Geração automática de interfaces Web para Sistemas de Informação Metamorphosis

José C. Ramalho[1] **, Giovani R. Librelotto**[1*]**, Pedro R. Henriques**[1]

[1]Departamento de Informática – Universidade do Minho
Campus de Gualtar – 4710-057 – Braga – Portugal

{jcr,grl,prh}@di.uminho.pt

Abstract. *Hoje em dia a presença na Web é quase mandatória para todos os intervenientes na sociedade da informação. Assim, não é estranho que muitas empresas e instituições dediquem uma grande fatia do seu tempo ao desenvolvimento e manutenção de* websites*. Esta tarefa tem sido uma grande consumidora de tempo e recursos.*
O que hoje é novidade é normalmente tido como garantido no dia seguinte pelos utilizadores. Os utilizadores nunca estão satisfeitos querem sempre mais: informação actualizada, acessos personalizados, ...
Para responder a estes requisitos os websites *têm de ser dinâmicos e têm de ser capazes de reconfigurar automaticamente a sua estrutura, o seu conteúdo e o seu aspecto visual.*
Este cenário tem favorecido a criação de ferramentas para geração automática e manutenção de websites*.*
Neste artigo, não propomos mais uma ferramenta desta espécie mas sim uma nova abordagem ao problema.
Nesta nova abordagem o sistema é dividido em duas grandes camadas. Uma camada física que corresponde ao sistema de informação que se quer expôr na Web, que normalmente é constituído por bases de dados, documentos XML, partes do sistema de ficheiros e toda a panóplia de ficheiros nos mais variados formatos que se possa imaginar, e que foi denominada por camada de recursos. Uma segunda camada de metainformação, designada por ontologia, que fornece uma panorâmica estruturada sobre os recursos de informação.
O ambiente ainda em desenvolvimento e baptizado de Metamorphosis*, é composto por vários componentes. Neste artigo iremos abordar o componente* TMBuilder *que extrai automaticamente a ontologia dos recursos de informação, e outro componente* DINavigator *que a partir da ontologia gera automaticamente o* website*. Todo o ambiente é baseado na tecnologia XML, desta maneira garante-se a portabilidade e a independência relativamente a plataformas de hardware e software.*

1. Introdução

Todos os dias, milhares de recursos de informação são colocados na Web. Este facto faz com que a Web cresça exponencialmente a cada dia que passa tornando as tarefas de pesquisa de informação cada vez mais complexas e dificeis. Na tentativa de resolver este problema surgiram várias iniciativas e uma nova área de investigação tem emergido: *Semantic Web*.

[*]Bolsista CNPq - Brasil

Quando nos referimos a *Semantic Web* estamos a referir-nos a uma rede semântica de conceitos. Cada conceito está relacionado com um conjunto específico de recursos de informação e pode ser relacionado com outros conceitos. Esta rede de conceitos pode então ser utilizada para navegar por entre recursos Web ou simplesmente num sistema de informação que tenha uma interface Web.

A ideia principal subjacente ao *Metamorphosis* é a integração da especificação destas redes semânticas ou ontologias com a navegação por entre os recursos relacionados e o seu armazenamento.

O projecto de desenvolvimento do *Metamorphosis* teve como requisito inicial a utilização da tecnologia XML para suportar todos os componentes do sistema com vista a manter a portabilidade e independência de plataformas de todos os componentes.

A ideia de utilizar XML para a especificação de ontologias não é nova e várias linguagens de anotação definidas em XML foram desenvolvidas. Das várias iniciativas houve uma que ganhou o estatuto de norma ISO: Topic Map ISO 13250. No entanto, neste cenário, há linguagens de anotação que têm de ser consideradas, como: RDF [LS99], RDFS [BG00], DAML [DAR01], OIL [OIL02] ou OWL [BvHH$^+$02]. A mais genérica e abstracta é a *Topic Map*, ou a sua versão mais recente em XML, *XTM*. Relativamente à dimensão da comunidade de utilizadores rivaliza com o *RDF*. Os utilizadores duma e de outra têm normalmente os mesmos objectivos mas diferem na abordagem que fazem à resolução do problema que pretendem resolver: o XTM induz uma abordagem *top-down* enquanto que o RDF é utilizado em abordagens *bottom-up*.

Quando se está a desenvolver uma interface Web para um sistema de informação usa-se, normalmente, uma perspectiva *top-down*. Para além deste aspecto, pretendia-se uma linguagem o mais abstracta possível para permitir facilmente modificações e acrescentos. Com este requisitos, o XTM surge como a melhor opção para especificação de uma ontologia a partir da qual é gerado um website.

Há muitas ferramentas que usam ontologias para a geração automática de websites ou para navegar num conjunto de recursos. A maioria destas ferramentas estão implementadas numa linguagem tradicional de programação como o *Java*, o *Perl* ou o *Python*. No *Metamorphosis* obtivemos os mesmos resultados recorrendo apenas à tecnologia XML. O sistema é apenas composto por documentos XML, uns com informação, outros com transformações *XSL* [RH02]. Além disso, o sistema foi enriquecido com novos componentes e terá outros que ainda estão em fase de desenvolvimento que vêm acrescentar funcionalidades muito interessantes a uma ferramenta deste tipo.

Na próxima secção dá-se uma breve descrição dos conceitos técnicos necessários para entender o resto do artigo: Topic Maps, XML, e transformações de documentos XML (XSL). A secção 4. introduz a arquitectura do sistema desenvolvido. A seguir, a secção 5., discute a implementação do componente de navegação encarregue da geração automática de websites. Finalmente, a secção 6. apresenta um dos casos de estudo que foram utilizados para o desenvolvimento do sistema. O artigo termina com algumas conclusões e com a descrição do trabalho em curso.

2. Semantic Web e Ontologias

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 a uma enorme quantidade de informação.

Apesar da complexidade crescente da Web, isto não é refletido no estado actual das tec-

nologias utilizadas para a sua manipulação. A maior parte das tarefas de interpretação, acesso, extração 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, 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 [BLHL01], esse tipo de construção leva a limitações e a um tratamento trivial por parte dos atuais browsers, do conteúdo das páginas Web – limita-se a um cabeçalho, links para outras páginas; mas, em geral, as ferramentas não possuem uma forma confiável de processar o conteúdo semântico das informações contidas em uma página.

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. Basicamente, os mecanismos a serem desenvolvidos para o estabelecimento da *Semantic Web* compreendem duas vertentes: a disponibilização de uma coleção de dados estruturados e regras de inferência associadas a essa coleção; e a criação de ferramentas capazes de percorrer a Web realizando tarefas complexas com base nessas estruturas de conhecimento.

Esse conjunto coerente de coleções estruturadas de informação forma uma ontologia. Uma ontologia é uma teoria lógica para descrever o significado pretendido de um vocabulário formal, isto é, seu compromisso com uma conceitualização particular do mundo. Estas incluem estruturas que permitem manipular termos de uma forma muito eficiente e útil para o utilizador e mecanismos de validação para comunicação entre programas. A importância de seu uso é devida à capacidade de representar hierarquias de classes de objetos (taxonomias) e seus relacionamentos.

Dentro de uma mesma área podem ser encontradas diferentes definições e classificações de ontologia. Na área de Inteligência Artificial, Guarino define a ontologia como uma caracterização axiomática do significado do vocabulário lógico [Gua97] e para Sowa, a ontologia define os tipos de coisas que existem no domínio de uma aplicação [Sow00].

Na área de Sistemas de Informação, na qual se encaixa este trabalho, ontologia é definida como um conjunto de conceitos e termos que podem ser usados para descrever alguma área do conhecimento ou construir uma representação para o conhecimento [ST99]. Segundo Chandrasekaran [Cha99], ontologias são teorias de conteúdo sobre os tipos de objetos, propriedades de objetos e relacionamentos entre objetos que são possíveis num domínio de conhecimento específico.

O desenvolvimento de ontologias irá alimentar o mecanismo de construção da parte semântica da Semantic Web. O modelo em camadas proposto por Berners-Lee[1] tem sido aceite principalmente como representação para a arquitetura 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 a Web.

A fim de prover o primeiro mecanismo necessário à *Semantic Web*, a anotação da informação em XML (*eXtensible Markup Language*)[2] vem sendo reconhecida como relevante. XML permite representar dados em formato semi-estruturado, o que ocorre com frequência no mundo real. Entretanto, XML por si mesmo, não permite acrescentar significado a tais estruturas. Ao usar XML

[1] Disponível em http://www.w3.org/2000/Talks/1206-xml2k-tbl/slide10-0.html

[2] eXtensible Markup Language - Language Syntax Specification, http://www.w3.org/TR/REC-xml

como sintaxe para transmissão de dados semi-estruturados, a descrição do significado deve ficar a cargo de alguma linguagem de especificação semântica.

Na próxima secção, aborda-se a especificação da semântica.

3. Topic Maps

XML Topic Maps (XTM) [PM01] é um formalismo para representar conhecimento acerca da estrutura de um conjunto de recursos de informação e para o organizar em *tópicos*. Esses tópicos têm 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 coleção desses tópicos e associações é chamada *topic maps*. Também pode ser visto como um paradigma que permite organizar, manter e navegar pela informação, permitindo transformá-la em conhecimento.

Topic maps pode ser definido como uma descrição de um ponto de vista sobre uma coleção de recursos, organizado formalmente por tópicos, e pela ligação de partes relevantes do conjunto de informação aos tópicos apropriados [Pep00].

Um mapa de tópicos expressa a opinião de alguém sobre o que os tópicos são, e quais as partes do conjunto de informação que são relevantes para cada tópico. *Charles Goldfarb* [GP01] (o pai das linguagens de anotação) geralmente compara topic maps com GPS (*Global Positioning System*) aplicado ao universo da informação. Falar sobre topic maps é falar sobre estrutura de conhecimento.

Permitindo criar um mapa virtual da informação, os recursos de informação mantém-se em sua forma original e não são modificados. Então, o mesmo recurso de informação pode ser usado de diferentes maneiras, por diferentes mapas de tópicos. Como é possível e fácil modificar um mapa, a reutilização da informação é conquistada.

3.1. As características do modelo XTM

Tópicos são o ponto principal de topic maps [PHE03]. Em um sentido mais genérico, podem ser qualquer coisa: uma pessoa, uma entidade, um conceito. Eles constituem a base para a criação de topic maps. Podem ser vistos como um link-múltiplo, o qual aponta para todas as suas ocorrências [BN99].

Cada tópico tem um tipo de tópico (*topic type*), ou talvez múltiplos tipos. Cada tipo de tópico pode ser visto como uma típica relação classe-instância. Os tipos representam as classes onde os tópicos estão agrupadas, i.e., a categoria de cada instância tópico. Segundo o modelo de topic maps, os tipos de tópicos também são tópicos.

Um tópico pode ter um ou mais nomes. A opção de especificar mais de um nome ao tópico pode ser utilizada em diferentes contextos (*scopes*), como idiomas, estilos, domínios, área geográfica, período histórico, etc.

Um tópico pode ter uma ou mais ocorrências. Um ou mais recursos de informações endereçáveis de um tópico constituem o conjunto de ocorrências de tópicos (*topic occurrences*). As ocorrências de tópicos podem ser, por exemplo, um artigo, uma imagem ou vídeo. Uma ocorrência representa a informação que é especificada como relevante para um certo tópico [Pep00]. As ocorrências podem ser endereçáveis através de uma URI (*Universal Resource Identifier*).

Ocorrências e tópicos existem em duas camadas diferentes, mas elas são ”conectadas” entre si. As ocorrências estabelecem rotas dos os tópicos para os recursos de informação, possi-

bilitando também prover uma razão de o porque que a rota existe. Neste ponto, a separação em duas camadas é percebida: tópicos e suas ocorrências; uma das grandes vantagens de XTM.

Entre todas as ocorrências de um tópico, uma distinção pode ser feita através de subgrupos. Cada subgrupo é definido por um papel de atuação (*role*) em comum. Os papéis de atuação em ocorrências (*occurrence role*) podem ser utilizados para distinguir gráficos de texto, menções de definições, etc. Os papéis de atuação em ocorrências são definidos pelos usuários, sendo assim podem variar em cada topic maps [BN99]. Para fazer a real distinção entre diferentes tipos de ocorrências, topic maps também utiliza o conceito de tipo de papel de atuação em ocorrência (*occurrence role type*) [Pep00].

As associações (*associations*) são responsáveis pelos relacionamentos entre os tópicos. Elas são ligações independentes da fonte dos documentos onde as ocorrências de tópicos são encontradas; elas representam a base do conhecimento, a qual contem a essência da informação que alguém criou e atualmente representa seu valor essencial. Um ilimitado número de tópicos podem ser relacionados por uma associação.

O poder de topic maps aumenta com a criação de associações porque, deste modo, é possível agrupar um conjunto de tópicos que de algum modo são relacionados. Isso é de grande importância ao prover interfaces intuitivas e amigáveis para a navegação de grandes quantidades de informação.

Assim como os tipos de tópicos agrupam vários tópicos e tipos papéis de atuação também suportam várias ocorrências, as associações entre tópicos devem ser agrupadas de acordo com seu tipo de associação (*association type*). Cada tópico que participa em uma associação tem um papel (*association role*) que expressa a sua atuação nessa associação. Os papéis de atuação em associação também são vistos como tópicos no modelo XTM.

3.2. Como definir um Topic Map

Antes de iniciar, é necessário definir exatamente o que é que será representado no Topic Map. E para isso, existem duas partes: a definição do escopo do Topic Map , isto é, decidindo a extensão do domínio que ele deve cobrir; e o projeto da ontologia básica. Em Topic Maps, uma ontologia é uma precisa descrição dos tipos de coisas as quais são encontradas no domínio coberto pelo topic map. Em outras palavras, o conjunto de tópicos que é usado para definir classes de tópicos, associações, papéis de atuação e ocorrências. Este termo também é usado em outras áreas, tal como na filosofia, onde tem outro significado.

Para ilustrar todas as idéias introduzidas e para descrever o processo de construção de um TM, é apresentado o caso de estudo onde o assunto é uma workshop. Esta workshop é o *XATA - XML: Aplicações e Tecnologias Associadas* realizado na Universidade do Minho, em 2003. Considerando este exemplo, tem-se quatro tipos de tópicos: Instituições, Artigos, Inscritos e Sessões.

Nos exemplos que se segue, é assumido que o participante de nome *Pedro Rangel Henriques* está inscrito no XATA como pertencente ao Departamento de Informática da Universidade do Minho, e é autor do artigo *Especificação XML de Aplicações para WWW*, que foi apresentado na sessão *Dialectos XML*. A ontologia básica consiste dos tipos de tópicos descritos no parágrafo anterior, além dos papéis de associações *pertence/contém*, *está na sessão/contém o artigo* e é *autor de/foi escrito por*, assim como os tipos de associações *Artigo e Sessão*, *Inscrito e Instituição* e *Inscrito e Artigo*.

Primeiramente, é definido os tópicos (os tipos de tópicos e suas instâncias), especificando seus identificadores e seus nomes. Abaixo, está um exemplo incompleto:

```
<?xml version="1.0" encoding="UTF-8"?>
<topicMap xmlns="http://www.topicmaps.org/xtm/1.0"
```

```
          xmlns:xlink="http://www.w3.org/1999/xlink">
  <topic id="Inscrito">
    <baseName>
      <baseNameString>Inscrito</baseNameString>
    </baseName>
  </topic>
  <topic id="pedrorangelhenriques">
    <instanceOf>
      <topicRef xlink:href="#Inscrito"/>
    </instanceOf>
    <baseName>
      <baseNameString>Pedro Rangel Henriques</baseNameString>
    </baseName>
  </topic>
</topicMap>
```

Aqui apenas é demonstrado a definição do tópico *Pedro-Rangel-Henriques* e seu tipo *Inscrito*. Os outros tópicos e seus tipos são criados de uma forma similar.

Após os tópicos, é adicionado as ocorrencias referentes aos próprios tópicos, usando o elemento *resourceRef* que contém a URL (endereço para o recurso) como o valor do atributo *xlink:href*. Por exemplo, uma ocorrência para o tópico *Pedro Rangel Henriques* pode ser o seu e-mail, como mostrado abaixo:

```
<topic id="pedrorangelhenriques">
  ...
  <occurrence>
    <instanceOf>
      <topicRef xlink:href="#email"/>
    </instanceOf>
    <resourceRef xlink:href="prh@di.uminho.pt"/>
  </occurrence>
</topic>
```

Note que o uso de *#email* as a *;* isso é possível porque *#email* também é um tópico, mais precisamente um tipo de ocorrência.

O "..." do código acima segue a definição do tópico *Pedro Rangel Henriques* como apresentado no fragmento anterior; apenas não é repetido para tornar o exemplo mais simples.

Num terceiro passo, é definido as associações entre os tópicos, expressando seus tipos e seus membros (um tópico com um papel de atuação explícito). No exemplo abaixo, é definida associação Autor e Artigo entre *Especificação XML de Aplicações para WWW* e *Pedro Rangel Henriques*. O primeiro desempenha o papel de *é escrito por* e o segundo, o papel de *é autor de*.

```
<association>
  <instanceOf>
    <topicRef xlink:href="#Inscrito-e-Artigo"/>
  </instanceOf>
  <member>
    <roleSpec>
      <topicRef xlink:href="#ID-escrito_por"/>
    </roleSpec>
    <topicRef xlink:href="#especificacaoxmldeaplicacoesparawww"/>
  </member>
  <member>
    <roleSpec>
      <topicRef xlink:href="#ID-autor_de"/>
    </roleSpec>
    <topicRef xlink:href="#pedrorangelhenriques"/>
  </member>
</association>
```

As referências *ID-escrito_por* e *ID-autor_de* são papéis de atuação em associações e são declaradas como tipos de tópicos, isto é, com um identificador e um nome, somente. A referência *Inscrito-e-Artigo* é um tipo de associação que define o tipo desta associação. A declaração do tópico é mostrada abaixo:

```
<topic id="Inscrito-e-Artigo">
  <baseName>
    <baseNameString>Inscrito e Artigo</baseNameString>
  </baseName>
  <baseName>
    <scope>
      <topicRef xlink:href="#ID-escrito_por"/>
    </scope>
    <baseNameString>é escrito por</baseNameString>
  </baseName>
  <baseName>
    <scope>
      <topicRef xlink:href="#ID-autor_de"/>
    </scope>
    <baseNameString>é autor de</baseNameString>
  </baseName>
</topic>
```

O TM acima significa que *Pedro Rangel Henriques* é autor de *Especificação XML de Aplicações para WWW* e, de modo inverso, indica que *Especificação XML de Aplicações para WWW* foi escrito por *Pedro Rangel Henriques*.

As razões para o uso de topic maps são, primeiramente, o facto de que existem muitas ferramentas baseadas em topic maps, o que contribui para a distribuição do processo. Dentre essas ferramentas, pode-se citar as apresentadas na tabela 1.

Ferramenta	**Descrição**
Ontopia Omnigator	Browser conceptual sobre XTM. Necessita do TomCat.
DINavigator	Browser conceptual sobre XTM. Baseado em XSLT.
TMNav	Editor gráfico de XTM.
Mapalizer	Gerador de Topic Maps a partir de um conjunto de documentos de entrada.
TM-Designer	Editor gráfico de ISO Topic Maps.
TM4J	Processador de Topic Maps.

Table 1: Algumas ferramentas baseadas em Topic Maps.

Segundo, topic maps é uma excelente tecnologia e um padrão que une vários níveis de profissionais, de técnicos de marketing até arquitectos de informação. Terceiro, uma grande parte do trabalho dos autores deste artigo é a criação de sites e protótipos de vários projectos e estilos, então, para facilitar o processo, foi construído o *DINavigator* que apresenta o conhecimento extraído das relações através de menus contextuais. Concluindo, topic maps mostra ser a perfeita tecnologia para esta tarefa.

Além do *DINavigator*, uma gama de ferramentas para o processamento de topic maps está disponível. Contudo, nenhuma delas é uma distribuição simples de instalar e usar, pois requerem outra tecnologia, como outro software, biblioteca ou linguagem de programação. É o caso do *Tomcat* que é necessário para o uso do *Omnigator*, assim como o *TM4J* se faz necessário para o uso do *Nexist*, ou a instalação do *Python* para o uso do *SemanText*, assim como um plug-in no *Protégé 2000* para a edição de topic maps. Assim, *DINavigator* provém uma simples solução para o processamento de topic maps, pois apenas tem a dependência de um simples parser XSLT.

O documento XTM é onde todos os metadados sobre a ontologia estão armazenados, e o *DINavigator* usa isto para a estruturação, relações e tudo mais o que for necessário. O documento XTM gerado pelo TM-Builder não é indicado para a edição manual, pois na próxima geração do XTM, todas as alterações feitas manualmente serão sobrescritas pelo processo.

4. Arquitectura do Metamorphosis

A motivação para o desenvolvimento do *Metamorphosis* veio de duas situações que surgiram no contexto de alguns projectos de desenvolvimento de software:

- Muitas vezes, para se testar algumas funcionalidades de um sistema que se está a desenvolver é necessário criar uma interface Web para realizar esses testes. Normalmente, estas interfaces não têm grandes requisitos quanto à aparência, o que se pretende é que sejam desenvolvidas o mais rapidamente possível.
- Quer-se expôr na Web um sistema de informação. Este sistema é composto por bases de dados, documentos XML ou documentos PDF. Quer-se que todos os itens de informação estejam acessíveis via Web e para isso constroem-se os primeiros índices. Estes índices acabam por ser enormes excedendo a capacidade dos browsers da Internet. Podia pensar-se em fraccioná-los alfabeticamente, mas há situações em que isso não é possível nem recomendável. Mas, é sempre possível arranjar um método para fraccionar a informação conceptualmente. É aqui que se começa a discutir a organização dos recursos de informação e é aqui que se introduz o *Metamorphosis*. Não é necessário alterar nenhum dos recursos de informação, apenas é necessário criar uma ontologia para esses recursos que reflita a visão que se quer para eles.

A figura 1 vem reforçar esta ideia e dá uma visão pictórica da arquitectura do *Metamorphosis*.

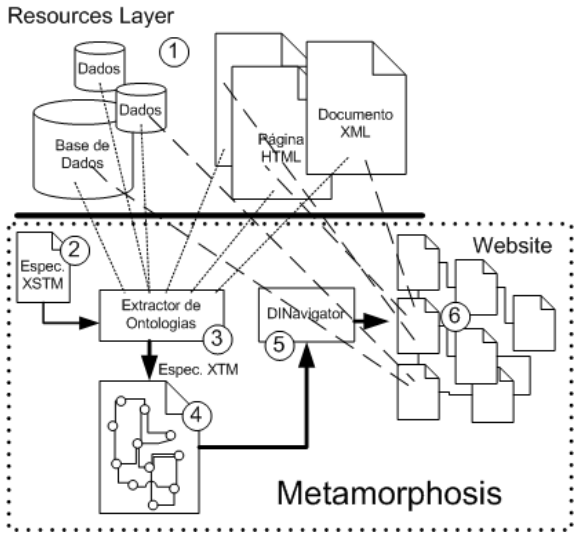

Figure 1: Metamorphosis

Esta arquitectura pode ser descrita da seguinte maneira:

(1) camada de recursos 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) especificação XSTM É um documento XML que fornece informações precisas sobre onde aceder para extrair as partes da informação necessárias para a construção da ontologia. Esta especificação descreve todos os tópicos a serem extraídos dos recursos de informação e as associações entre estes tópicos; este componente é descrito em detalhe em [LRH03b].

(3) Ontology Builder/Extractor Este componente utiliza a especificação XSTM para ir aos recursos de informação buscar a informação de que necessita para construir a ontologia.

Está implementado como uma folha de estilo XSL e vai tirar partido da organização da informação: por exemplo, quando se está a trabalhar com uma base de dados relacional, procura a informação relativa às relações e vai gerar automaticamente todas as associações entre os tópicos relevantes.

(4) especificação XTM Esta é a especificação do topic map gerada automaticamente pelo Ontology Builder/Extractor.

(5) DINavigator Este é o componente que está a ser discutido com detalhe neste artigo. Também implementado em XSL, toma como entrada um *Topic Map* e produz um website inteiro de acordo com algumas regras.

(6) Website Este é website final através do qual é possível navegar por entre os vários recursos de informação que compõem o sistema de informação original.

Na próxima secção, descreve-se em detalhe o *DINavigator* e a geração do website.

5. DINavigator - o componente de navegação

Até agora, as ontologias especificadas em XTM são um conjunto de registos, onde cada registo representa um conceito, que identifica recursos (informação física) e participa de relações (associações). Então, primeiramente, usa-se essas relações entre os conceitos para navegar na informação e, posteriormente, quando encontrado o recurso de informação desejado, recuperar essa informação.

A ideia principal da navegação conceptual pode ser descrita como: quando se está posicionado sobre um certo conceito, a ferramenta de navegação mostrará as informações associadas a este conceito em particular; se for escolhido algum dos outros conceitos relacionados, a posição muda para o conceito e a visão muda de acordo com a escolha; se for escolhido algum dos recursos de informação, o sistema mostrará o próprio recurso.

Em termos de implementação, foi estabelecido que a navegação se faria através de web browsers. Então, todas as visões dos conceitos seriam apresentadas em páginas HTML. Assim, a implementação desse componente de navegação é reduzida a transformações XML. Contudo, há duas possibilidades: transformação a pedido em tempo de execução (website dinâmico) e transformação em batch (website estático).

Este navegador é definido como *DINavigator*, e é apresentado na figura 2, onde é encontrada a visualização do topic map referente ao *DI*, (criado pelo TM-Builder) no *DINavigator*. Este navegador fornece o total acesso à ontologia extraída, permitindo a navegação através dos conceitos definidos na especificação em XSTM.

A visualização de um documento XTM no *DINavigator* apresenta no seu lado esquerdo, a árvore obtida a partir da ontologia, na forma de um menu hierárquico. No corpo do navegador, pode-se navegar a partir da própria ontologia, por um índice alfabético de todos os tópicos ou pela sua hierarquia de classes, visualizada graficamente.

5.1. Transformações em tempo de execução

As transformações em tempo de execução foram implementadas com transformações XML com auxílio de uma CGI. A página inicial é uma chamada a uma CGI parametrizada com o principal conceito da ontologia (que vai servir de ponto de entrada para a navegação). A CGI aplica a transformação na ontologia e produz uma visão HTML. Nessa visão, todos os links gerados são chamadas à mesma CGI mas com diferentes parâmetros, para novas posições na ontologia.

Esta é a melhor solução porque somente necessita de dois ficheiros: a ontologia (especificada em XTM) e a CGI. Contudo, as transformações em tempo de execução consomem muito tempo. Por isto, esta solução tem sido utilizada apenas em pequenos protótipos.

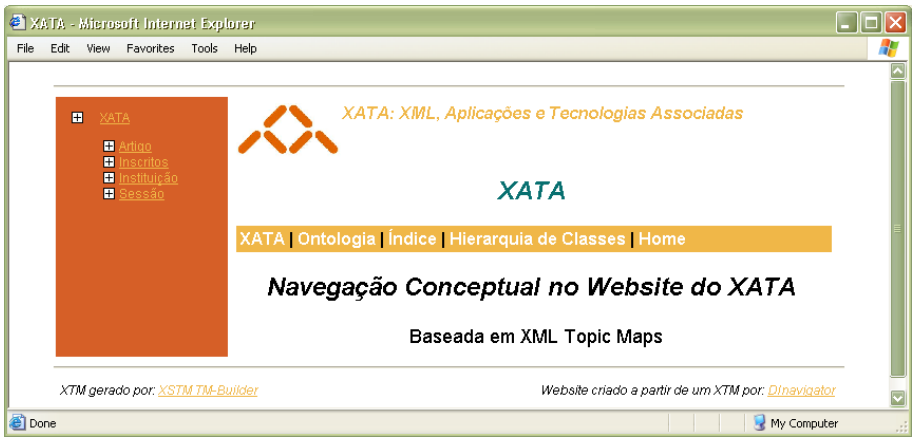

Figure 2: Visualização do Topic Map do XATA no *DINavigator*

5.2. Transformações em batch

Essa solução corresponde ao atual componente que está sendo usado no *Metamorphosis*. Foi criada uma stylesheet XSL que transforma a ontologia num website. Esse website corresponde a todas as possíveis visões da ontologia.

Assim, o *DINavigator* é visto como um gerador de websites, baseado em XSL e usando topic maps para esse propósito. Esta é uma maneira simples de criar de websites inteiros, incluindo o design, o conteúdo e os links aos tópicos. Em outras palavras, o *DINavigator* é simples e poderoso.

O *DINavigator* foi criado como uma consequência directa da criação de um *framework* em XSLT para a linguagem XSTM [LRH03b]. Quando essa linguagem foi desenvolvida, houve a necessidade de criar uma ferramenta (ou protótipo) para esse trabalho, para validar os documentos XTM gerados a partir do TM-Builder. O *DINavigator* foi criado por três razões:

- Para ser um protótipo eficiente neste trabalho, e diminuindo o tempo de criação de um navegador para um XTM;
- Para criar uma ferramenta que usa topic maps, e que é facilmente distribuída, instalada e usada por todos;
- Para unir as linguagens técnicas e métodos de projetistas, programadores e arquitetos de informação.

O *DINavigator* oferece um framework para permitir rápidas modificações em todo o website. O website gerado pelo *DINavigator* é transformado através de parsers XSLT, e o resultado é cuidadosamente verificado até à produção completa das páginas HTML. Isto simplesmente significa que é possível a rápida criação de protótipos e a expansão usando essa mesma ferramenta para se obter um ambiente de produção. Isso diminui radicalmente o custo e o tempo de desenvolvimento.

O Topic map é usado no *DINavigator* para todas as estruturas, relações, preparação de conteúdo, recursos e ocorrências, ou seja, o topic map expressa a ontologia. Isso significa que qualquer outro topic map, independente de ter sido obtido através do *TM-Builder*, pode ser usado no *DINavigator* para a obtenção de um completo mapa de metadados.

6. Caso de estudo: XATA 2003

A fim de demonstrar o uso do TM-Builder para a extração de ontologias a partir de uma fonte XML, esta seção apresenta o caso de estudo da workshop *XML, Aplicações e Tecnologias As-*

sociadas (*XATA*[3]), que ocorreu na Universidade do Minho, em Braga - Portugal, nos dias 13 e 14 de Fevereiro de 2003, visando reunir a comunidade XML de língua portuguesa. Participaram pesquisadores e utilizadores de XML, tanto oriundos de universidades quanto de empresas, permitindo assim um compartilhamento de informação entre o mundo acadêmico e o mundo profissional.

Nesta workshop, vários artigos foram submetidos para avaliação; e os aprovados foram devidamente apresentados durante a conferência. As apresentações dos artigos foram divididas em sessões, cada qual com um tema associado, como Tecnologia e Web Services, XML e Base de Dados, entre outras. Após cada exposição, uma mesa redonda era formada para debater o assunto do artigo, consoante as dúvidas que o mesmo havia criado.

Como não poderia deixar de ser, o evento foi todo baseado em XML, desde sua divulgação quanto sua produção. Portanto, todas as informações referentes ao *XATA* estão armazenadas em documentos XML. A parte interessante, a este caso de estudo, do XML-Schema do documento que contém as informações essenciais sobre o evento é apresentado na figura 3. Obviamente este XML-Schema é mais completo, porém, para nosso caso de estudo, o importante está ressaltado nessa figura.

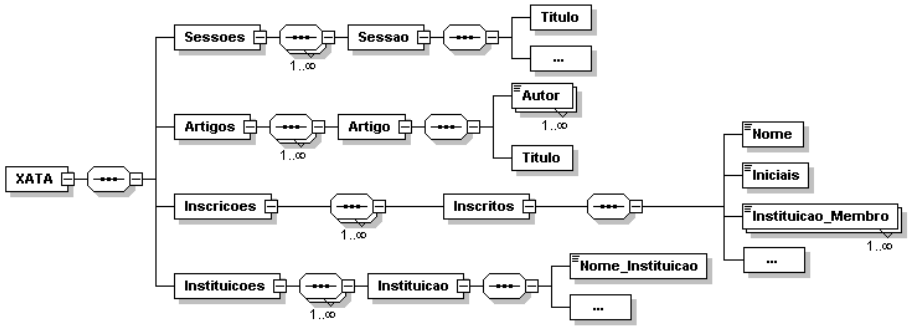

Figure 3: XML-Schema do XATA.

Como a linguagem XSTM depende apenas da estrutura do documento, e não da instância, a partir deste XML-Schema é possível definir a especificação da ontologia do XATA. Portanto, são cinco etapas que devem ser seguidas: definição dos tipos de tópicos, dos próprios tópicos, dos papeis de atuação em ocorrências, dos tipos de associação e, por fim, das próprias associações. This section presents a case study about the workshop *XML, Aplicações e Tecnologias Associadas* (*XATA*).

6.1. A especificação XSTM para o XATA

O elemento raiz do XML-Schema de XSTM é *xstm*, o qual possui cinco sub-elementos, cada um referente a uma parte da ontologia expressada por XTM. Os seus sub-elementos são: *topicType*, *topic*, *occurrenceRole*, *assocType* e *assoc*.

Inicialmente, são definidos os tipos de tópicos. Nesta ontologia, os tópicos são agrupados em Instituição, Autor, Assistente, Sessão e Artigo. Em XSTM, os tipos de tópicos são declarados pelo elemento *topicType*, contendo um identificador (*id*) – para ser referenciado em outros momentos na especificação – e um nome (*name*) – para a visualização do XTM em um navegador. Como exemplo, é mostrado abaixo a declaração do tipo de tópico *Artigo*.

```
<xstm>
    <topicTypes>
```

[3] `http://www.di.uminho.pt/~jcr/XML/conferencias/xata2003/`

```
        <topicType>
            <id>ID-Artigo</id>
            <instanceOf>XATA</instanceOf>
            <name>Artigo</name>
        </topicType>
        <topicType>...</topicType>
    </topicTypes>
    ...
</xstm>
```

Enquanto que os tipos de tópicos são conceitos abstratos definidos pela ontologia, tópicos são elementos reais nos documentos XML tomados como entrada. Para definí-los, é usado o elemento *topic* que possui dois sub-elementos: o caminho XPath referente ao próprio elemento (*xpath*) e o seu tipo (*type*). Abaixo, é encontrado a declaração XSTM para a definição dos tópicos referentes ao tipo de tópico *Artigo*.

```
    <topics>
        <topic>
            <xpath id="@numero" name="Titulo">//Artigo</xpath>
            <type>Artigo</type>
        </topic>
        <topic>...</topic>
        ...
    <topics>
```

Até o momento, todos os tópicos, e seus respectivos tipos, estão declarados. Mas em XTM, tópicos sem qualquer associação relacionada aos mesmos não possuem funcionalidade. Então, é necessário definir as associações entre os tópicos. Várias associações podem ser inferidas a partir do *XATA*, por isso, vamos tomar como exemplo a associação entre os tipos de tópicos *Artigo* e *Autor*.

Uma vez definidos os tópicos e seus tipos, o próximo passo é a definição dos tipos de associação, que também é um conceito abstrato na ontologia. Ele define o papel de atuação de cada um dos membros das associações. É declarado com o elemento *assocType* que possui um identificador (*id*), um nome (*name*) e os membros deste tipo de associação (*memberAssoc*). Cada membro é definido por um contexto *scope* – o identificador do papel de atuação – e a sua respectiva descrição (*description*). Sendo assim, é visualizado abaixo a especificação do tipo de associação entre *Autor* e *Artigo*.

```
    <assocTypes>
        <assocType>
            <id>ID-autor_artigo</id>
            <name>Autor e Artigo</name>
            <memberAssoc>
                <scope>ID-escrito_por</scope>
                <description>é escrito por</description>
            </memberAssoc>
            <memberAssoc>
                <scope>ID-autor_de</scope>
                <description>é autor</description>
            </memberAssoc>
        </assocType>
        <assocType>...</assocType>
    </assocTypes>
```

Para finalizar a especificação em XSTM, o elemento *assoc* permite a especificação de todas as associações que envolvem dois ou mais tópicos; elas são encontradas e extraídas a partir do documento XML fonte.

Neste âmbito, quando refere-se a relacionamentos entre nodos da árvore XML (elementos e atributos), não está se referindo ao modelo entidades-relacionamento. Portanto, os nomes 1-para-1, 1-para-N e M-para-N não tem exatamente o mesmo significado usado na perspectiva tradicional.

Neste contexto, há quatro tipos de relacionamentos entre elementos (ou atributos) que são descritos por três elementos filhos de *assoc*:

- o elemento *one2one* descreve as associações entre elementos e seus atributos, com seu atributo *type* com o valor *attribute*;
- o elemento *one2one* também descreve as associações entre elementos distintos, com seu atributo *type* com o valor *subelement*;
- o elemento *one2N* define as associações *um para muitos*;
- o elemento *M2N* define as associações *muitos para muitos*;
- o elemento *all2all* define as associações que são definidas por uma tabela intermediária.

A estrutura dos sub-elementos de *assoc* são muito similares. Cada um dos três sub-elementos acima descritos possui o seu tipo (*type*) – o identificador do tipo de associação correspondente – e os membros que fazem parte desta associação (*members*). Os membros possuem dois elementos filhos: *topicAssoc* que identifica o tipo de tópico pertencente a esta associação e *role*, que demonstra o papel de atuação do tópico na atual associação.

O elemento *one2one* expressa relacionamentos que podem ser obtidos a partir de algum elo de ligação entre os tópicos encontrados no documento XML. Como por exemplo, no caso específico da associação entre *Autor* e *Artigo*, os autores de cada artigos podem ser identificados devido ao conteúdo do caminho XPath *//Artigo/Autor*, o qual é uma referência às iniciais dos autores encontradas em *//Inscritos/Iniciais*. Assim, a associação entre os tipos de tópicos *Autor* e *Artigo*, referente ao *XATA*, foi especificada da maneira abaixo demonstrada:

```
    <assocs>
        <one2one type="subelement">
            <type>autor_artigo</type>
            <members11>
                <element>
                    <topicAssoc ref="Autor">Artigo</topicAssoc>
                    <role>eh_escrito_por</role>
                </element>
                <elementRef>
                    <topicAssoc ref="Iniciais">Inscritos</topicAssoc>
                    <role>eh_autor</role>
                </elementRef>
            </members11>
        </one2one>
    </assocs>
    ...
</xstm>
```

7. Conclusão e Trabalho Futuro

O *Metamorphosis* tem sido utilizado em vários projectos de pequena e média dimensão.

Até ao momento, provou ser uma boa ferramenta de prototipagem. As interfaces Web são criadas rapidamente e sem grandes dificuldades por parte dos utilizadores.

Até agora pudemos identificar as seguintes áreas de intervenção:

Páginas pessoais - Com o *Metamorphosis* tem sido possível criar rapidamente websites pessoais com uma aparência normalizada mesmo quando os recursos de informação têm uma estrutura diferente.

Conferências e Workshops - Como o caso de estudo indica, o sistema tem sido utilizado para suportar os websistes de algumas conferências, como o XATA (XML: Aplicações e Tecnologias Associadas[4]).

[4]http://www.di.uminho.pt/ jcr/XML/conferencias/xata2003/

E-Learning - Neste momento, a equipe do *Metamorphosis* está também a desenvolver uma plataforma para a produção de conteúdos baseada na tecnologia XML. Desta iniciativa, surgiram pequenos projectos que deram origem a várias aplicações XML: uma para a produção de guiões para aulas, outra para a produção de apresentações e outra para a produção de testes e exames [LRH03a]. Para integrar todas aquelas aplicações está-se a utilizar o *Metamorphosis* para a criação da interface Web.

Arquivos - Estão em curso vários projectos, em dois arquivos históricos nacionais, que visam a disponibilização do acervo documental daqueles arquivos na Web. O sistema de informação por detrás de um arquivo histórico é gigantesco e o *Metamorphosis* está a ser utilizado para prototipar a interface Web.

No entanto, o *Metamorphosis* ainda não está acabado:

- Está a ser testado com sistemas de informação de médio e grande porte;
- Está a ser adaptado para trabalhar com qualquer recurso de informação;
- Está a estudar-se um modelo relacional para os *Topic Maps* (se a ontologia fôr muito grande um documento XML não será a maneira ideal de a armazenar);
- Está a ser estudado um componente que irá permitir ao utilizador especificar o aspecto visual do website gerado (por enquanto, apenas há uma configuração da interface).

Neste momento, está a ser criada a página Web do projecto e será divulgada brevemente (também esta será criada com o *Metamorphosis*).

References

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

T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. In *Scientific American*. http://www.sciam.com/2001/0501issue/0501berners-lee.html, May 2001.

Michel Biezunski and Steven R. Newcomb. Topic Maps Frequently Asked Questions, September, 1999. www.infoloom.com.

Sean Bechhofer, Frank van Harmelen, Jim Hendler, Ian Horrocks, Deborah L. McGuinness, Peter F. Patel-Schneider, and Lynn Andrea Stein. Web Ontology Language (OWL) Reference Version 1.0. World Wide Web Consortium, November, 2002. http://www.w3.org/TR/owl-ref/.

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

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

Charles F. Goldfarb and Paul Prescod. *XML Handbook.* Prentice Hall, 4th edition, 2001.

Nicola Guarino. 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. Springer Verlag, http://www.ladseb.pd.cnr.it/infor/Ontology/Papers/SCIE97.pdf, 1997.

Giovani Librelotto, José C. Ramalho, and Pedro R. Henriques. ADRIAN – a platform for e-learning content production. In *Second International Conference on Multimedia and Information & Communication Technologies in Education (m-ICTE 2003)*, 2003.

Giovani Librelotto, José C. Ramalho, and Pedro R. Henriques. TMBuilder: Um Construtor de Ontolotias baseado em Topic Maps. In *XXIX Conferencia Latinoamericana de Informática – CLEI*, 2003.

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

OIL. Welcome to OIL, 2002. http://www.ontoknowledge.org/oil.

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

Jack Park, Sam Hunting, and Douglas C. Engelbart. *XML Topic Maps: Creating and Using Topic Maps for the Web*. Prentice Hall, 2003.

Steve Pepper and Graham Moore. XML Topic Maps (XTM) 1.0. TopicMaps.Org Specification, August, 2001. http://www.topicmaps.org/xtm/1.0/.

José Carlos Ramalho and Pedro Henriques. *XML & XSL Da Teoria à Prática*. FCA Editora, 2002.

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

W. Swatout and A. Tate. Ontologies. In *IEEE Inteligent Systems and their applications*, volume vl 14, n 1. IEEE, Jan/Fev 1999.