia para o planeta *Terra*

Em uma taxonomia, as sub-classes herdam propriedades das classes acima delas (super-classes). Por exemplo, qualquer animal pode ser considerado um ser vivo; mas não pode, ao mesmo tempo, ser planta ou animal; isto significa que ambos partilham

[5]Figura adaptada a partir da original encontrada em *http://www.labschool.org/hondasite/taxonomy.html*

as propriedades dos organismos, mas depois cada um tem suas características distintas. Todos os filhos de um nodo pai são disjuntos. Todos os nodos filhos herdam informações dos nodos pai. Um nodo filho pode ter mais propriedades que o seu nodo pai; e, no mínimo, ele terá todas as propriedades de seu nodo pai.

1.3.5 Thesauri

Thesaurus é um instrumento que reúne termos escolhidos a partir de uma estrutura conceptual previamente estabelecida, destinados à indexação e à recuperação de documentos e informações num determinado domínio. Basicamente, ele extende as taxonomias a fim de torná-las mais completas para a descrição de um domínio, permitindo outros tipos de relações entre as classes, além de uma simples hierarquia [Garshol, 2004a].

Além da sua capacidade de organização, um thesaurus também tem um valor didático, porque utiliza conceitos específicos da área de conhecimento que contempla e permite, por meio das relações entre os termos, uma melhor compreensão da área[6].

Um thesaurus pode então ser visto como uma rede de termos interrelacionados através de um conjunto (geralmente não muito grande) de relações semânticas dentro de um domínio particular. A rede contém referências cruzadas e fornece as associações entre eles. Dado um certo termo, o thesaurus indica quais os termos que têm o mesmo significado, qual a super-classe, as sub-classes, etc.

Em suma e de acordo com a norma ISO 2788, 1986:2 standard (1986) [ISO, 1986]:

> Thesaurus is the vocabulary of a controlled indexing language, formally organized so that the a priori relationships between concepts (for example as *broader* and *narrower*) are made explicit.

Um *thesaurus* é um índice controlado utilizado tanto na indexação, quanto na recuperação de documentos. Situando-se entre a ontologia (qualquer relação entre termos) e a taxonomia (relação hierárquica), o thesaurus descreve relações de sinonimia e hierarquias, das quais se destacam:

Uso (*Use*): refere outro termo que deve ser seleccionado ao invés de um certo termo. Significa que ambos os termos são sinónimos;

Termo Genérico ou Superior: relação hierárquica que indica a super-classe (termo genérico);

Termo Específico: relação hierárquica que indica as sub-classes (termos específicos);

Termo Relacionado: refere outro termo relacionado de forma não hierárquica que está relacionado com um certo termo, sem ser um sinónimo (*Uso*);

Nota de Contexto (*Scope Note*): um texto anexado ao termo explicando seu significado dentro do thesaurus.

Um exemplo de um thesaurus está descrito na Figura 1.3. Este thesaurus é baseado na taxonomia da Figura 1.2, referindo-se também aos conceitos encontrados no domínio do planeta Terra. Além das relações definidas na taxonomia (do tipo "é-um"), este thesaurus contém relações do tipo "termo relacionado" (ligando termos com algo em comum) e "uso" (apresentando um uso de uma classe).

[6] Para uma definição do termo *thesaurus*, veja *The American Heritage Dictionary of the English Language: Fourth Edition*, em http://www.bartleby.com/61/6/T0160600.html – Acedido em 15/11/2004.

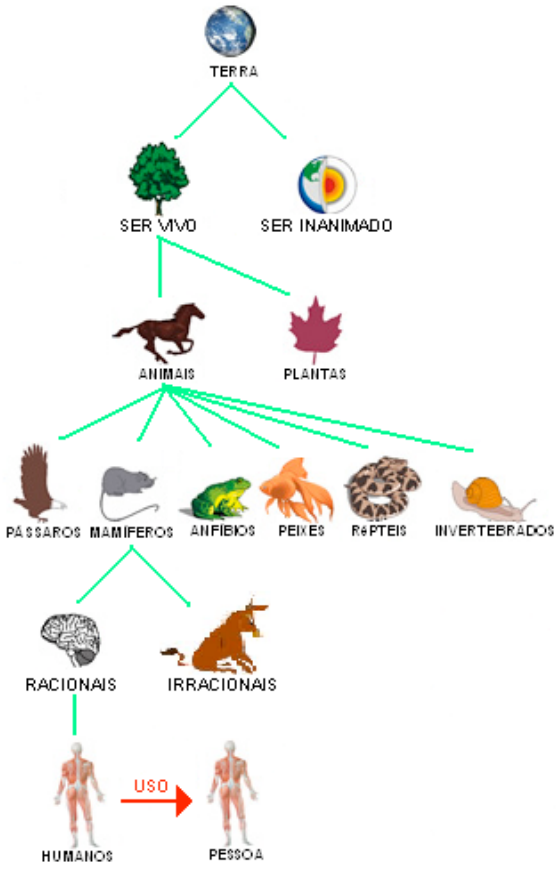

Figura 1.3: Um thesaurus para o planeta *Terra*

Um bom exemplo de thesaurus é o da UNESCO [UNESCO, 1995]. O Thesaurus UNESCO é um vocabulário controlado[7] desenvolvido pela *University of London Computer Centre* (ULCC) com permissão da *United Nations Educational, Scientific and Cultural Organization* (UNESCO), que inclui a indexação e pesquisa nas diversas áreas do conhecimento contidas nas bases de dados da UNESCO.

1.3.6 Ontologias

Hoje em dia, os sistemas convencionais de consulta utilizam técnicas de base sintáctica sobre uma forma de concordância léxica, mais do que uma aplicação da base de conhecimento do campo de interesse. Em muitos domínios, o utilizador está interessado em encontrar informação onde a relevância dos documentos não pode ser medida através do uso de sistemas de busca por palavras chaves. Neste contexto, algumas propostas envolvem a criação de metadados que envolvem a noção de ontologia.

Dentro de uma mesma área podem ser encontradas diferentes definições e caracterizações de **ontologia**[8]. Na área de Inteligência Artificial, Guarino de-

[7]Vocabulário controlado é um documento que especifica quais são os termos preferidos para serem usados domínio em questão. Padronizando os termos, é possível evitar problemas de inconsistência nas buscas realizadas. Por exemplo, quando se procura pela palavra-chave "professores de história", podem não aparecer resultados relevantes porque o documento que continha a informação desejada possui o termo "docentes" ao invés de "professores".

[8]Para uma definição do termo *ontology*, veja *The American Heritage Dictionary of the English Lan-*

fine a ontologia como uma caracterização axiomática do significado do vocabulário lógico [Guarino, 1997]; para Sowa, a ontologia define os tipos de coisas que existem no domínio de uma aplicação [Sowa, 2000].

Segundo Gruber [Gruber, 1993], uma ontologia é uma especificação explícita de uma conceptualização. Também pode ser entendida como um conjunto de termos hierarquicamente estruturados para a descrição de um domínio, o qual pode ser utilizado como um esqueleto fundamental para uma base de conhecimento.

Na área de Sistemas de Informação, na qual se encaixa este mini-curso, ontologia é definida como um conjunto de conceitos e termos ligados entre si (numa rede) que podem ser usados para descrever alguma área do conhecimento ou construir uma representação para o conhecimento [Swartout and Tate, 1999]. Segundo Chandrasekaran [Chandrasekaran, 1999], ontologias são teorias de conteúdo sobre os tipos de objectos, propriedades de objectos e relacionamentos entre objectos que são possíveis em um domínio de conhecimento específico.

Uma ontologia também pode ser vista como uma teoria lógica para descrever o significado pretendido de um vocabulário formal, isto é, seu comprometimento com uma conceptualização particular de um domínio. Estas incluem estruturas que permitem manipular termos de uma forma muito eficiente e útil para o utilizador e mecanismos de validação para comunicação inter-programas. A importância de seu uso é devida à capacidade de representar hierarquias de classes de objectos (taxonomias) e seus relacionamentos.

As ontologias colaboram no sentido de se obter uma Web onde os recursos disponíveis são acessíveis não somente por seres humanos, mas também por processos automatizados. Elas podem ser vistas como metadados que representam explicitamente a semântica dos dados, de forma processável por máquina, tornando-se factor chave para o desenvolvimento da *Semantic Web*. Isto reflecte a visão de Berners Lee [Berners-Lee et al., 2001].

Os sistemas de raciocínio baseados em ontologias podem operacionalizar esta semântica no sentido de agregar vários serviços à Web. As ontologias ajudam as pessoas e os computadores tanto a aceder a informação que eles precisam quanto a comunicarem-se entre si de forma efectiva. Isto é possível tornando explícita a ligação entre a forma e o conteúdo da informação. As ontologias têm, portanto, um papel crucial no sentido que permitem o acesso, a interoperabilidade e a comunicação baseados em conteúdo, fornecendo à Web um nível de serviço qualitativamente novo.

O desenvolvimento de ontologias irá fornecer o mecanismo de construção da parte semântica da *Semantic Web*. O modelo em camadas proposto por Berners-Lee [Berners-Lee, 2000] tem sido aceite principalmente como representação para a arquitectura da *Semantic Web*. O desenvolvimento de tais mecanismos depende, obrigatoriamente, de linguagens que expressem a informação de maneira a ser entendida por máquinas. O desafio é proporcionar uma linguagem que manipule igualmente, de maneira eficiente, dados e regras para deduções sobre esses dados e que permita que regras existentes em qualquer sistema de representação de conhecimento possam ser exportadas para

---

guage: Fourth Edition, em http://www.bartleby.com/61/32/O0083200.html – Acedido em 15/11/2004.

a Web.

O desenvolvimento de ontologias deverá representar uma parcela significativa de esforço no desenvolvimento de qualquer aplicação no futuro. Dessa forma, o desenvolvimento de ambientes para construção e manipulação de ontologias é fundamental. Tais ambientes devem ser compostos de um repositório de ontologias que possa ser manipulado por projectistas, utilizadores e programas de aplicação, permitindo a navegação, pesquisa e reutilização de termos. Quando novos termos forem acrescidos à ontologia, o ambiente deve verificar a consistência do repositório.

Uma ontologia define os termos usados para descrever e representar uma área de conhecimento. A ontologia formaliza o conhecimento através da utilização de cinco componentes [Noy and Mcguinness, 2001]:

1. Conceitos que são a representação de algo, ou de qualquer coisa, acerca do domínio em questão. As propriedades de um conceito são denominadas de atributos. Por exemplo, um conceito poderia ser uma pessoa, tendo como atributo a idade;
2. Relacionamentos que são as interligações entre os conceitos do domínio. Por exemplo, pode-se ter um relacionamento entre pessoa e universidade, através da relação "estuda em";
3. Propriedades das classes, e seus valores permitidos;
4. Axiomas que representam as condições que irão restringir a interpretação dos conceitos e relações. Eles modelam predicados que terão de ser sempre verdadeiros. Por exemplo, define-se que a idade de uma pessoa corresponde à data actual subtraindo a data de seu nascimento;
5. Instâncias que representam os elementos de uma ontologia, ou seja, são as concretizações dos conceitos e relações que foram estabelecidas pela ontologia.

Assim, pode-se dizer que a ontologia visa (em alguns aspectos) desenvolver um conjunto de regras que possibilitem a interpretação das informações disponibilizadas na Web, ou seja, a extracção do seu significado. Resumidamente, a utilização de ontologias oferece vantagens como: possibilitar o compartilhamento e a interoperabilidade do conhecimento entre diferentes domínios; estruturar o domínio de forma que se permita sua compreensão com maior clareza e objectividade; reutilizar conceitos em diferentes domínios.

Baseando-se uma vez mais no planeta Terra, uma ontologia para este domínio apresenta-se na Figura 1.4. Além das relações definidas no thesaurus descrito na Figura 1.3, a ontologia permite a inclusão de todo e qualquer relacionamento que possa ser encontrado no domínio em questão, além de permitir a inclusão de propriedades às classes. Neste exemplo, é efectuada uma relação denominada "come" entre "humano" e "peixes", definindo que "humano come peixe".

Um exemplo mais complexo é o de uma ontologia sobre um domínio relativo ao *conserto de computadores*, a qual pode incluir certos conceitos, como disco rígido (*hard disk*), unidade de disquete, placa-mãe, CPU e unidade de CD-ROM. Esta ontologia pode conter certas declarações como: "*o técnico repara o computador*" e "*o computador é reparado pelo técnico*". Então, os termos *técnico*, *repara*, *é reparado por* e *computador* necessitam estar contidos também na ontologia, desempenhando papéis quer de conceito,

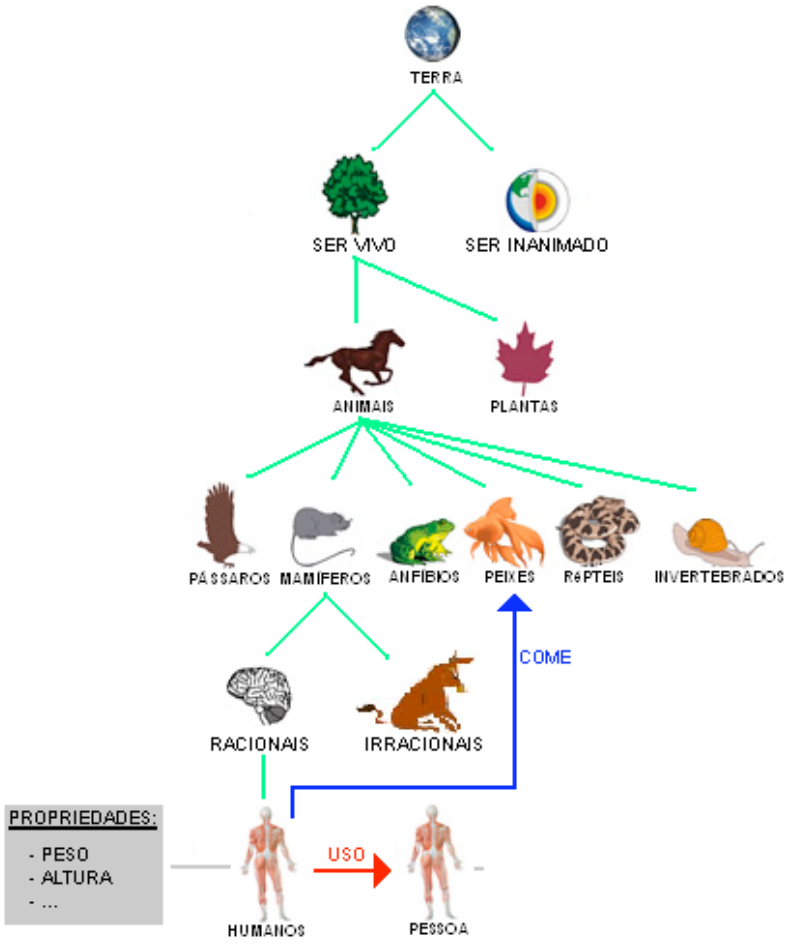

Figura 1.4: Uma ontologia para o planeta *Terra*

quer de relação. Assim, o termo *repara* introduz um relacionamento entre os conceitos *técnico* e *computador*, tal e qual como o termo *é reparado por*.

De acordo com Guarino [Guarino, 1997] e Gruber [Gruber, 1995], uma ontologia representa um vocabulário comum de um domínio. Assim, define o significado dos termos e as relações entre eles, organizados em um uma taxonomia e contém primitivas de modelação como classes, relações, funções, axiomas e instâncias.

As principais linguagens para a representação de ontologias são:

Lógica de Predicados: Cycl [Lenat and Guha, 1990];
Frame :

- Ontolingua [Farquhar et al., 1996];
- F-Logic [Kifer et al., 1995];
- CML (*Conceptual Modeling Language*) [Schreiber et al., 1994];
- OCML (*Operational Conceptual Modeling Language*) [Motta, 1999];

Lógica descritiva: Loom [McGregor, 1991];
Outras: Telos [Mylopoulos et al., 1990].

Porém, este mini-curso aborda ontologias para Web, onde as normas principais são: RDF(*Resource Description Framework*), RDF Schema, OIL (*Ontology Inference*

Layer), DAML+OIL (*DARPA Agent Markup Language*), XOL (*Ontology Exchange Language*), SHOE (*Simple HTML Ontology Extensions*), *Topic Maps* e OWL (*Ontology Web Language*). Tais linguagens serão detalhadas na Secção 1.6.

É importante ressaltar que existem diferentes conexões entre os componentes da ontologia, seus paradigmas de representação do conhecimento e suas linguagens de representação.

1.3.6.1 Rede Semânticas (*Semantic Networks*)

Em *Redes Semânticas*, tão faladas na área de Inteligência Artificial (IA), os objectos são representados como nodos de um grafo, com relações entre objectos sendo representadas por arcos. Os nodos são organizados em uma estrutura taxonómica e os arcos representam relações binárias. Tudo que pode ser expresso em lógica de primeira ordem [van Dalen, 1994, Hodges, 1997] pode também ser expresso como uma rede semântica. Um exemplo de uma rede semântica é a WordNet[9] [Fellbaum, 1999].

Com as Redes Semânticas, pretende-se em IA descrever uma estrutura para representar conhecimento através da organização de conceitos. Esta sub-secção tem a intenção de afirmar que as Redes Semânticas são um caso particular de ontologias.

1.3.7 Conexão entre os modelos

Ontologias, thesauri, taxonomias, índices e dicionários são similares nos seguintes aspectos:

- São abordagens para estruturar, classificar, modelar e representar conceitos e relacionamentos pertencentes a algum tema de interesse significante para alguma comunidade;
- Permitem que uma comunidade adopte e use o mesmo conjunto de termos de um modo uniforme;
- O significado dos termos são especificados de alguma maneira a um certo nível;

Portanto, há uma forte conexão entre os modos de expressar conhecimento descritos nas secções anteriores, conforme se pretende esquematizar na Figura 1.5. Os conceitos e seus relacionamentos são descritos e definidos em modos diferentes em ontologias, thesauri e taxonomia, como mostra a Figura 1.5. Diferentes aspectos de estrutura de conhecimento são implementados em cada um deles.

Uma ontologia descreve os conceitos existentes em um mundo particular; em uma ontologia, todos os conceitos relevantes para um certo domínio são definidos à custa de qualquer relação binária que se julga interessante citar, enquanto que nos thesauri só se usam certas relações específicas. A taxonomia clarifica os relacionamentos hierárquicos entre os conceitos, criando uma estrutura de classes/subclasses. Os índices fornecem os apontamentos para os locais onde cada conceito é referenciado, enquanto que os dicionários fornecem as definições dos conceitos.

[9] Para mais informações sobre WordNet, veja *http://www.cogsci.princeton.edu/ wn/*

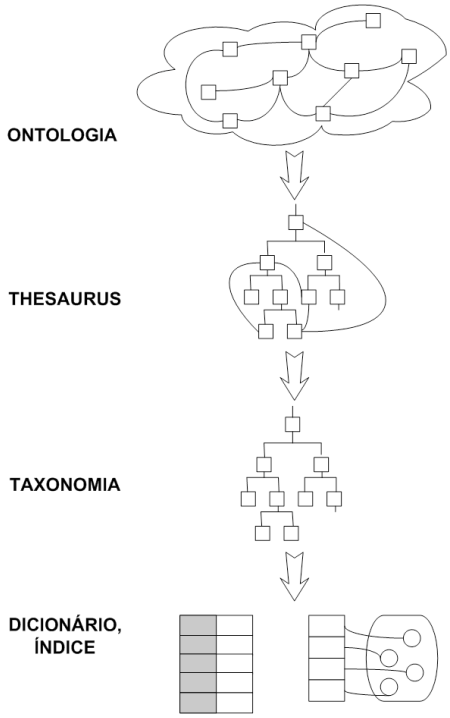

Figura 1.5: Relacionamento entre ontologia, thesaurus, taxonomia, índice e dicionário

1.4 Semantic Web

A *World Wide Web* (Web) é sem dúvida um dos maiores sucessos na história dos empreendimentos humanos, contando com utilizadores de todo o mundo, manipulando e acedendo uma enorme quantidade de informação. A Web está em amplo crescimento não apenas em seu tamanho; a sua complexidade cresce na mesma proporção. Porém, este crescimento não se reflecte no estado actual das tecnologias utilizadas para sua manipulação. A maior parte das tarefas de interpretação, acesso, extracção e manutenção da informação ainda é deixada a cargo dos utilizadores.

Os motores de busca são ineficientes quando se trata de fazer inferências complexas e relacionar assuntos aparentemente disjuntos. A simples anotação de páginas HTML por intermédio das tags <META>, ou mesmo o emprego de padrões de metadados[10], não é suficiente para incluir a semântica desejada, que possibilitaria a execução de tarefas mais sofisticadas e mais úteis do que as actualmente existentes.

Na abordagem de Tim Berners-Lee [Berners-Lee et al., 2001], esse tipo de construção leva a limitações e a um tratamento trivial do conteúdo das páginas Web por parte dos actuais browsers – limita-se a um cabeçalho; mas, em geral, as ferramentas não processam o conteúdo semântico das informações contidas em uma página.

Suponha-se que se queria pesquisar na Web a seguinte expressão: "*em quais cidades de Portugal está a chover neste exacto momento*". Mesmo havendo hoje uma

[10]Metadados são dados que descrevem dados mais complexos. Um catálogo de biblioteca é um bom exemplo de metadados porque nos permite obter dados sobre os livros contidos na biblioteca. Os metadados dão informações sobre quem escreveu, quando foi publicado, que assunto é discutido, etc.

infinidade de motores de buscas e esses fazendo uso de algoritmos cada vez mais inteligentes, provavelmente uma consulta complexa do tipo "*cidades Portugal chuva agora*" não retornaria nenhum resultado. Se, de outra forma, esta consulta for feita com as palavras "*cidades Portugal chuva*", seriam listados milhares de websites[11]; isto se dá porque os motores de busca actuais se preocupam apenas com a existência das referidas palavras-chave na página pesquisada e não com a semântica de tal busca. A menos que alguém possa aceder os sites encontrados, um a um, fazendo ligações entre os resultados, não se teria ainda resposta para essa pesquisa.

A diferença entre as duas frases acima descritas é o termo *agora*. Para encontrar o conjunto de sites que possuem a resposta para a primeira frase, o *agora* foi incluído pensando em sua semântica: o importante é saber em que cidades de Portugal está a chover *neste exacto instante*, e não onde choveu em algum momento no passado. Portanto, essa procura deveria ser baseada na semântica de cada uma das expressões utilizadas, não apenas na sintaxe, como é o caso dos motores de busca actuais.

Com base nessas premissas, surgiu a ideia da *Semantic Web*, na qual o conhecimento da Web é armazenado por meio da utilização de (meta) dados processáveis por ferramentas. Pretende-se que a *Semantic Web* não seja separada da Web, mas uma extensão desta tecnologia.

Tim Berners-Lee vê a necessidade de evolução da Web, até que ela tenha o poder de fazer com que as informações possuam formato tal que as ferramentas venham a fazer associações entre informações que se relacionam [Berners-Lee et al., 2001]. Quando isso ocorrer de facto, terá sido implementada a *Semantic Web*.

Basicamente, os mecanismos a serem desenvolvidos para o estabelecimento da *Semantic Web* compreendem duas vertentes:

1. a disponibilização de uma colecção de dados estruturados e regras de inferência associadas a essa colecção;
2. a criação de ferramentas capazes de percorrer a Web realizando tarefas complexas com base nessas estruturas de conhecimento.

Para se conseguir uma *Semantic Web*, é necessário desenvolver mecanismos para por conhecimento (semântica[12]) na Web. Se for aceite que o conhecimento humano é baseado no entendimento de significados compartilhados, então isto segue o que a *Semantic Web* necessita envolver um mecanismo pelo qual estes significados compartilhados possam ser representados e aplicados.

1.4.1 Arquitectura da Semantic Web

Considere a estrutura de *Semantic Web* apresentada na Figura 1.6 [Berners-Lee, 2000]. Percebe-se que *Unicode*[13] e *URI* (*Uniform Resource Identifier*)[14] constituem a base para

[11] No dia 29 de Março de 2005, foram encontrados 64.100 websites no Google.com a partir da busca pelas palavras-chave "cidades Portugal chuva".

[12] Para uma definição do termo *semantics*, veja *The American Heritage Dictionary of the English Language: Fourth Edition*, em *http://www.bartleby.com/61/83/S0248300.html* – Acedido em 15/11/2004.

[13] O Unicode fornece um único número para cada caracter, não importa a plataforma, não importa o programa, não importa a língua. *http://www.unicode.org/* – Acedido em 31/03/2005.

[14] Uma URI fornece um simples e extensível método para identificar um recurso. *http://www.gbiv.com/protocols/uri/rfc/rfc3986.html* – Acedido em 31/05/2005.

a legibilidade e o endereçamento na *Semantic Web*.

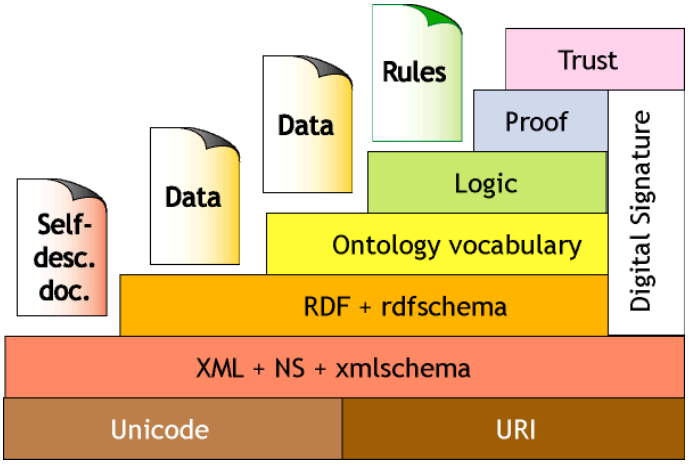

Figura 1.6: Arquitectura da *Semantic Web*. Fonte: Tim Berners-Lee (2000)

Acima disto, há o *XML* e os *namespaces*: é sabido que os browsers actuais suportam XML; para os antigos, o XML pode ser transformado em HTML através de folhas de estilo XSL. Esta camada é vista como a *camada sintáctica*.

Acima destes dois níveis, entra-se no contexto da representação dos dados e dos metadados e o seu esquema. RDF, juntamente com RDF Schema (RDFS), fornece uma estrutura bem adaptada para esta necessidade. A sugestão que tem sido feita é que a especificação Topic Maps também pode satisfazer esta necessidade.

No nível da camada ontológica, encontram-se linguagens que permitam a especificação de ontologias. *Ontology Inference Layer* (OIL) [Fensel et al., 2001] é uma linguagem de inferência e representação baseada na Web, que combina a utilização de modelação de primitivas provenientes das linguagens baseadas em frames com a semântica formal e, ainda com serviços de raciocinador provenientes de lógicas de descrição. OIL é um exemplo particular de aplicação de RDFS, iniciado como um projecto Europeu para trazer informação semântica para a Web. *DARPA Agent Markup Language* (DAML) [DARPA, 2001] é um projecto americano que foi integrado com OIL, resultando em DAML+OIL [Connolly et al., 2001], o qual define uma série de construções específicas para representação de ontologias em RDF.

O W3C está actualmente envolvido no desenvolvimento de um padrão para representação de ontologias, o *Web Ontology Language* (OWL) [Bechhofer et al., 2002] – que é amplamente baseado no DAML+OIL. OWL adiciona mais vocabulário para a descrição das classes e propriedades como, por exemplo, relações entre estas classes, cardinalidades, igualdades, tipos e características mais apurados das propriedades e enumerações das classes.

Finalizando a visão de *Semantic Web* da W3C apresentada na Figura 1.6, há ainda as camadas superiores, as quais ainda estão sob desenvolvimento: a camada lógica (*Logic*) expressa conhecimento através de regras, enquanto a camada de prova (*Proof*) utiliza essas regras para inferir conhecimento. A camada confiança (*Trust*) fornece mecanismos para determinar o grau de confiança do conhecimento obtido. As assinatura digitais

(*Digital Signature*) introduzem várias camadas para garantir segurança, através do uso de codificação dos dados (*encryption*) e assinaturas digitais.

1.5 XML e Documentos Estruturados

Extensible Markup Language (XML) [Bray et al., 2000] é uma linguagem de anotação para estruturação e marcação de documentos, desenvolvida sob a orientação do *World Wide Web Consortium* (W3C). XML permite organizar e armazenar informação estruturada e semi-estruturada.

Antes de entrarmos em mais pormenores sobre a XML e para facilitar a aprendizagem ao leitor, convém introduzir alguns conceitos necessários à sua correcta aprendizagem.

1.5.1 Conceitos

O objectivo de qualquer documento é o de guardar informação relevante para posterior consulta e/ou partilha. No entanto, a importância dos documentos depende da forma como a sua informação se encontra estruturada.

Para definir a estrutura de um documento é necessário utilizar uma série de *anotações* (*etiquetas*) de forma a dividir o documento num conjunto de elementos lógicos, tornando mais fácil a sua interpretação. Existem dois tipos de anotações: *Procedimental* e *Descritiva* [Ramalho and Henriques, 2002]. A primeira está mais virada para o aspecto físico final com o que o documento será impresso, enquanto a segunda se preocupa mais com a interpretação do seu conteúdo, fazendo a sua classificação por componentes. No âmbito deste mini-curso, o tipo de anotação a que nos referiremos será a anotação descritiva.

Vejamos o exemplo de um documento sobre o aluno *Giovani Librelotto*, anotado descritivamente:

```
1 <aluno>
2   <nome>Giovani Librelotto</nome>
3   <numero>0787</numero>
4   <curso>Doutorando em Informática</curso>
5 </aluno>
```

Como podemos verificar acima, existem quatro anotações – *aluno*, *nome*, *numero* e *curso* – que permitem identificar a maneira de interpretar o significado de cada parte do documento, i.e., o leitor facilmente se apercebe que a informação relativa a um aluno é constituída pelo seu nome, número mecanográfico e o curso que frequenta.

Este tipo de anotação apresenta grandes vantagens pois permite tratar de forma independente os diversos elementos que constituem o documento, assim como retirar alguma informação semântica dos dados.

A apresentação da informação de forma estruturada facilita os seguintes aspectos:

- **Validação** - o conhecimento prévio da estrutura de um documento permite a verificação de que a informação contida se encontra de acordo com as suas regras estruturais;

- **Reutilização** - se os documentos tiverem uma estrutura conhecida, será possível localizar alguns elementos e utilizá-los para outros fins;
- **Normalização** - o conhecimento da estrutura associada à validação permite garantir uma produção normalizada.

Isso facilita declarações mais precisas do conteúdo e resultados mais significativos de busca através de múltiplas plataformas. XML também permite o surgimento de uma nova geração de aplicações de manipulação e visualização de dados via internet.

1.5.1.1 Linguagem de anotação

Dada a possibilidade de definição livre das anotações que se julguem necessárias num determinado domínio, tem-se assistido, nos últimos anos, à definição de conjuntos de anotações para domínios específicos. É neste contexto que surge o conceito de *Linguagem de Anotação*.

Para além de definir quais as anotações a utilizar dentro de determinado contexto, uma linguagem de anotação especifica também a ordem pela qual essas anotações devem ser utilizadas.

A primeira linguagem de especificação para a definição de linguagens de anotação universalmente aceite foi a SGML (*Standardized Generalized Markup Language*). Esta surge em 1986 pela mão de um comité do *American National Standards Institute* (ANSI) designado por *Computer Languages for the Processing of Text*, cujo objectivo era normalizar a metodologia de especificação, definição e utilização de anotações em documentos. A SGML foi desenhada com o intuito de permitir a definição e a utilização de formatos de documentos, sendo suficientemente formal para permitir a validação de documentos, tendo estrutura suficiente para permitir a especificação e o manuseamento de documentos complexos e sendo extensível de modo a suportar a gestão de grandes repositórios de informação.

No final dos anos 80 Tim Berners-Lee (um investigador do CERN), ao tomar conhecimento do SGML, decide aproveitar um conjunto de anotações de uma linguagem de anotação utilizada no CERN da linguagem em que estava a trabalhar, dando origem ao HTML (*HyperText Markup Language*).

Mais tarde, em 1992, para dar um suporte formal ao HTML, este passou a ser uma linguagem formalmente especificada em SGML. Apesar da sua aceitação universal, o HTML não consegue disfarçar as suas limitações, nomeadamente no que diz respeito ao número limitado de elementos e atributos disponíveis. Perante estas limitações, estão criadas as condições para o aparecimento da XML.

Contrariamente ao que se passa com o HTML, a XML não se encontra limitada relativamente ao número de anotações de que dispõe. Na realidade o autor de um documento XML pode definir as anotações que desejar, relacioná-las com as existentes na estrutura e acrescentar-lhes os atributos que achar conveniente. Estamos assim perante uma linguagem que é:

- **Extensível** – o autor pode definir as anotações à medida das suas necessidades;

- **Estruturada** – o conjunto de anotações a utilizar num dado contexto determina uma estrutura para esse mesmo contexto;
- **Passível de ser validada** – é possível efectuar a validação do conteúdo do documento relativamente à sua estrutura.

Estas características evidenciam a XML não como uma simples linguagem mas sim como uma metalinguagem, i.e., uma linguagem que permite a definição de novas linguagens de anotação.

1.5.1.2 Documentos XML

Um documento XML é uma estrutura lógica [Ramalho and Henriques, 2002] sob a forma de uma hierarquia em árvore, onde um elemento especial é identificado como a raiz da árvore e por conseguinte do documento XML.

O próximo exemplo fornece, de uma forma bastante simplificada, a informação relativa a uma disciplina, onde: *disciplina* é a raiz de um documento que contém elementos (anotações) do tipo *designacao* e *turmas* as quais por sua vez contêm elementos do tipo *turma*. Cada *turma* é igualmente caracterizada pelas anotações *curso* e *ano*.

```
<?xml version="1.0" encoding="ISO-8859-1"?> <disciplina>
  <designacao>Processamento de Linguagens I</designacao>
  <turmas>
    <turma>
      <curso>Licenciatura em Engenharia de Sistemas e Informática</curso>
      <ano>3</ano>
    </turma>
    <turma>
      <curso>Licenciatura em Matemática e Ciências da Computação</curso>
      <ano>3</ano>
    </turma>
  </turmas>
</disciplina>
```

Como se pode verificar um documento contém dois tipos de informação: *dados* e *anotações*.

No exemplo apresentado podem-se encontrar duas das três partes que constituem um documento XML [Ramalho and Henriques, 2002]: *"Declaração"* e *"Texto anotado"ou "instância"*.

De fora ficou o DTD (*Document Type Declaration*). De uma forma genérica um DTD consiste num conjunto de declarações que especificam um tipo ou uma classe de documentos. De forma mais formal podemos dizer que um DTD [Ramalho and Henriques, 2002]:

- Define a estrutura de uma família (classe ou tipo) de documentos;
- Especifica quais as anotações/marcas disponíveis para anotar cada um dos elementos constituintes dos documentos deste tipo;
- Especifica quais os atributos que estão associados a cada elemento. Para cada atributo, indica qual o domínio e quais os valores por omissão;
- Define a estrutura do conteúdo de cada elemento: que sub-elementos tem; em que ordem; onde é que pode aparecer texto normal; onde é que podem aparecer dados que não são texto.

Vejamos o exemplo de um possível DTD para o caso da *disciplina* apresentado no exemplo anterior:

```
1 <?xml version="1.0" encoding="ISO-8859-1"?>
2 <!ELEMENT disciplina (designacao, turmas)>
3 <!ELEMENT turmas (turma+)>
4 <!ELEMENT turma (curso,ano)>
5 <!ELEMENT designacao (#PCDATA)>
6 <!ELEMENT curso (#PCDATA)>
7 <!ELEMENT ano (#PCDATA)>
```

Como podemos verificar este DTD define a estrutura de uma *disciplina*:

linha 2: uma *disciplina* é uma sequência constituída por uma designação (*designacao*) e por um elemento do tipo *turmas*;
linha 3: cada elemento do tipo *turmas* possui uma ou mais anotações do tipo *turma*;
linha 4: cada *turma* mantém informação sobre o *curso* e o *ano* em que é leccionada;
linhas 5-7: os elementos *designacao*, *curso*, *ano* são apenas texto.

Com o aparecimento dos DTD e a sua associação (opcional) aos documentos XML, surgem os conceitos de: *"documento válido"* e *"documento bem-formado"*.

Documento válido

Um documento diz-se *"válido"* se tiver um DTD associado, e se o conteúdo do documento estiver de acordo com as especificações no DTD.

Vamos então associar um DTD ao documento XML das disciplinas atrás apresentado, de forma a obtermos um documento válido.

Neste caso, optou-se por juntar o DTD ao documento. Esta é apenas uma das maneiras de associar um DTD a um documento. Normalmente, estes são externos e os documentos contêm uma referência para o DTD em questão [Ramalho and Henriques, 2002].

```
1  <?xml version="1.0" encoding="ISO-8859-1"?>
2  <!DOCTYPE disciplina [
3  <!ELEMENT disciplina (designacao, turmas)>
4  <!ELEMENT turmas (turma+)>
5  <!ELEMENT turma (curso,ano)>
6  <!ELEMENT designacao (#PCDATA)>
7  <!ELEMENT curso (#PCDATA)>
8  <!ELEMENT ano (#PCDATA)>
9  ]>
10 <disciplina>
11   <designacao>Processamento de Linguagens I</designacao>
12   <turmas>
13     <turma>
14       <curso>Licenciatura em Engenharia de Sistemas e Informática</curso>
15       <ano>3</ano>
16     </turma>
17     <turma>
18       <curso>Licenciatura em Matemática e Ciências da Computação</curso>
19       <ano>3</ano>
20     </turma>
21   </turmas>
22 </disciplina>
```

Documento bem-formado

Um documento XML diz-se *"bem-formado"* se:

- Começar pela declaração XML;

- Tiver um ou mais elementos;
- Tiver um só elemento raiz, isto é, só existir um elemento para o qual nem a anotação de início nem a de fim estão dentro de qualquer outro elemento;
- Todas as outras anotações estiverem aninhadas correctamente, i.e., uma anotação de abertura que surja dentro de um elemento tem a respectiva anotação de fecho antes desse elemento fechar;
- Os valores dos atributos estiverem entre aspas.

Perante a definição de documento bem-formado, olhando com atenção para o exemplo da disciplina anteriormente apresentado, depressa nos apercebemos de que estamos perante um documento bem-formado.

Após a exposição destes conceitos damos por terminada esta secção de conceitos iniciais sobre XML. Para mais informações sobre estes e outros conceitos sobre XML consultar [Ramalho and Henriques, 2002].

1.5.2 *Namespaces*

Já vimos que a extensibilidade da XML permite ao utilizador definir uma infinidade de anotações nos seus documentos. Apesar de todas as vantagens que daí advêm, poderão surgir alguns problemas, nomeadamente no que respeita ao uso de documentos XML dentro de outros documentos XML.

A possibilidade de num documento XML se efectuar a importação de outros documentos XML veio permitir a existência de conflitos entre os nomes dos elementos/atributos dos documentos importados, sem que estes sejam utilizados no mesmo contexto, nem tenham a mesma semântica. Surgem assim os *namespaces*, com o intuito de minorar este tipo de conflitos entre nomes de elementos/atributos.

Um *namespace* [Bray et al., 1999] é uma colecção de nomes, definindo uma única vez num documento XML, identificado por uma referência URI, que é posteriormente usadas em outros documentos XML como nomes de elementos ou atributos. Um *namespace* funciona como uma etiqueta que permite identificar univocamente os diversos nomes dos elementos e atributos de um documento XML.

Segundo recomendação do W3C, um *namespace* deve ser definido através de um *Universal Resource Identifier* (URI) a usar nos documentos XML como prefixo dos nomes dos elementos e atributos que o constituem, garantido-se assim a desejada unicidade.

Para se poder definir tal *namespace*, foi criado um atributo *global*, i.e., um atributo que pode ser instanciado em qualquer elemento do documento, denominado por: *xmlns*.

Vejamos um exemplo de um possível *namespace* para o caso da *disciplina* que temos vindo a utilizar:

```
1 <?xml version="1.0" encoding="ISO-8859-1"?>
2 <disciplina xmlns="http://www.urlficticio.pt/namespaces/disciplina">
3   <designacao>Processamento de Linguagens I</designacao>
4   <turmas>
5     <turma>
6       <curso>Licenciatura em Engenharia de Sistemas e Informática</curso>
7       <ano>3</ano>
8     </turma>
9     <turma>
```

```
10        <curso>Licenciatura em Matemática e Ciências da Computação</curso>
11        <ano>3</ano>
12      </turma>
13    </turmas>
14  </disciplina>
```

Como se pode observar na linha 2, o elemento *disciplina* passou a dispor de um atributo *xmlns*, que aponta para um URL (*http://www.urlficticio.pt/namespaces/disciplina*), tal como recomendado pelo W3C.

No exemplo apresentado, estamos perante um *namespace* por omissão, uma vez que a ele não está associado qualquer prefixo (abreviatura). Desta maneira, todos os elementos do documento, contidos no elemento onde está a definição do *namespace*, pertencem ao *namespace* declarado (neste caso, os descendentes de *disciplina*).

No entanto, é possível associar prefixos aos *namespaces*, de forma a tornar mais fácil a sua associação aos nomes dos elementos/atributos e permitir que alguns elementos não fiquem associados ao *namespace*.

Se em vez de um *namespace* por omissão usarmos um *namespace* com prefixo teríamos:

```
1  <?xml version="1.0" encoding="ISO-8859-1"?>
2  <disc:disciplina xmlns:disc="http://www.urlficticio.pt/namespaces/disciplina">
3    <disc:designacao>Processamento de Linguagens I</disc:designacao>
4    <disc:turmas>
5      <disc:turma>
6        <disc:curso>Licenciatura em Engenharia de Sistemas e Informática</disc:curso>
7        <disc:ano>3</disc:ano>
8      </disc:turma>
9      <disc:turma>
10       <disc:curso>Licenciatura em Matemática e Ciências da Computação</disc:curso>
11       <disc:ano>3</disc:ano>
12     </disc:turma>
13   </disc:turmas>
14 </disc:disciplina>
15
```

Como se pode observar todos os nomes dos elementos são agora antecedidos pelo prefixo *disc*, caso contrário, estariam fora do *namespace* declarado.

É importante referir que, sempre que se definir um *namespace*, todos os elementos a ele associados só podem ser referenciados através da concatenação do *namespace* com o seu nome. Este facto é válido tanto no processamento do documento, como em qualquer *query* que se queira fazer ao seu conteúdo.

1.5.3 XML *Schemas*

O conceito de XML *Schema* prende-se com a definição da estrutura de um documento XML, e surge como uma evolução dos DTD, tentando solucionar alguns problemas sentidos na sua utilização.

As principais vantagens dos XML *Schemas* relativamente aos DTD são:

- Contrariamente aos DTD, que possuem uma sintaxe própria, os XML *Schemas* são especificados em XML, o que permite o seu processamento por parte das ferramentas actualmente capazes de processar documentos XML;
- Suportam os tipos de dados primitivos da maioria das linguagens de programação e permitem, adicionalmente, ao utilizador definir outros tipos de dados;

- Suportam a utilização de *Namespaces*, com todas as vantagens que daí advêm (secção 1.5.2);
- Permitem especificar algumas restrições semânticas sobre o conteúdo de elementos e atributos.

Vejamos agora uma possível definição de um *Schema* para o exemplo da *disciplina* que temos vindo a utilizar.

```
1  <?xml version="1.0" encoding="iso-8859-1"?> <schema xmlns="http://www.w3.org/2001/XMLSchema">
2    <element name="ano" type="int"/>
3    <element name="curso" type="string"/>
4    <element name="designacao" type="string"/>
5    <element name="disciplina">
6      <complexType>
7        <sequence>
8          <element ref="designacao"/>
9          <element ref="turmas"/>
10       </sequence>
11     </complexType>
12   </element>
13   <element name="turma">
14     <complexType>
15       <sequence>
16         <element ref="curso"/>
17         <element ref="ano"/>
18       </sequence>
19     </complexType>
20   </element>
21   <element name="turmas">
22     <complexType>
23       <sequence>
24         <element ref="turma" maxOccurs="unbounded"/>
25       </sequence>
26     </complexType>
27   </element>
28 </schema>
```

Como se pode observar um XML *Schema* é um documento XML, sendo iniciado por uma declaração XML (linha 1), e por um elemento raiz de nome *schema* (linha 2), onde se declara o *Namespace* usual para um Schema. Nas restantes linhas deste exemplo vão sendo definidos os diversos elementos existentes.

Reparemos na definição dos elementos mais simples:

```
1  <element name="ano" type="int"/>
2  <element name="curso" type="string"/>
3  <element name="designacao" type="string"/>
```

Para definir um novo tipo de elemento é necessário utilizar o elemento *element* e indicar o nome do novo elemento no atributo *name*. Na definição destes tipos de elementos surge um atributo *type* que determina o tipo de dados que o elemento representa.

Reparemos agora no atributo *maxOccurs* da linha 25

```
1  <element ref="turma" maxOccurs="unbounded"/>
```

Este atributo indica o máximo de ocorrências do elemento a que está associado. No nosso exemplo o elemento *turma* pode ocorrer uma infinidade de vezes. Existe também a possibilidade de definir o número mínimo de ocorrências através do atributo *minOccurs*.

Na explicação deste exemplo foram deliberadamente deixadas de lado algumas referências a outros elementos da especificação de um XML *Schema*, nomeadamente *complexType* e *sequence*, uma vez que estão directamente relacionados com o tema Tipos de

Dados, o que não é abordado nesse mini-curso. Para uma consulta mais pormenorizada sobre detalhes de XML *Schemas* aconselha-se o livro [Duckett et al., 2001].

Com esta secção pretendeu-se introduzir a linguagem XML e os seus principais conceitos, necessários, na próxima secção, às linguagens de descrição de ontologias para a Semantic Web. Para tal e para além de algumas noções iniciais (importância da XML, anotação descritiva, linguagem de anotação), foram apresentados os importantes conceitos de *namespaces* e de XML *Schema*.

1.6 Linguagens para a descrição de Ontologias para a Semantic Web

Para estabelecer a *Semantic Web* é necessário a disponibilização de uma colecção de dados estruturados e regras de inferência associadas a essa colecção. Esse conjunto coerente de colecções estruturadas de informação forma uma ontologia; e a criação de ferramentas capazes de percorrer a Web realizando tarefas complexas com base nessas estruturas de conhecimento.

Após ter apresentado os modelos abstractos para descrever o conhecimento, incluindo ontologias (Secção 1.3) e uma introdução à linguagem de anotação XML (Secção 1.5), é fundamental referir linguagens concretas para se poder usar tais modelos; ou seja, propor sistemas de notação para fazer descrições que os humanos e as máquinas entendam. Para isso, apresenta-se nesta secção algumas propostas que são normas internacionais, nomeadamente RDF, RDF Schema, XOL, SHOE, OWL e Topic Maps.

1.6.1 RDF

Basicamente, RDF – *Resource Description Framework* [Lassila and Swick, 1999] – define um modelo de dados para uma descrição semântica processável por computador, o qual se parece com um diagrama de entidades e relacionamentos, em que as entidades são recursos e as propriedades RDF são representadas por relacionamentos entre recursos.

Uma declaração RDF pode ser representada através de grafo pesado (etiquetado) orientado. A Figura 1.7 ilustra esta representação diagramática.

Figura 1.7: Esquema de representação da RDF em grafo

Os arcos representam as propriedades, enquanto que os nós representam os recursos. Quando um nó é um literal, é representado por um rectângulo. A direcção da seta é relevante: o arco sempre começa no sujeito e aponta para o objecto da declaração.

1.6.1.1 O modelo básico da RDF

Uma descrição RDF assenta em três tipos de objectos, que constituem a base deste modelo:

Recursos: um recurso é qualquer dado ou fonte de informação que se quer descrever em RDF. Pode ser uma coisa (física ou lógica), uma tabela, uma base de dados,

uma página da Web, um website inteiro ou parte deste. Pode ser também um objecto não acessível de maneira electrónica, como um livro, uma revista ou um CD. Recursos são sempre especificados por URI's.

Propriedades: uma propriedade é uma característica (atributo) ou uma relação utilizada para descrever o recurso.

Declarações: uma declaração é um recurso mais as propriedades desse recurso, mais o valor de cada uma dessas propriedades; o valor, por sua vez, é outro recurso.

Essas três partes individuais são chamadas respectivamente de **sujeito**, **predicado** e **objecto**. Em outras palavras, o modelo básico primitivo RDF consiste em triplas formadas por: objecto, propriedade e valor.

Como se disse acima, cada tripla pode ser representada por um grafo, com o objecto e o valor sendo nodos e as propriedades etiquetas do ramo orientado do objecto para o valor. Isto pode também ser representado através de grafos rotulados como na figura 1.8.

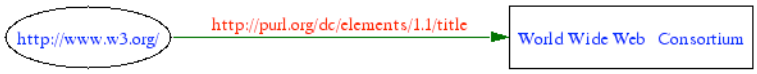

Figura 1.8: Diagrama de nós e arcos de RDF

1.6.1.2 Exemplos

Considere a seguinte frase: *Real Madrid possui o atleta Ronaldo*. Esta frase tem as seguintes partes:

Sujeito (recurso)	Real Madrid
Predicado (propriedade)	possui atleta
Objecto (literal)	Ronaldo

Este diagrama pode ser lido, de maneira mais geral, como: "<sujeito> <predicado> <objecto>".

Nesse exemplo, considere que se quer acrescentar mais características ao objecto da declaração (Ronaldo). Observe a seguinte frase: *A pessoa, cujo nome é Ronaldo, possui o e-mail r9ronaldo@realmadrid.com e é um atleta*.

A diferença entre a frase anterior e essa é que na primeira temos uma declaração com uma única propriedade ligada a um objecto (um literal). Já nessa segunda frase, há uma propriedade estruturada (atleta) que possui duas outras propriedades (nome e e-mail).

Em RDF, propriedades estruturadas são representadas como outro recurso. Assim, de acordo com as frases acima, os dois recursos – *Real Madrid* e *Ronaldo*[15] – estão relacionados pelo predicado *atleta*. Textualmente, pode-se representar este domínio da seguinte forma:

[15] A URI que serve como chave única para o Real Madrid é *http://www.realmadrid.com*, enquanto que para o atleta *Ronaldo* é *http://www.r9ronaldo.com/*.

O *Real Madrid* possui um *atleta* que tem o nome *Ronaldo*, cujo e-mail é *r9ronaldo@realmadrid.com*.

O grafo RDF que representa este domínio é apresentado na Figura 1.9.

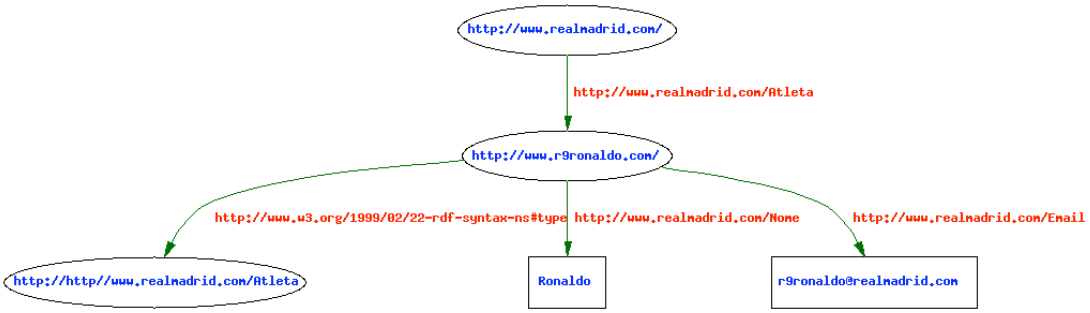

Figura 1.9: Grafo RDF exemplo

1.6.1.3 Sintaxe de RDF/XML

O World Wide Web Consortium (W3C) [W3C, 2005] propõe uma especificação XML para codificação de um modelo de dados em RDF: a sintaxe de serialização RDF/XML [Beckett and McBride, 2004].

Para exemplificar esta sintaxe, será usada a seguinte frase acima: a pessoa referenciada pelo identificador *r9ronaldo* tem o nome *Ronaldo* e o e-mail *r9ronaldo@realmadrid.com*. O Real Madrid, cujo Website é *http://www.realmadrid.com/*, tem este atleta em sua equipa.

Usando a sintaxe de RDF/XML, tem-se:

```
1  <?xml version="1.0" encoding="UTF-8"?> <rdf:RDF
2  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
3           xmlns:rm="http://www.realmadrid.com/">
4      <rdf:Description rdf:about="http://www.realmadrid.com/">
5          <rm:Atleta>
6              <rdf:Description rdf:about="http://www.r9ronaldo.com/">
7                  <rdf:type rdf:resource="http://http://www.realmadrid.com/Atleta"/>
8                  <rm:Nome>Ronaldo</rm:Nome>
9                  <rm:Email>r9ronaldo@realmadrid.com</rm:Email>
10             </rdf:Description>
11         </rm:Atleta>
12     </rdf:Description>
13 </rdf:RDF>
```

Observe que no exemplo acima, o prefixo *rm* refere-se a um *namespace* específico definido na declaração XML (linha 3).

A sintaxe RDF/XML permite que se faça relacionamento entre dados. Faz-se necessário, então, um nível de esquema onde é declarada a existência de novas propriedades. Em RDF, o RDF-Schema [Brickley and Guha, 2000a] permite verificar, por exemplo, que a carta de condução de um motorista tem o nome de uma pessoa, e não um modelo de carro.

Mais detalhes sobre RDF podem ser encontrados em [Lassila and Swick, 1999, Powers, 2003, Brickley, 2000, Ricker, 2000] e [Ahmed et al., 2001].

1.6.2 RDF Schema

RDF Schema é responsável por fornecer mecanismos para declaração das propriedades RDF, as quais representam relacionamentos entre recursos. Um esquema não define somente as propriedades dos recursos, mas também os tipos de recursos que estão sendo descritos. Pode ser entendido como uma espécie de dicionário onde são definidos os termos que serão utilizados em declarações RDF.

Conforme apresentado na subsecção anterior, RDF foi concebido para descrever metadados sobre recursos. RDF divide-se em duas partes:

1. RDF define como descrever recursos em termos de suas propriedades e valores;
2. RDF Schema define propriedades específicas que podem ser utilizadas para definir esquemas.

Essas duas definições juntas costumam ser referidas como RDF(S) [Staab et al., 2000].

RDF Schema é um sistema de classes extensível e genérico que pode ser utilizado como base para esquemas de um domínio específico. Esses esquemas podem ser compartilhados e estendidos através de refinamento de subclasses. Além disso, definições de metadados podem ser reutilizadas através do compartilhamento de esquemas.

O vocabulário de RDF está definido em dois namespaces: *rdf* e *rdfs*. Declarações RDF devem ser precedidas de um dos dois prefixos abaixo descritos:

```
1 rdfs=http://www.w3.org/2000/01/rdf-schema#
2 rdf=http://www.w3.org/1999/02/22-rdf-sysntax-ns#.
```

RDF Schema (instâncias de modelos de dados RDF) pode ser visto como diagramas Entidade-Relacionamento (DER), já que as propriedades da RDF representam relacionamentos entre recursos. RDF Schema [Brickley and Guha, 2000a] é responsável por fornecer mecanismos para declaração dessas propriedades. Um esquema não define somente as propriedades dos recursos, mas também os tipos de recursos que estão sendo descritos. Pode ser entendido como uma espécie de dicionário onde são definidos os termos que serão utilizados em declarações RDF.

A linguagem RDF Schema fornece os mecanismos necessários à definição de elementos, de classes de recursos, de possíveis restrições de classes e relacionamentos e detecção de violação de restrições.

Como tudo em RDF é considerado um recurso, RDF Schema estabelece que esses recursos podem ser organizados em classes. Um recurso pode ser instância de uma ou mais classes. A propriedade *rdf:type* é utilizada para indicar as classes das quais um recurso é instância.

As classes podem estar organizadas em uma hierarquia. Isso significa que qualquer recurso de um tipo que é subclasse de outro, é também considerado como sendo do tipo da superclasse. Tal relacionamento entre classes é denotado através da propriedade *rdfs:subClassOf*.

O sistema de tipos de RDF Schema é semelhante ao sistema de tipos do modelo de classes de UML [Fowler and Scott, 2000, OMG, 2003], com algumas pequenas diferenças. Uma delas é que, ao invés de definir classes em termos das propriedades que

suas instâncias devem ter, um RDF Schema define propriedades em termos das classes de recursos aos quais elas se aplicam. Este é o papel de *rdfs:domain* e *rdfs:range*. Por exemplo, pode-se definir que a propriedade *professor* possui um domínio *aula* e um valor *string*, enquanto em UML seria definida uma classe *aula* com um atributo chamado *professor* do tipo *string* [Brickley and Guha, 2000a]. A especificação de RDF Schema não define nenhum tipo específico de dados, mas permite que eles sejam usados como valor da propriedade *rdfs:range*.

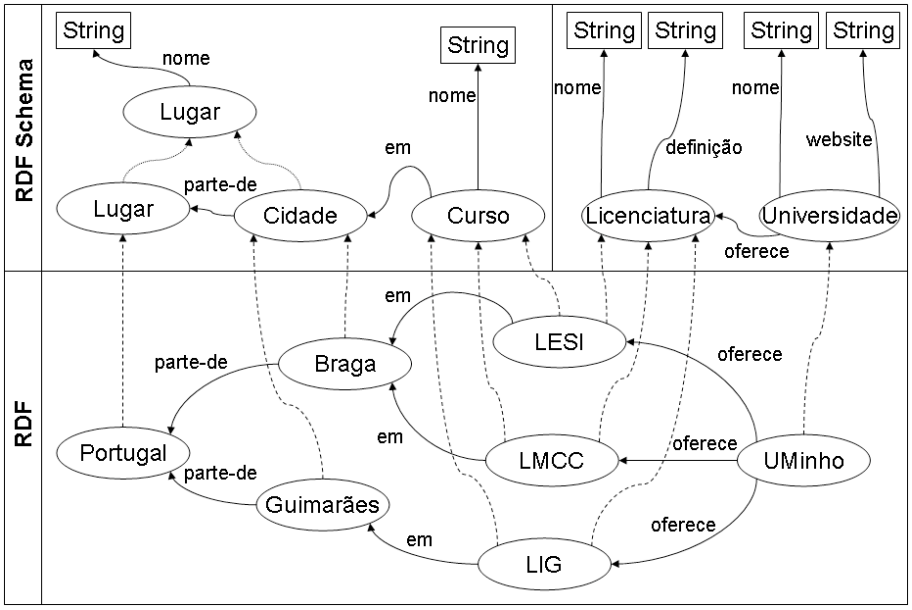

Figura 1.10: Um exemplo envolvendo RDF e RDF Schema

Para fornecer diferentes visões sobre os mesmos recursos, RDF(S) também auxilia a definir visões integradas de recursos heterogéneos. Por exemplo, a Figura 1.10 cita os cursos de licenciaturas[16] da Universidade do Minho que são relacionadas com computação. Tais licenciaturas possuem características distintas entre si (como nome e definição), estando também situadas em diferentes cidades. Além disso, cada licenciatura tem seu tempo total de duração do curso.

As camadas RDF e RDF Schema apresentadas na Figura 1.10 fornecem uma solução para a estruturação e representação destes dados.

Outro exemplo de aplicação utilizando o RDF Schema pode descrever a classe *Veículo*, a qual possui as sub-classes *Veículo de Passageiro* e *Veículo de Carga*, conforme mostra a Figura 1.11.

A codificação deste exemplo na sintaxe RDF Schema está abaixo representada:

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22/rdf-syntax-ns#"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
  <rdf:Description ID="Veiculo">
    <rdf:type resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
    <rdfs:subClassOf resource="http://www.w3.org/TR/rdf-schema#Resource"/>
  </rdf:Description>
  <rdf:Description ID="VeiculodePassageiro">
    <rdf:type resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
```

[16]Um curso de licenciatura em Portugal é o mesmo que um curso de graduação no Brasil.

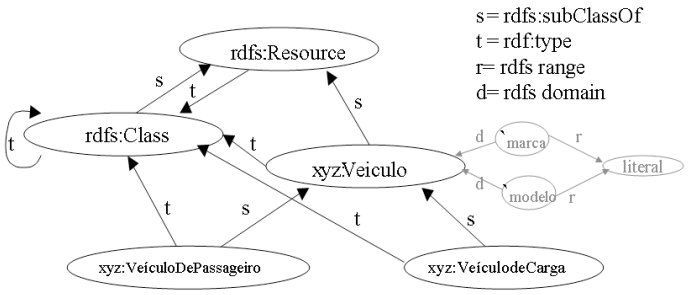

Figura 1.11: Exemplo de aplicação de RDF Schema

```
    <rdfs:subClassOf resource="#Veiculo"/>
  </rdf:Description>
  <rdf:Description ID="VeiculodeCarga">
    <rdf:type resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
    <rdfs:subClassOf resource="#Veiculo"/>
  </rdf:Description>
</rdf>
```

Detalhes mais aprofundados sobre RDF Schema podem ser encontrados em [Brickley and Guha, 2000a, Fikes and McGuinness, 2001, Brickley and Guha, 2000b, Powers, 2003] e [Brickley, 2000].

1.6.3 XOL

A linguagem *XML-based Ontology Exchange Layer* (XOL) fornece um formato que permite o intercâmbio de definições contidas em um conjunto de ontologias que se relacionam [Karp et al., 1999].

A sintaxe da XOL baseia-se em XML por ser razoavelmente simples de ser validada. A semântica de XOL baseia-se em OKBC-Lite[17](*Open Knowledge Base Connectivity*), um modelo de conhecimento simplificado.

A estrutura da sintaxe XOL possui 4 partes principais:

- cabeçalho (nome, versão);
- classes/subclasses;
- slots (propriedades);
- instâncias.

Um exemplo simples de uma ontologia representada em XOL é apresentado na Figura 1.12. A classe *Pessoa* possui duas sub-classes, nomeadamente *Homem* e *Mulher*. A classe *Pessoa* possui dois tipos de relacionamentos entre seus membros: *tem pai* e *tem irmão*.

De acordo a Figura 1.12, um exemplo de ontologia representado na sintaxe XOL seria assim descrita:

```
<module>
    <name>Familia - ontologia</name>
    <version>1.2</versao>.
    <documentation>Esse exemplo visa mostrar a sintaxe XOL.</documentation>
    <class>
        <name>Homem</name>
```

[17] *http://www.ai.sri.com/ okbc/*

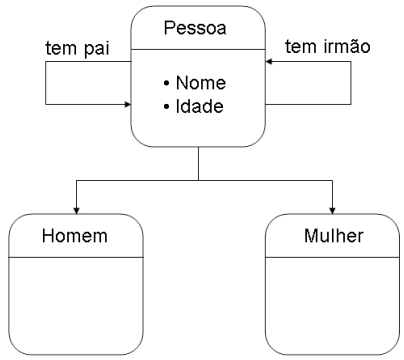

Figura 1.12: Exemplo de ontologia em XOL

```
7           <documentation>Classe de pessoas de sexo masculino.</documentation>
8           <subclass-of>Pessoa</subclass-of>
9       </class>
10      <class>
11          <name>Mulher</name>
12          <documentation>Classe de pessoas de sexo feminino.</documentation>
13          <subclass-of>Pessoa</subclass-of>
14       </class>
15       <slot>
16          <name>tem_irmaos</name>
17          <documentation>...</documentation>
18          <domain>Pessoa</domain>
19          <slot-value-type>Homem</slot-value-type>
20       </slot>
21      <slot>
22          <name>tem_pai</name>
23          <documentation>...</documentation>
24          <domain>Pessoa</domain>
25          <slot-value-type>Homem</slot-value-type>
26          <slot-inverse>pai_de<slot-inverse>
27      </slot>
28      <individual>
29          <name>Pedro</name>
30          <instance-of>Homem</instance-of>
31          <slot-values>
32               <name>tem_pai</name>
33               <value>Paulo</value>
34          </slot-values>
35      </individual>
36      <individual>
37          <name>Paulo</name>
38          <instance-of>Homem</instance-of>
39          <slot-values>
40               <name>pai-de</name>
41               <value>Pedro</value>
42          </slot-values>
43      </individual>
44  </module>
```

As características da ontologia estão descritas no princípio (linhas 2–4). Nos elementos <class> encontram-se as classes desta ontologia – homem e mulher (linhas 5–14). Nos elementos <slot> declaram-se as propriedades dos relacionamentos (linhas 15–27). Por fim, os elementos <individual> apresentam as instâncias desta ontologia.

O que vale observar neste exemplo são os relacionamentos inversos: *Pedro diz que seu pai é Paulo* (linhas 32–33), enquanto *Paulo diz que é pai de Pedro* (linhas 40–41).

1.6.4 SHOE

SHOE [Luke and Helfin, 2000] é uma linguagem de representação de conhecimento baseada em HTML que adiciona etiquetas necessárias para a representação semântica de dados em páginas Web. SHOE é uma extensão do HTML que permite:

- definição de ontologias;
- definição de regras (através de cláusulas Horn);
- fazer anotações em páginas HTML a partir de propriedades de uma ou mais ontologias.

As etiquetas de SHOE estão divididas em duas categorias:

1. Etiquetas para construir ontologias. As ontologias em SHOE são conjuntos de regras que definem que tipo de declarações podem fazer os documentos SHOE e o que significam estas declarações. Por exemplo, uma ontologia SHOE pode decidir que uma entidade de dados é um *automóvel* e isso significa que *automóvel* pode possuir um *nome*;
2. etiquetas que permitam anotar documentos Web que pertençam a uma ou mais ontologias, declarar entidades e criar declarações sobre essas entidades sob as regras estabelecidas pelas ontologias. Por exemplo, um documento SHOE que se pertença à ontologia descrita anteriormente pode declarar um automóvel chamado *Audi A3*.

SHOE oferece uma base de conhecimento semântico, pois foi projectada tendo em mente as necessidades da busca de informação na Web. Ela possui várias formas de poder gerir os dados que se distribuem e está orientada principalmente a facilitar o trabalho dos agentes da Web e aos agentes inteligentes de maneira que possam tornar as pesquisas na Web mais completas. Entretanto, possui uma semântica limitada.

Um exemplo simples de como criar axiomas em SHOE é visto a seguir:

```
<DEF-INFERENCE DESCRIPTION= "irmaos(?pes1,?pes2) if tem pai (pes1,?pessoa) and tem
pai(pes2,?pessoa)">
  <INF-IF>
    <RELATION NAME="tem pai">
        <ARG POS=1 VALUE="pes1" USAGE=VAR>
        <ARG POS=2 VALUE="pessoa" USAGE=VAR>
     </RELATION>
    <RELATION NAME="tem pai">
        <ARG POS=1 VALUE="pes2" USAGE=VAR>
        <ARG POS=2 VALUE="pessoa" USAGE=VAR>
     </RELATION>
  </INF-IF>
  <INF-THEN>
     <RELATION NAME="irmaos"
        <ARG POS=1 VALUE="pes1" USAGE=VAR>
        <ARG POS=2 VALUE="pes2" USAGE=VAR>
     </RELATION>
  </INF-THEN>
</DEF-INFERENCE>
```

O exemplo acima apresenta a definição de uma regra que define se duas pessoas são irmãos: será sempre verdade se ambas possuírem o mesmo pai.

1.6.5 OWL

OWL (*Web Ontology Language*) [Patel-Schneider et al., 2004] é um trabalho em andamento no W3C, desde Março de 2002, que faz parte da actual lista de recomendações da W3C para o desenvolvimento da *Semantic Web* [Patel-Schneider et al., 2004]:

XML: fornece a sintaxe para descrever a estrutura de documentos, mas não dispõe de uma componente semântica que dê significado aos documentos;

XML Schema: é uma linguagem para definir a sintaxe e ainda restrições relacionadas com estruturas de documentos XML;

RDF: é um modelo de dados para objectos (recursos) e relações existentes entre eles, fornecendo uma semântica simples para esse modelo, o qual pode ser representado através de uma sintaxe XML;

RDF Schema: é uma notação para descrever as propriedades e classes dos recursos de RDF, com uma semântica referente a hierarquias e generalizações de tais classes e propriedades;

OWL: completa a linguagem anterior, adicionando-lhe construções para a descrição das classes e propriedades como, por exemplo, relações entre estas classes, cardinalidades, igualdades, tipos e características mais apurados; ou seja, permite definir um maior número de restrições semânticas que a rede de recursos de informação deve respeitar.

OWL é utilizada para representar explicitamente o conjunto de termos de um vocabulário e os relacionamentos entre estes termos. OWL é uma linguagem para a descrição de ontologias desenvolvida para o uso de aplicações para a Web que necessitam processar o conteúdo de recursos de informação disponíveis na Web, em vez de somente apresentá-las aos utilizadores; ou seja, OWL é utilizada quando as informações contidas em documentos precisam ser processadas por aplicações, ao contrário do que ocorre quando estes conteúdos apenas são mostrados aos utilizadores humanos que então procedem à sua interpretação.

OWL possui três sub-linguagens, que foram projectadas conforme o grupo de implementadores e utilizadores: OWL Lite, OWL DL e OWL Full [McGuinness and van Harmelen, 2004].

OWL Lite: hierarquia e características simples de restrição. Ex: valores de cardinalidade limitados a 0 e 1;

OWL DL: maior expressividade sem perda de poder computacional e capacidade de decisão dos sistemas. Inclui todos os construtores do OWL;

OWL Full: nível máximo de expressividade e liberdade sintáctica do RDF. Ex: uma classe pode ser tratada simultaneamente como uma colecção de elementos individuais e como um elemento individual.

OWL oferece uma maior facilidade para expressar a semântica que XML, RDF e RDF Schema, pois supera estas linguagens em sua habilidade em representar, de forma legível, o conteúdo na Web. OIL e DAML iniciaram como projectos de pesquisa, e com o tempo fundiram-se em DAML+OIL. Esta linguagem foi adaptada pela W3C e normalizada como OWL, a qual sucede às anteriores. Portanto, OWL é uma revisão das linguagens DAML (*DARPA Agent Markup Language*) e OIL (*Ontology Inference Layer*),

acrescentando-lhes características aprendidas no desenvolvimento e nas aplicações da DAML+OIL.

A sintaxe da linguagem OWL para definição de uma ontologia inicia com uma declaração, em RDF, contendo os *namespaces* que serão utilizados na ontologia. Basicamente, a ontologia começa da forma que se ilustra no exemplo seguinte [Bechhofer et al., 2004]:

```
1 <rdf:RDF
2    xmlns ="http://www.w3.org/2001/sw/WebOnt/guide-src/wine#"
3    xmlns:vin="http://www.w3.org/2001/sw/WebOnt/guide-src/wine#"
4    xmlns:food="http://www.w3.org/2001/sw/WebOnt/guide-src/food#"
5    xmlns:owl="http://www.w3.org/2002/07/owl#"
6    xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
7    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
8    xmlns:xsd="http://www.w3.org/2000/10/XMLSchema#">
```

As três primeiras declarações indicam o *namespace* associado à ontologia que está sendo desenvolvida que, no exemplo mostrado, se refere a vinhos e comidas. Na linha 5, é indicado o *namespace* específico da OWL e nas linhas subsequentes têm-se as referências tradicionais aos vocabulários específicos de RDF, RDF Schema e XML Schema, respectivamente.

Uma vez que os *namespaces* estão definidos, inclui-se, através do elemento <owl:Ontology>, informações como comentários, controle de versões e inclusão de outras ontologias, caso esteja havendo extensão de uma já existente. Abaixo apresenta-se a título de exemplo um fragmento do código OWL referente a estas informações [Smith et al., 2004]:

```
1 <owl:Ontology rdf:about="">
2    <rdfs:comment>Um exemplo de ontologia OWL</rdfs:comment>
3    <owl:priorVersion
4    rdf:resource="http://www.w3.org/2001/sw/WebOnt/guide-src/wine-112102.owl"/>
5    <owl:imports rdf:resource="http://www.w3.org/2001/sw/WebOnt/guide-src/food.owl"/>
6    ...
```

Têm-se no código acima a representação de um cabeçalho OWL contendo informações diversas representadas por:

- *rdfs:comment* apresenta algum comentário referente à ontologia modelada;
- *owl:priorVersion* é um elemento utilizado para o controle de versões da ontologia;
- *owl:imports* indica a importação de uma ontologia já existente definida pelo recurso apresentado em seguida.

As classes e subclasses OWL são definidas, respectivamente, através dos elementos *owl:Class* e *rdfs:subClassOf*. Por exemplo, a declaração de classe e sub-classe em OWL é realizada da seguinte forma:

```
1 <owl:Class rdf:ID="Curso"/>
2 <owl:Class rdf:ID="Licenciatura">
3    <rdfs:subClassOf rdf:resource="#Curso"/>
4    ...
5 </owl:Class>
```

No código OWL acima, mostra-se a declaração de duas classes: *Curso* (linha 1) e *Licenciatura* (linha 2 a 5). Sendo que, na linha 3, é especificado que *Licenciatura* é uma sub-classe de *Curso*.

Através do elemento *owl:ObjectProperty* define-se as propriedades das classes e subclasses. Utiliza-se – em conjunto com o elemento <ObjectProperty> – o elemento *rdfs:domain* para especificar a qual classe a referida propriedade está sendo atribuída. Abaixo, segue um simples exemplo da declaração de propriedades em OWL:

```
1 <owl:ObjectProperty rdf:ID="nome"/>
2    <rdfs:domain rdf:resource="#Curso"/>
3 </owl:ObjectProperty>
```

A propriedade *nome* está declarada na primeira linha e é pertencente à classe *Curso*, de acordo com a linha 2.

A OWL oferece vários elementos como *owl:Restriction*, *owl:cardinality*, *owl:minCardinality*, *owl:maxCardinality*, *owl:inverseOf*, entre diversos outros, os quais permitem determinar, respectivamente, as restrições, cardinalidades, cardinalidade mínima, cardinalidade máxima, relação inversa, de modo a obter uma ontologia mais completa e melhor modelada.

Para maiores detalhes sobre OWL, recomenda-se a leitura de [Bechhofer et al., 2002, Smith et al., 2004, McGuinness and van Harmelen, 2004, Bechhofer et al., 2004, Patel-Schneider et al., 2004, Daconta et al., 2003, Fensel, 2002, Coakes, 2003].

1.6.6 Topic Maps

A norma ISO 13250 *Topic Maps* define um formalismo para representar conhecimento acerca da estrutura de um conjunto de recursos de informação e para o organizar em tópicos. Esses tópicos possuem ocorrências e associações que representam e definem relacionamentos entre os tópicos. A informação sobre os tópicos pode ser inferida ao examinar as associações e ocorrências ligadas ao tópico. Uma colecção de tópicos e associações concretas é chamada topic map. Também pode ser visto como um paradigma que permite organizar, manter e navegar pela informação, permitindo transformá-la em conhecimento.

O formalismo para descrição de conhecimento Topic Maps assenta-se em três conceitos básicos, designados pela sigla *TAO* [Pepper, 2000]: Tópicos, Associações, e Ocorrências. Apesar da simplicidade desde triângulo basilar a abrangência da é tal que a definição possibilita representar estruturas complexas de informação de uma maneira intuitiva.

Nas sub-secções seguintes serão apresentados os elementos principais de Topic Maps (TAO), os quais estão ilustrados na Figura 1.13.

Os Topic Maps podem ser expressados usando XML. Para isto, um grupo de pesquisadores, liderados por Steve Pepper e Graham Moore, definiu a linguagem XTM (*XML Topic Maps*) [Pepper and Moore, 2001], criando o *TopicMaps.org*. *TopicMaps.Org*[18] é uma associação independente de grupos de desenvolvimento, projectando a aplicabilidade da norma Topic Maps para a internet com o uso das características da família de especificação XML. Portanto, XTM é o formato para intercâmbio (interoperabilidade) de Topic Maps entre aplicações.

[18] http://www.topicmaps.org

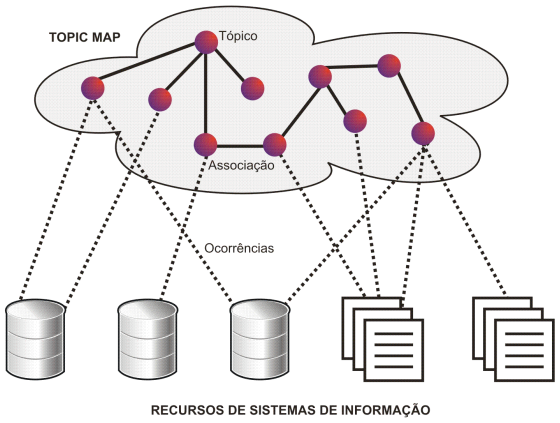

Figura 1.13: Ilustração simples da norma Topic Maps

1.6.6.1 A linguagem XTM

XTM é um dialecto XML criado formalmente para descrever todos os conceitos relativos a Topic Maps introduzidos pela norma ISO 13250 [Biezunsky et al., 1999]. Assim o respectivo DTD define as notações (*tags*) necessárias e suficientes para identificar os diferentes elementos que constituem um topic map.

1.6.6.2 Tópicos (*Topics*)

Um tópico em Topic Maps representa um tema em um domínio de aplicação. Por exemplo, no domínio *Futebol em Portugal*, podem ser tópicos: *Futebol Clube do Porto*, *Sporting de Braga*, *Sporting Lisboa e Benfica*, *Super Liga*, etc. Em termos técnicos, o relacionamento entre um tópico e um tema é definido como reificação[19].

Um tópico possui cinco características principais:

Identificador: permite que um tópico possa ser diferenciado dos demais tópicos, pois em um topic map o identificador é único;

Tipo: um tópico pode ser instância de outro tópico, definindo assim uma relação taxonómica;

Nomes: nomes (legíveis por humanos) alternativos para designar o conceito/tema em causa;

Identidade de tema: indica qual tema do universo de discurso um tópico reifica. Quando dois tópicos tiverem a mesma identidade de tema, considera-se que ambos identificam a mesma coisa, portanto devem ser unidos em um único tópico;

Ocorrências: recursos de informação que, embora externos, são relevantes para caracterizar/descrever o tópico.

Para exemplificar os conceitos aqui apresentados, será usado o domínio *Doutoramento em Informática*, onde serão necessários tópicos como os que se seguem para representar temas como: Departamento, Aluno, Professor, Tese, Escola e Universidade.

[19] Do inglês – *reify*: transformar algo abstracto em concreto. Reificação é o processo de criação de um tópico para um tema.

De acordo com a sintaxe XTM, cada tópico estará declarado em um elemento <topic>, como exemplificado a seguir:

```
1  <topic id="aluno">
2      <baseName>
3          <baseNameString>Aluno</baseNameString>
4      </baseName>
5  </topic>
6  <topic id="professor">
7      <baseName>
8          <baseNameString>Professor</baseNameString>
9      </baseName>
10 </topic> ...
```

No código XTM acima, são apresentados dois tópicos: *pessoa* (linha 1) e *professor* (linha 6). Ambos apresentam um nome cada (*baseName*), como característica.

1.6.6.3 Tipos de Tópicos (*Topic Types*)

Os tópicos podem ser categorizados dependendo do seu tipo (*topic types*). Em Topic Maps, qualquer tópico pode ser instância de zero ou mais tipos de tópicos, assim *Pedro Rangel Henriques* pode ser mapeado como um tópico do tipo *professor* e *Giovani Rubert Librelotto* como um tópico do tipo *aluno*. No universo de discurso de uma universidade, os tipos de tópicos podem ser: aluno, professor, grupo, departamento, artigo, etc.

A relação *tipo de tópico/tópico* pode ser vista como a relação genérica *super-classe/sub-classe*; embora no mundo dos Topic Maps seja encarado como a relação *classe/instância*, na realidade um tópico que é uma instância de um tipo de tópico pode, por sua vez, ser depois instanciado por outro tópico; ou seja, pode ser usado como tipo de tópico.

Um exemplo dessa definição afirma que *aluno* e *professor* são do tipo *pessoa*. Todas as características de *pessoa* são herdadas pelos tópicos instanciados por *aluno* e por *professor*. Uma representação disto encontra-se na sintaxe XTM encontrada abaixo:

```
1  <topic id="pessoa"/>
2  <topic id="professor">
3      <instanceOf>
4          <topicRef xlink:href="#pessoa"/>
5      </instanceOf>
6  </topic>
7  <topic id="aluno">
8      <instanceOf>
9          <topicRef xlink:href="#pessoa"/>
10     </instanceOf>
11 </topic>
12 <topic id="pedro-henriques">
13     <instanceOf>
14         <topicRef xlink:href="#professor"/>
15     </instanceOf>
16 </topic>
17 <topic id="giovani-librelotto">
18     <instanceOf>
19         <topicRef xlink:href="#aluno"/>
20     </instanceOf>
21 </topic>
```

Em XTM, as relações *tipo de tópico/tópico* são sempre definidas pelo elemento *<instanceOf>*. Por exemplo: o tipo de tópico de *professor* é uma instância de *pessoa* (linha 4); por sua vez, o tópico de *Pedro Rangel Henriques* é *professor* (linha 14),

definindo assim uma hierarquia de três níveis envolvendo os tópicos *pessoa*, *professor* e *Pedro Rangel Henriques*. O mesmo ocorre com os tópicos *pessoa*, *aluno* e *Giovani Rubert Librelotto*.

1.6.6.4 Ocorrência (*Occurrence*)

Um tópico pode estar ligado a um ou mais recursos de informação, os quais são relevantes ao tópico em questão de algum modo. Os tais recursos são chamados ocorrências de tópicos. Em termos técnicos, uma ocorrência é uma conexão que relaciona um tópico com um recurso. As ocorrências podem ser de dois tipos:

Referência ao recurso (*resourceRef*): Uma ocorrência *resourceRef* conecta o recurso relevante para o tópico usando a notação denominada XLink/XPointer [DeRose et al., 2001, DeRose et al., 2002] URI (*Universal Resource Identifier*) [IETF, 1998]. Esta referência funciona de forma similar a *hyperlinks* HTML na web. Consequentemente, todos os recursos endereçáveis por XLink/XPointer URI podem se tornar ocorrências em Topic Maps – significa que tudo o que pode ser endereçado na internet, pode ser ocorrência de tópico;

Referência a dados (*resourceData*): Uma ocorrência *resource data* associa um valor (expressado em literal) a um tópico. Isso pode ser usado para associar metadados aos tópicos. A norma Topic Maps não define qualquer tipo de dados para este tipo de recurso. Portanto, a interpretação da sequência de caracteres referente à informação do recurso é dada pela aplicação que interpretará o topic map.

Para exemplificar, considere-se as seguintes ocorrências do tópico que corresponde ao tema *Universidade do Minho*, designação que aqui se usa para identificar o tópico:

- O website da *Universidade do Minho* – recurso documental que tem imensa informação sobre a universidade e que por já existir fisicamente no endereço *http://www.uminho.pt* não vai ser aqui inserido, guardando-se apenas uma referência que será precisamente o URI (ver abaixo linhas 5-10, em particular a linha 9);
- O ano de fundação da *Universidade do Minho* – recurso de informação que, neste caso por se restringir a um único dado (o número 1973), vai ser aqui directamente inserido, escrevendo-se na ocorrência o respectivo valor (ver abaixo linhas 11-16, em particular a linha 15).

Lista abaixo, na sintaxe XTM, o exemplo da declaração do tópico *Universidade do Minho* com as suas duas ocorrências:

```
1  <topic id="uminho">
2      <baseName>
3          <baseNameString>Universidade do Minho</baseNameString>
4      </baseName>
5      <occurrence>
6         <instanceOf>
7            <topicRef xlink:href="#website"/>
8         </instanceOf>
9         <resourceRef xlink:href="http://www.uminho.pt"/>
10     </occurrence>
11     <occurrence>
```

```
12          <instanceOf>
13             <topicRef xlink:href="#ano-fundacao"/>
14          </instanceOf>
15          <resourceData>1973</resourceData>
16       </occurrence>
17 </topic>
```

1.6.6.5 Associação (*Association*)

Até agora, todos os conceitos discutidos foram referentes a tópicos como o princípio de organização básico para a informação. As definições *tópico*, *tipo de tópico* e *ocorrência* nos permitem organizar os recursos de informação, criando estruturas simples, que apenas individualizam os temas (ou conceitos) do domínio de aplicação; falta, agora, interligá-los.

Uma associação permite descrever relacionamentos entre tópicos. Uma associação é (formalmente) um elemento de vínculo que define um relacionamento entre dois ou mais tópicos. Conforme o universo de discurso apresentado nesta subsecção, alguns exemplos de associações podem ser:

- Aluno *escreve* artigos – artigos *são escritos por* alunos;
- Universidade *acolhe* alunos – alunos *estudam na* universidade;
- Professor *pertence a um* grupo – grupos *contém* professores;
- etc...

Para exemplificar, a sintaxe XTM para uma associação do tipo *aluno-escreve-artigo* (linha 3) entre *Giovani Librelotto* (linha 9), desempenhando o papel *aluno* (linha 7), e *Metamorphosis* (linha 15), desempenhando o papel *artigo* (linha 13), pode ser assim representada:

```
1  <association id="giovani-escreve-metamorphosis">
2     <instanceOf>
3        <topicRef xlink:href="#aluno-escreve-artigo"/>
4     </instanceOf>
5     <member>
6        <roleSpec>
7           <topicRef xlink:href="#aluno"/>
8        </roleSpec>
9        <topicRef xlink:href="#grl"/>
10    </member>
11    <member>
12       <roleSpec>
13           <topicRef xlink:href="#artigo"/>
14       </roleSpec>
15       <topicRef xlink:href="#metamorphosis"/>
16    </member>
17 </association>
```

As associações entre tópicos podem ser agrupadas de acordo com o seu tipo, da mesma forma como os tópicos e as ocorrências. Também como nos outros casos, o tipo de associação é um tópico. Assim, a definição de um tipo de associação é feita, na linguagem XTM, no elemento <*topic*>; como tal, informa o nome da associação, além dos nomes dos papéis de actuação dos seus membros. Cada nome encontra-se em um elemento <*baseName*>, dentro de um contexto particular (<*scope*>), conforme apresentado a seguir:

```
1  <topic id="aluno-escreve-artigo">
2     <baseName>
3        <baseNameString>Aluno escreve artigos</baseNameString>
4     </baseName>
5     <baseName>
6        <scope>
7          <topicRef xlink:href="#aluno"/>
8        </scope>
9        <baseNameString>escreve o artigo</baseNameString>
10    </baseName>
11    <baseName>
12       <scope>
13         <topicRef xlink:href="#artigo"/>
14       </scope>
15       <baseNameString>é escrito por</baseNameString>
16    </baseName>
17 </topic>
```

Na definição formal do tópico que corresponde ao tipo de associação acima *aluno-escreve-artigo* (linha 1), introduz-se na linha 3 o seu nome geral; na linha 9, o nome do papel que desempenhará o membro do tipo de tópico *aluno* (linha 7); por fim, na linha 15, o nome do papel que desempenhará o membro do tipo de tópico *artigo* (linha 13).

Assim, a leitura da associação identificada por *giovani-escreve-metamorphosis*, de acordo com seu tipo definido no tópico *aluno-escreve-artigo* é realizada da seguinte forma:

- Aluno Giovani escreve o artigo Metamorphosis.
- Artigo Metamorphosis é escrito pelo aluno Giovani.

1.6.6.6 TMCL

A norma ISO 19756 *Topic Map Constraint Language* (TMCL) [Moore et al., 2004] não é, na verdade, uma linguagem concreta, mas sim é uma lista de requisitos para a criação de uma linguagem formal para a definição de esquemas e restrições sobre Topic Maps. TMCL especifica restrições em topic maps em conformidade com o modelo de dados de Topic Maps (*Topic Maps Data Model* – TMDM) [Garshol et al., 2003]. A futura linguagem TMCL irá fornecer a base para uma linguagem formal para restrições, incluindo uma sintaxe.

TMCL também poderá ser utilizada na optimização do armazenamento de topic maps e nas consultas TMQL [Garshol and Barta, 2005]. Isso pode auxiliar na geração de interfaces intuitivas ao utilizador, para a criação e manutenção de topic maps.

TMCL possui uma série de requisitos [Nishikawa et al., 2004], dos quais destacam-se os seguintes:

Sintaxe XML: A linguagem TMCL deve possuir somente uma sintaxe normalizada XML;

Modelo e sintaxe: TMCL deve definir: (a) uma sintaxe para expressar restrições; (b) um modelo para representação interna destas restrições; (c) o comportamento da validação TMCL;

Restrições em características de tópicos: A linguagem deve ser capaz de expressar restrições em características de tópicos: nomes, ocorrências e identificadores de tema;

Conjunto de caracteres: O conjunto de caracteres para TMCL deve ser Unicode.

Em relação a sintaxe, TMCL deverá ter uma sintaxe amigável baseada em XML, que talvez possa ser baseada em XTM.

Os resultados de uma validação devem ser a confirmação de que o topic map satisfaz o conjunto de restrições fornecido. Caso alguma restrição não é satisfeita, então mensagens de erro devem ser mostradas. Essas mensagens de erro podem também incluir advertências (*warnings*), dependendo da regra de validação a ser aplicada.

TMCL não prevê a especificação de restrições sobre os identificadores de tópicos, assim como em que situações cada tipo de restrições pode ser aplicado. Além disso, TMCL não vai definir vocabulários explícitos para certos domínios; contudo, pode ser usada para propósitos ilustrativos em cenários de uso.

O progresso do desenvolvimento de TMCL está parado em discussão às dependências na família de normas de Topic Maps [Garshol, 2002, Biezunsky et al., 2003] e pode também ficar dependente de OWL [Bechhofer et al., 2002] e dos tipos de dados de XML-Schema [Biron, 2001].

No momento, o projecto de *Topic Maps Constraint Language* está sob desenvolvimento pelos membros do comité *ISO/IEC JTC 1/SC 34* [Moore et al., 2004]. Como se disse no início, a TMCL não é ainda uma linguagem concreta; é apenas uma lista se requisitos. O passo seguinte será a definição de uma linguagem que implemente todos os seus requisitos.

O mecanismo de restrições TMCL abrange todos os objectos[20] encontrados num topic map. Porém, alguns destes objectos devem possuir um conjunto maior de restrições. Dentre todos, o mais importante é a associação, pois ela representa a semântica principal de um topic map: as relações entre tópicos.

Como TMCL ainda não é uma linguagem concreta para a especificação de restrições sobre Topic Maps, algumas linguagens surgem, no momento, como alternativas para essa validação, como XTche [Librelotto et al., 2004a], AsTMa! [Barta, 2003] e OSL [Garshol, 2004b].

1.7 Ferramentas de suporte a Ontologias e à Semantic Web

Essa secção tem o objectivo de apresentar brevemente algumas ferramentas que permitem o desenvolvimento de ontologias a nível de: criação, visualização, dedução, navegação, edição, integração, compartilhamento, reutilização, etc.

1.7.1 Protégé

O *Protegé* [Protégé, 2005] é uma ferramenta gráfica para a edição de ontologias e aquisição de conhecimento. Ele inclui um mecanismo que permite a modelagem conceptual para várias das linguagens citadas anteriormente, tais como OWL, XTM e RDF.

[20] Por objectos em Topic Maps entendem-se todos os elementos básicos designados por TAO [Pepper, 2000] – tópicos, associações, ocorrências – e os elementos a eles associados – nomes, contextos, papéis de actuação em associações, actores em associações, etc.

Foi desenvolvida pela Universidade de Standford, e tem como características desenvolvimento e manutenção de sistemas de base de conhecimento além de modelo de representação baseado em frames, compatível com OKBC e exportação da representação em RDF(S).

A API do *Protégé* permite que aplicações utilizem, acedam e visualizem as bases de conhecimento criadas com o *Protégé*. Uma imagem referente ao *Protégé* está apresentada na Figura 1.14.

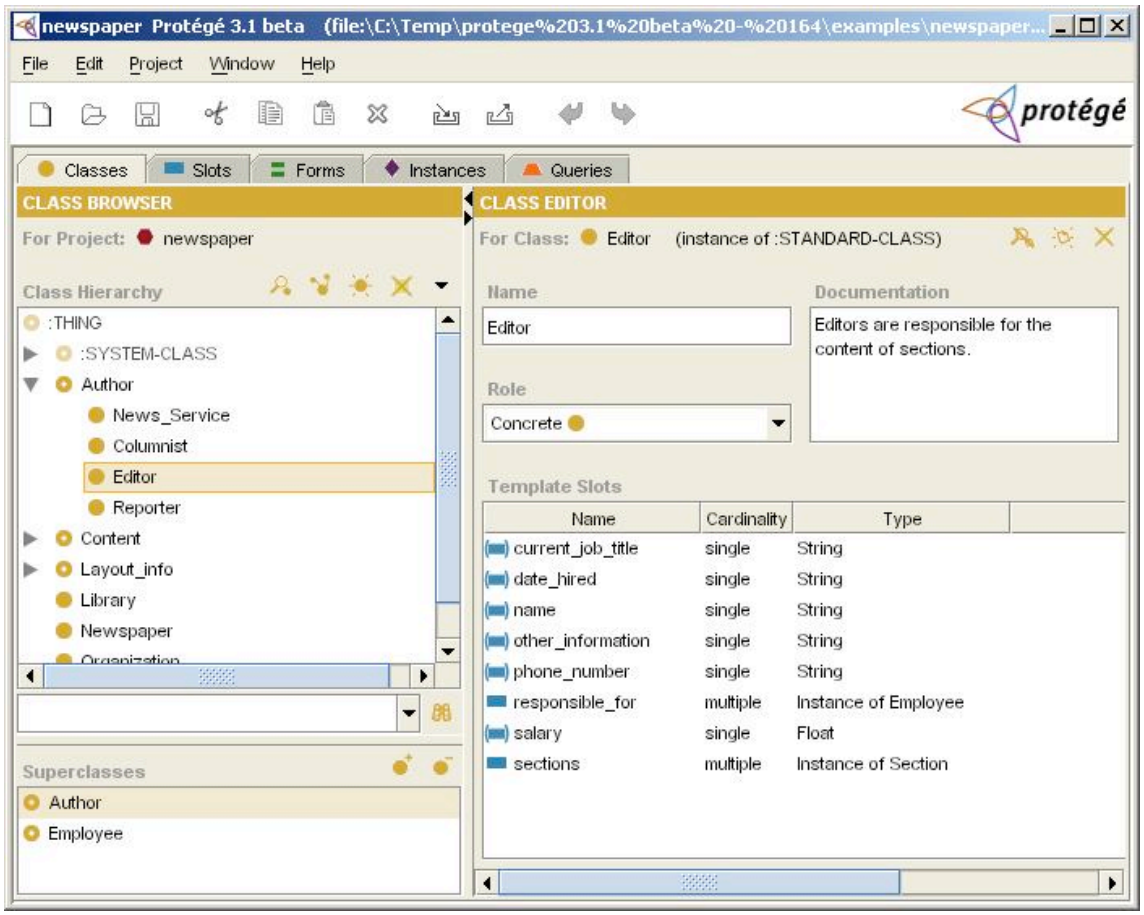

Figura 1.14: Protégé

O *Protégé* é um sistema integrado de desenvolvimento e gestão de bases de conhecimento, onde uma ontologia no *Protégé* consiste basicamente de:

Classes: conceitos do domínio abordado que constituem uma hierarquia taxonómica;
Slots: descrevem propriedades de classes e instâncias;
Facetas: descrevem propriedades de slots e permitem a especificação de restrições nos valores dos slots.

Para detalhes de como instalar e utilizar o *Protégé*, recomenda-se a URL *http://protege.stanford.edu/doc/users.html*, a qual é composta por uma FAQ (*Frequently Asked Questions*), guia de utilizador e uma série de tutoriais, artigos e apresentações sobre esse sistema.

1.7.2 OntoEdit

O *OntoEdit* [Mdche et al., 2000] foi desenvolvido pela Universidade Karlsruhe na Alemanha tem como características o desenvolvimento de projecto e manutenção de ontologias, suporte a ontologias multilingual e modelo de representação baseado em frames.

O *OntoEdit* também pode ser visto como um ambiente de engenharia de ontologias que possui três fases de desenvolvimento: especificação de requisitos, refinamento e avaliação [Felicíssimo et al., 2003].

Na fase de especificação de requisitos, estes são colectados e devem descrever o que a ontologia dará suporte. Por natureza, essa tarefa é realizada pelos especialistas do domínio acompanhados pelos especialistas de modelagem. Essa fase também deve gerar os subsídios que guiam o projectista da ontologia na decisão sobre os conceitos relevantes e sua estrutura hierárquica na ontologia.

Na fase de refinamento, uma ontologia madura é produzida e orientada à aplicação de acordo com a especificação dada na fase anterior. O engenheiro de ontologia pode desenvolver a hierarquia dos conceitos, propriedades e axiomas tão independente quanto seja possível da linguagem de representação concreta.

No *OntoEdit* o modelo conceptual da ontologia é armazenado de forma que seja possível fazer a transformação dessa representação conceptual para a maioria das linguagens de representação de ontologias como RDF(S), XML ou DAML+OIL. A fase de avaliação serve para provar a utilidade do desenvolvimento de ontologias e de seu ambiente de software associado. Nela, o engenheiro de ontologia checa se a ontologia corresponde às especificações do documento de requisitos.

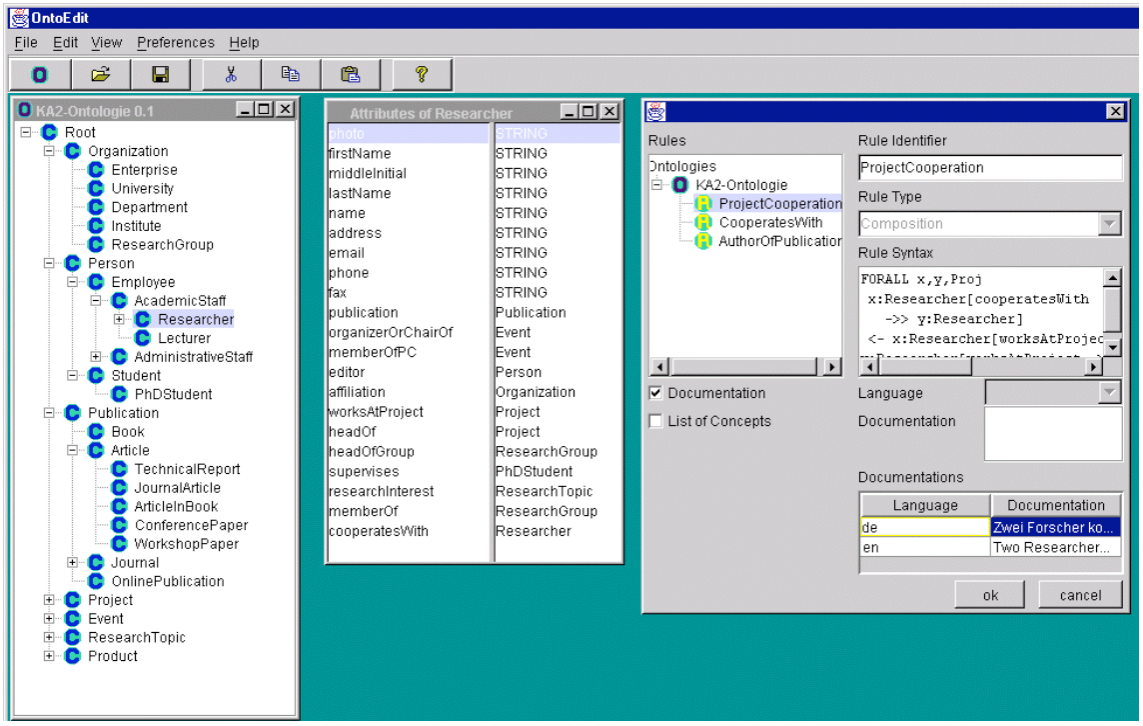

Figura 1.15: OntoEdit

Uma imagem referente ao *OntoEdit* pode ser visualizada na Figura 1.15. Para

detalhes mais aprofundados em relação ao *OntoEdit*, em relação à sua instalação e utilização, recomenda-se a leitura do seu tutorial [Ontoprise, 2003].

1.7.3 OntoClean

Uma metodologia geral e independente de domínio para construção de ontologias denominada *OntoClean* – criada por Guarino e Welty [Guarino and Welty, 2002] – busca fornecer uma espécie de guia, baseado em noções ontológicas gerais, provenientes da ontologia filosófica, sobre que decisões ontológicas precisam ser tomadas, bem como a maneira que estas decisões podem ser avaliadas.

A ferramenta *OntoClean* é baseada nas meta-propriedades rigidez, identidade, unidade e dependência [Guarino and Welty, 2000] que são utilizadas para caracterizar aspectos relevantes do significado pretendido de propriedades e relacionamentos que compõem uma ontologia e também impõem várias restrições em sua estrutura taxonómica. A essência do *OntoClean* consiste na associação de cada propriedade de um domínio a um conjunto de meta-propriedades que descrevem seu comportamento com respeito às noções ontológicas de identidade, unidade, essência e dependência. Após analisar cada propriedade, o modelador adquire uma melhor compreensão sobre a estrutura do domínio e a natureza do que está sendo representado pela ontologia.

A decisão sobre a posição de uma entidade em uma hierarquia consiste em uma das decisões ontológicas mais importantes que o modelador deve tomar ao se construir uma ontologia; e a possibilidade de avaliação de tais decisões, através de um fundamento formal, fornecido pela metodologia *OntoClean*, pode ser vista como um ponto importante em modelagem conceptual [Guarino and Welty, 2002].

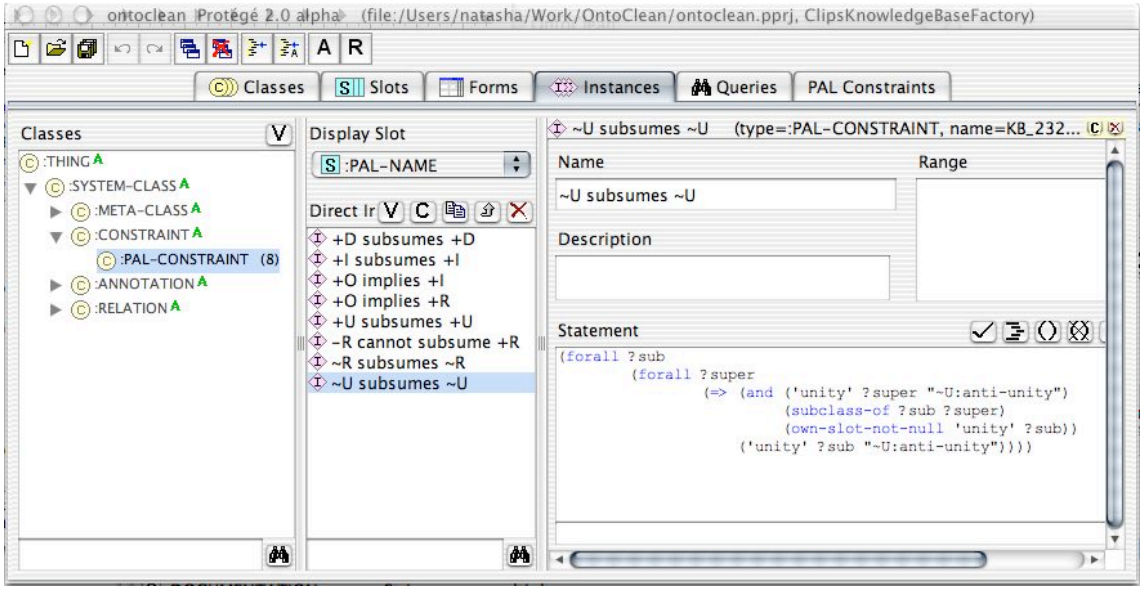

Figura 1.16: OntoClean

Apesar da aparente utilidade da metodologia *OntoClean*, a ser utilizada como ponto de partida para criação e validação de modelos conceituais, Maria Lúcia Villela [Villela and de Paiva Oliveira, 2003] constatou a ausência de trabalhos na literatura que fazem o seu uso, bem como das restrições por ela impostas.

1.7.4 Metamorphosis

O *Metamorphosis* [Librelotto et al., 2003b] é um ambiente orientado a Topic Maps composto por módulos capazes de extrair dados de recursos heterogéneos de informação, construir uma ontologia a partir dos mesmos (baseado em uma especificação), validá-la e gerar um sistema de navegação conceptual, permitindo atingir a interoperabilidade semântica entre as fontes.

A principal ideia do *Metamorphosis* é integrar a especificação de redes de conceitos ou ontologias, com sua navegação e armazenamento, assim como sua extracção automática e sua validação, a partir de recursos heterogéneos de informação.

O *Metamorphosis* toma como entrada:

Recursos de Informação: compostos por um ou mais recursos de dados: documentos XML, páginas HTML, bases de dados, etc. O *Metamorphosis* não modifica nenhuma fonte, apenas usa parte de seus dados para construir a rede semântica, através de uma ontologia definida para tais fontes de dados;

Especificações XML: a descrição das fontes de dados (escrita em XSDS – *XML Specification for DataSources*); a descrição da ontologia a ser construída (escrita em XS4TM – *XML Specification for Topic Maps*); e a descrição das regras a serem obedecidas por um topic map (estrita em XTche – *Topic Maps Schema and Constraint Language*).

e gera, como saída:

Um Website Conceptual: O website gerado permite a navegação através do sistema de informação dirigido por conceitos organizados em uma rede semântica.

A Figura 1.17 vem reforçar esta ideia e dá uma visão pictórica da arquitectura do *Metamorphosis*.

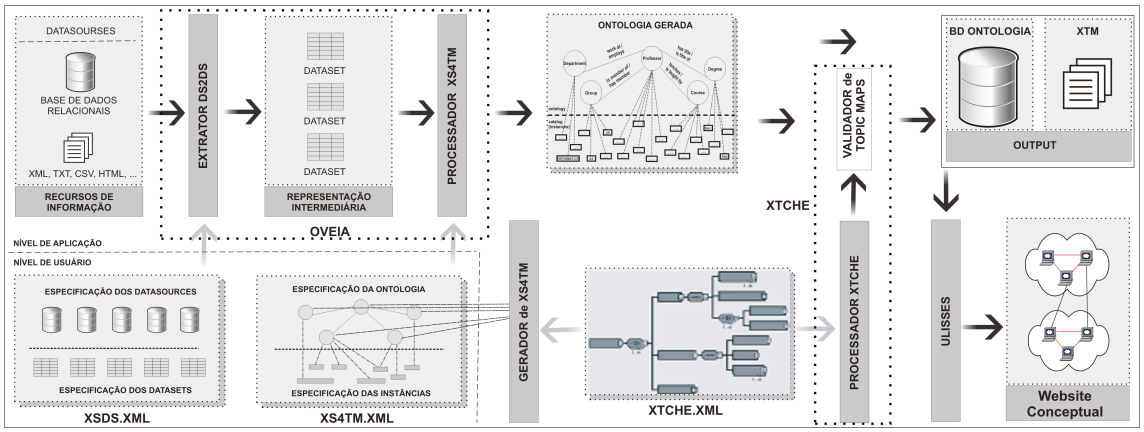

Figura 1.17: Metamorphosis

Esta arquitectura pode ser descrita da seguinte forma:

(1) Camada de recursos de informação: Este componente é composto pelos vários recursos de informação: documentos XML, páginas Web, bases de dados, ... O *Metamorphosis* não interfere com nenhum deles, apenas utiliza parte da informação de cada um para construir a ontologia ou rede semântica.

(2) Especificações XSDS e XS4TM: São documentos XML que fornecem informações precisas sobre onde aceder para extrair as partes da informação necessárias para a construção da ontologia; este componente é descrito em detalhes em [Librelotto et al., 2004b].

(3) Oveia: Este componente utiliza a especificação XS4TM para ir aos recursos de informação buscar a informação de que necessita para construir a ontologia. O *Oveia* [Librelotto et al., 2004b] é um extractor de topic maps em sistemas heterogéneos de informação.

(4) TM gerado: Este é o topic map de acordo com a sintaxe *XML Topic Maps*. Além da possibilidade de armazenamento dos topic maps em formato XTM, os topic maps gerados pelo *Oveia* também podem ser armazenados em uma representação relacional. Para isso, a norma Topic Maps foi mapeada para um modelo relacional; este modelo é chamado de *BD Ontologia*.

(5) Especificação XTche: É uma linguagem de especificação de restrições para topic maps, baseada nos requisitos de TMCL (*Topic Map Constraint Language*) [Nishikawa and Moore, 2003] definidos pela ISO/IEC. Permite especificar regras para a validação semântica de topic maps.

(6) Processador XTche: Este componente é responsável pela validação de topic maps de acordo com um conjunto de restrições especificado em XTche. Se os topic maps a serem validados estiverem de acordo com a especificação XTche, nada acontece; caso apresentem alguma irregularidade, mensagens de erros serão mostradas, indicando os pontos em que o topic map não está correto.

(7) Ulisses: Toma como entrada um *topic map* e produz uma visualização na web, de acordo com algumas regras. O *Ulisses* tanto fornece a navegação conceptual a partir da sintaxe XTM, como a partir do modelo relacional apresentado no *Oveia*.

(8) Website: É o website gerado através do qual é possível navegar semanticamente pelos vários recursos de informação que compõem o sistema de informação original.

O modo de funcionamento do *Metamorphosis* permite uma interdependência entre tais módulos; assim, quando o utilizador for efectuar o processamento de um conjunto de recursos de informação de acordo com especificações XSDS, XS4TM e XTche, cada etapa é realizada em separado:

1. no *Oveia* [Librelotto et al., 2004b], define-se os ficheiros com as especificações das fontes de informação e da ontologia a ser aplicada. Após o seu processamento, um sumário indica o tempo total e a quantidade de tópicos e associações extraídos que serão encontrados no topic map obtido;
2. no *XTche* [Librelotto et al., 2004a], indica-se o nome dos ficheiros que possuem o topic map a ser validado e o conjunto de restrições a ser aplicado. O resultado do processamento é a confirmação (ou não) da validação do topic map;
3. no *Ulisses* [Librelotto et al., 2003a], basta informar o topic map que será a fonte para a construção do website semântico.

O funcionamento independente dos módulos permite que uma tarefa específica seja refeita, caso necessário.

1.8 Conclusão

A *Semantic Web* é vista como uma nova geração da Web actual, a qual tem como finalidade atribuir um significado aos conteúdos publicados na Web de modo que seja perceptível tanto pelo humano como pelo computador. Para isto, as informações devem estar representadas de alguma forma que se possa extrair o conhecimento nelas inserido.

Com isto em mente, esse mini-curso teve por objectivo sugerir o uso de ontologias, um sistema de representação de conhecimento, como base para a Semantic Web. Para isso, inicialmente introduziu-se as definições de dado, informação e conhecimento, de modo a evitar ambiguidade nas secções subsequentes.

Para chegar à definição de ontologia, foram apresentados previamente outros modelos de representação de conhecimento que podem servir de base para uma ontologia, nomeadamente estruturas de facetas, dicionários, índices, taxonomias e thesauri. Conclui-se essa apresentação com uma secção sobre a conexão entre as várias abordagens, mostrando como elas se relacionam.

A fim de permitir uma codificação do conhecimento a ser representado na *Semantic Web*, a Secção 1.6 descreveu as principais linguagens para a especificação de ontologias, thesauri e taxonomias, no âmbito da *Semantic Web*: RDF, RDF Schema, XOL, SHOE, OWL e Topic Maps. Todas essas linguagens são baseadas em formato XML, o que justifica a apresentação da linguagem de anotação XML, realizada na Secção 1.5.

A Secção 1.7 introduziu alguns dos principais ambientes e ferramentas para a edição e processamento de ontologias para a *Semantic Web*, representadas nas linguagens descritas ao longo deste documento.

De um modo geral e sucinto, pretendeu-se ao longo deste mini-curso clarificar o que é uma ontologia no âmbito da *Semantic Web*, mostrar como a representar e quais as ferramentas para seu processamento.

Referências

Ahmed, K., Ayers, D., Birbeck, M., Cousins, J., Dodds, D., Lubell, J., Nic, M., Rivers-Moore, D., Watt, A., Worden, R., and Wrightson, A. (2001). *Professional XML Meta Data*. Wrox Programmer to Programmer Series.

Barta, R. (2003). AsTMa! Bond University, TR. `http://astma.it.bond.edu.au/constraining.xsp`.

Bechhofer, S., Hendler, J., Horrocks, I., Patel-Schneider, P. F., and Stein, L. A. (2004). Web Ontology Language (OWL) – Reference. `http://www.w3.org/TR/owl-ref/`.

Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D., Patel-Schneider, P., and Stein, L. A. (2002). Web Ontology Language (OWL) Reference Version 1.0. World Wide Web Consortium. `http://www.w3.org/TR/owl-ref/`.

Beckett, D. and McBride, B. (2004). RDF/XML Syntax Specification (Revised). World Wide Web Consortium. `http://www.w3.org/TR/rdf-syntax-grammar/`.

Berners-Lee, T. (2000). W3C – Semantic Web – XML 2000. http://www.w3.org/2000/Talks/1206-xml2k-tbl/.

Berners-Lee, T., Hendler, J., and Lassila, O. (2001). The Semantic Web. In *Scientific American*. http://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC%588EF21.

Biezunsky, M., Bryan, M., and Newcomb, S. (1999). ISO/IEC 13250 - Topic Maps. ISO/IEC JTC 1/SC34. http://www.y12.doe.gov/sgml/sc34/document/0129.pdf.

Biezunsky, M., Bryan, M., and Newcomb, S. (2003). Summary of Voting on SC 34 N0358 Restatement of Topic Maps. ISO/IEC 13250:2002 ISO/IEC JTC1 SC34 N0388. http://www.y12.doe.gov/sgml/sc34/document/0388.htm.

Biron, P. V. (2001). XML Schema part 2: Datatypes. http://www.w3.org/TR/xmlschema-2.

Bray, T., Hollander, D., and Layman, A. (1999). Namespaces in XML. World Wide Web Consortium. http://www.w3.org/TR/REC-xml-names/.

Bray, T., Paoli, J., Sperberg-McQueen, C. M., and Maler, E. (2000). Extensible Markup Language (XML). World Wide Web Consortium. http://www.w3.org/TR/REC-xml.

Brickley, D. (2000). *RDF Specifications: Containing Resource Description Framework RDF Schema and Resource Description Framework RDF Model and Syntax Specification*. iUniverse.com.

Brickley, D. and Guha, R. V. (2000a). Resource Description Framework (RDF) Schema Specification 1.0. World Wide Web Consortium. http://www.w3.org/TR/2000/CR-rdf-schema-20000327/.

Brickley, D. and Guha, R. V. (2000b). Resource Description Framework (RDF) Schema specification 1.0. World Wide Web Consortium. http://www.w3.org/TR/2000/CR-rdf-schema-20000327/.

Chandrasekaran, B. (1999). What Are Ontologies, and Why do We Need Them? In *IEEE Inteligent Systems and their applications*, volume vl 9, n 1. IEEE.

Coakes, E. (2003). *Knowledge Management: Current Issues and Challenges*. Idea Group Publishing.

Connolly, D., van Harmelen, F., Horrocks, I., McGuinness, D. L., Patel-Schneider, P. F., and Stein, L. A. (2001). A series of notes covering DAML+OIL as W3C technical reports. World Wide Web Consortium. http://www.w3.org/TR/daml+oil-reference.

Daconta, M. C., Obrst, L. J., and Smith, K. T. (2003). *The Semantic Web : A Guide to the Future of XML, Web Services, and Knowledge Management*. John Wiley & Sons.

DARPA (2001). DAML. Darpa Agent Markup Language Program. http://www.daml.org/.

de Almeida, J. J. D. (2003). *Dicionários Dinâmicos Multi-Fonte*. PhD thesis, Departamento de Informática, Escola de Engenharia, Universidade do Minho.

DeRose, S., Jr., R. D., Grosso, P., Maler, E., Marsh, J., and Walsh, N. (2002). XML Pointer Language (XPointer). http://www.w3.org/TR/xptr.

DeRose, S., Maler, E., and Orchard, D. (2001). XML Linking Language (XLink) - Version 1.0. http://www.w3.org/TR/xlink.

Duckett, J., Griffin, O., Mohr, S., Norton, F., Ozu, N., Stokes-Rees, I., Tennison, J., Williams, K., and Cagle, K. (2001). *Professional XML Schemas*. Wrox Press.

Farquhar, A., Fikes, R., and Rice, J. (1996). The Ontolingua Server: A tool for Collaborative Ontology Construction. In *KWA96*. Banff, Canada.

Felicíssimo, C. H., da Silva, L. F., Breitman, K. K., and do Prado Leite, J. C. S. (2003). Geração de ontologias subsidiada pela engenharia de requisitos. In *WER*, pages 255–269. http://wer.inf.puc-rio.br/WERpapers/artigos/artigos_WER03/carolina_feli%cissimo.pdf.

Fellbaum, C. (1999). *WORDNET: an electronic lexical database and some of its applications*. Cambridge, MA: MIT Press.

Fensel, D. (2002). *Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential*. MIT Press.

Fensel, D., van Harmelen, F., Horrocks, I., McGuinness, D., and Patel-Schneider, P. F. (2001). Oil: An ontology infrstructure for the semantic web. In *IEEE Intelligent Systems*, volume 16(2), pages 38–44. IEEE.

Fikes, R. and McGuinness, D. (2001). An Axiomatic Semantics for RDF, RDF-S, and DAML+OIL. World Wide Web Consortium. http://www.w3.org/TR/daml+oil-axioms.

Fowler, M. and Scott, K. (2000). *UML Essencial*. Bookman.

Frost, R. (1986). *Introduction to Knowledge Base Systems*. New York: MacMillan Publishing.

Garshol, L. M. (2002). Guide to the topic map standardization process. ISO/IEC JTC1 SC34 N0323. http://www.y12.doe.gov/sgml/sc34/document/0323.htm.

Garshol, L. M. (2004a). Metadata? Thesauri? Taxonomies? Topic Maps! Ontopia. http://www.ontopia.net/topicmaps/materials/tm-vs-thesauri.html.

Garshol, L. M. (2004b). The Ontopia Schema Language – Reference Specification. http://www.ontopia.net/omnigator/docs/schema/spec.html.

Garshol, L. M. and Barta, R. (2005). Topic Map Constraint Language (TMCL). ISO/IEC JTC1/SC34. http://www.isotopicmaps.org/tmql/spec.html.

Garshol, L. M., Moore, G., Newcomb, S. R., Biezunski, M., and Bryan, M. (2003). Topic Maps – Data Model.

Gruber, T. R. (1993). Towards Principles for the Design of Ontologies Used for Knowledge Sharing. In Guarino, N. and Poli, R., editors, *Formal Ontology in Conceptual Analysis and Knowledge Representation*, Deventer, The Netherlands. Kluwer Academic Publishers.

Gruber, T. R. (1995). How to Design an Ontology. In *Web page as part of the Ontolingua guided tour*. Accessed in April 2002.

Guarino, N. (1997). Semantic Matching: Formal Ontological Distinctions for Information Organization, Extraction, and Integration. In *Information Extraction: A Multidisciplinary Approach to an Emerging Information Technology*, pages 139–170, `http://www.ladseb.pd.cnr.it/infor/Ontology/Papers/SCIE97.pdf`. Springer Verlag.

Guarino, N. and Welty, C. (2000). Towards a methodology for ontology based model engineering. In *Proceedings of the ECOOP-2000 Workshop on Model Engineering*.

Guarino, N. and Welty, C. (2002). Evaluating Ontological Decisions with ONTOCLEAN. In *Communications of the ACM*, volume 5(2), pages 61–65.

Hodges, W. (1997). *A Shorter Model Theory*. Cambridge University Press.

IETF (1998). Uniform Resource Identifiers (URI). Internet Engineering Task Force. `http://www.ietf.org/rfc/rfc2396.txt`.

ISO (1986). ISO 2788:1986 – Documentation – Guidelines for the establishment and development of monolingual thesauri.

Karp, P. D., Chaudhri, V. K., and Thomere, J. (1999). Xol: an xml-based ontology exchange language. http://www.ai.r.sri.com/pkarp/xol.

Kay, M. (1979). Functional grammar. *5th Annual Meeting of the Berkeley Linguistic Society*.

Kifer, M., Lausen, G., and Wu, J. (1995). Logical Foundations of Object-Oriented and Frame-Based Languages. In *Journal of the ACM*.

Lamport, L. and Paulson, L. C. (1999). Should your specification language be typed. *ACM Trans. Program. Lang. Syst.*, 21(3):502–526.

Lassila, O. and Swick, R. R. (1999). Resource Description Framework (RDF) Model and Syntax Specification. World Wide Web Consortium. `http://www.w3.org/TR/REC-rdf-syntax`.

Lenat, D. B. and Guha, R. V. (1990). Building large knowledge-based systems. Representation and Inference in the yc Project. In *Addison-Wesley, Readings, Massachusetts*.

Librelotto, G. R., Ramalho, J. C., and Henriques, P. R. (2003a). Ontology driven Websites – Metamorphosis: a framework to specify and manage ontology driven websites. In *VII International Conference on Electronic Publishing*, Guimarães, Portugal.

Librelotto, G. R., Ramalho, J. C., and Henriques, P. R. (2003b). Ontology driven Websites with Topic Maps. In *The International Conference on Web Engineering*, Oviedo, Spain.

Librelotto, G. R., Ramalho, J. C., and Henriques, P. R. (2004a). XTche - A Topic Maps Schema and Constraint Language. In *XML 2004 Conference and Exposition*, Washington D.C., U.S.A. IDEAlliance.

Librelotto, G. R., Souza, W., Ramalho, J. C., and Henriques, P. R. (2004b). Using the Ontology Paradigm to Integrate Information Systems. In *International Conference on Knowledge Engineering and Decision Support*, pages 497–504, Porto, Portugal.

Luke, S. and Helfin, J. (2000). Shoe 1.01 proposed specification. SHOE Project. http://www.csumd.edu/projects/plus/SHOE/spec.html.

McGregor, R. (1991). Inside the LOOM classifier. In *SIGART bulletin*.

McGuinness, D. L. and van Harmelen, F. (2004). Web Ontology Language (OWL) – Overview. `http://www.w3.org/TR/owl-features/`.

Mdche, A., Schurr, P., Staab, S., and Studer, S. (2000). The otk tool repository - ontoedit. `http://www.ontoknowledge.org/tools/ontoedit.shtml`.

Moore, G., Nishikawa, M., and Bogachev, D. (2004). Topic Map Constraint Language. ISO/IEC JTC 1/SC 34. `http://www.jtc1sc34.org/repository/0549.htm`.

Motta, E. (1999). Reusable Components for Knowledge Modeling. In *IOS Press*.

Mylopoulos, J., Borgida, A., Jarke, M., and Koubarakis, M. (1990). Telos: representing knowledge about information systems. *ACM Trans. Inf. Syst.*, 8(4):325–362.

Nishikawa, M. and Moore, G. (2003). Topic Map Constraint Language (TMCL) Requirements and Use Cases. ISO/IEC JTC 1/SC34 N0405rev. `http://www.isotopicmaps.org/tmcl/requirements.html`.

Nishikawa, M., Moore, G., and Bogachev, D. (2004). Topic Map Constraint Language (TMCL) Requirements and Use Cases. ISO/IEC JTC 1/SC34 N0548. `http://www.jtc1sc34.org/repository/0548.htm`.

Noy, N. F. and Mcguinness, D. L. (2001). Ontology Development 101: A Guide to Creating Your First Ontology. Stanford University, Stanford. `http://www.ksl.stanford.edu/people/dlm/papers/ontology101/ontology101-n%oymcguinness.html`.

OMG (2003). OMG Unified Modeling Language Specification. Object Management Group. `http://www.omg.org/docs/formal/03-03-01.pdf`.

Ontoprise (2003). How to work with ontoedit. `http://www.ontoprise.de/documents/tutorial_ontoedit.pdf`.

Park, J. and Hunting, S. (2003). *XML Topic Maps: Creating and Using Topic Maps for the Web*, volume ISBN 0-201-74960-2. Addison Wesley.

Patel-Schneider, P. F., Hayes, P., Horrocks, I., and van Harmelen, F. (2004). Web Ontology Language (OWL) – Abstract Syntax and Semantics. `http://www.w3.org/TR/2004/REC-owl-semantics-20040210/`.

Pepper, S. (2000). The TAO of Topic Maps - finding the way in the age of infoglut. Ontopia. `http://www.ontopia.net/topicmaps/materials/tao.html`.

Pepper, S. and Moore, G. (2001). XML Topic Maps (XTM) 1.0. TopicMaps.Org Specification. `http://www.topicmaps.org/xtm/1.0/`.

Powers, S. (2003). *Practical RDF*. O'Reilly & Associates, 1st edition edition.

Prescott, L. M., Harley, J. P., and Harley, J. (1996). *Microbiology*. Wm. C. Brown Pub. Dubuque, Iowa, third edition edition. pp. 390-414.

Protégé (2005). The protégé ontology editor and knowledge acquisition system. http://protege.stanford.edu/.

Ramalho, J. C. and Henriques, P. (2002). *XML & XSL Da Teoria à Prática*. FCA Editora.

Ricker, J. (2000). *Managing Metadata With XML and RDF: Improving Workflow for Web Applications*. John Wiley & Sons.

Schreiber, G., Wielinga, B., Akkermans, H., van de Velde, W., and Anjewierden, A. (1994). CML: The CommonKADS Conceptual Modeling Language. In et al., S., editor, *A Future of Knowledge Acquisition, Proc. 8th European Knowledge Acquisition Workshop (EKAW94), Hoegaarden*. Lecture Notes in Artificial Intelligence 867, Springer-Verlag.

Shannon, C. E. (1974). A Mathematical Theory of Communication. In *Key Papers in the Development of Information Theory*. New York: IEEE.

Smith, M. K., Welty, C., and McGuinness, D. (2004). Web Ontology Language (OWL) – Guide Version 1.0. http://www.w3.org/TR/2004/REC-owl-guide-20040210/.

Sowa, J. F. (2000). *Knowledge Representation: logical, philosophical and computational foundations*. Brooks/Cole.

Staab, S., Erdmann, M., Maedche, A., and Decker, S. (2000). An extensible approach for modeling ontologies in RDF(S). ECDL 2000 Workshop on the Semantic Web. http://www.y12.doe.gov/sgml/sc34/document/0391.htm.

Swartout, W. and Tate, A. (1999). Ontologies. In *IEEE Inteligent Systems and their applications*, volume vl 14, n 1. IEEE.

UNESCO (1995). UNESCO Thesaurus. United Nations Educational, Scientific and Cultural Organization. http://www.ulcc.ac.uk/unesco/.

van Dalen, D. (1994). *Logic and Structure*. ISBN: 3-540-57839-0. Springer, 3rd augmented edition, (corrected 2nd printing 1997) edition.

Villela, M. L. B. and de Paiva Oliveira, A. (2003). Validação de diagramas de classe por meio de propriedades ontológicas. http://www.dcc.ufmg.br/pos/html/spg2003/anais/marialucia-ufv.htm.

W3C (2005). World Wide Web Consortium. http://www.w3.org.