ADRIAN: Sistema de Suporte à Produção de Conteúdos

Ramalho J. C. [1], Henriques P. R. [1], Librelotto G. R. [1]

[1]Centro de Ciências e Tecnologias da Computação (CCTC)
Universidade do Minho, Braga, Portugal

Resumo. Neste artigo, descreve-se um protótipo dum sistema para apoio à produção de conteúdos para E-Learning, baptizado com o nome de ADRIAN. Efectivamente, se analisarmos com cuidado os produtos mais utilizados para implementação de plataformas E-Learning constatámos que fazem praticamente tudo desde a gestão do processo do aluno à análise da frequência e assiduidade deste. No entanto, no que diz respeito a conteúdos, permitem ao docente publicar aquilo que este bem entender não dando qualquer suporte à produção do conteúdo propriamente dito. Foi para preencher este vazio que o ADRIAN foi pensado.

1 Introdução

Hoje em dia, a Web é um dos veículos mais atraentes para a distribuição e acesso à informação. A sua aplicação à área do ensino à distância era uma questão de tempo. Hoje é a realidade.

Em 1994, realizámos as primeiras experiências de utilização da Web para o ensino à distância. Naquela altura, utilizávamos mailing-lists para interagir com os alunos e páginas Web para disponibilizar conteúdos.

Actualmente, a procura de cursos em formato E-Learning cresce de dia para dia. A possibilidade de poder seguir um curso a partir do conforto de casa ou simplesmente eliminando distâncias e barreiras geográficas é um conceito muito atractivo. Do lado institucional, os cursos em formato de E-Learning têm também algumas vantagens como a economia de espaço físico e temporal (estes cursos estão disponíveis 24 horas). No entanto, apesar da economia física, o E-Learning exige mais recursos: a preparação de conteúdos e a gestão de alunos consome tempo e recursos humanos.

É aqui que surgem os sistemas de gestão de ensino ("LMS – Learning Management Systems"). Já existe um número considerável destes sistemas em utilização um pouco por todo o mundo. Alguns deles, devido à qualidade de serviços que oferecem são considerados referências: Blackboard, Luvit, WebCT, Lotus Learning Space, ... Alguns resultaram do desenvolvimento de protótipos académicos, outros são desenvolvimentos da insdústria de software. Um sistema destes fornece funcionalidades para gerir os processos dos alunos, para facilitar e gerir a comunicação entre alunos, e entre estes e o professor, para controlar os acessos ao sistema e produzir estatísticas, para gerir horários, para efectuar a avaliação dos alunos e, uma plataforma aberta para a publicação de conteúdos.

No entanto, este sistemas não prevêm qualquer tipo de regra quanto à tecnologia ou ao formato a utilizar na preparação dos conteúdos. Apesar disto poder ser visto como uma vantagem, nalguns contextos pode levar a uma proliferação anárquica de formatos e à impossibilidade de criação de um suporte à criação de conteúdos.

O sistema ADRIAN (sistema de suporte à criação e gestão de conteúdos) está a ser desenvolvido num contexto muito especial: os autores são professores de informática, alguns fazem parte de uma comissão que visa implementar o E-Learning no Campus e são responsáveis por várias disciplinas que estão a migrar para esta forma de ensino.

De momento, o sistema corresponde à integração de cinco componentes que foram desenvolvidos separadamente: três aplicações de produção de conteúdos (uma para exames, uma para aulas práticas/laboratoriais e uma para apresentações multimédia), um componente de integração de conteúdos (este componente corresponde a um navegador semântico que utiliza ontologias para guiar o utilizador), e um componente de gestão de actualizações (utiliza uma especificação de dependências para que sempre que um novo objecto de aprendizagem seja adicionado ao sistema este é alterado reflectindo o novo acrescento: actualização da ontologia ...).

Na apresentação, estes componentes serão apresentados em detalhe com particular incidência na integração e na gestão de actualizações.

Relativamento aos objectos de aprendizagem, pode-se considerar que são constituídos por duas partes: meta-informação e o conteúdo propriamente dito. Já existem várias normas para a meta-informação de um objecto de aprendizagem como o LOM (Learning Object Metadata) e o IMS/EML. Neste projecto, inclui-se alguma meta-informação nos objectos de aprendizagem (um subconjunto pequeno) pois o projecto centrou-se no conteúdo e quanto a este não há normas nem propostas. É nossa convicção de que estamos na altura de poder avançar neste ponto e é aqui que pretendemos contribuir.

2 Arquitectura do sistema

A arquitectura functional do ADRIAN é mostrada na figura seguinte.

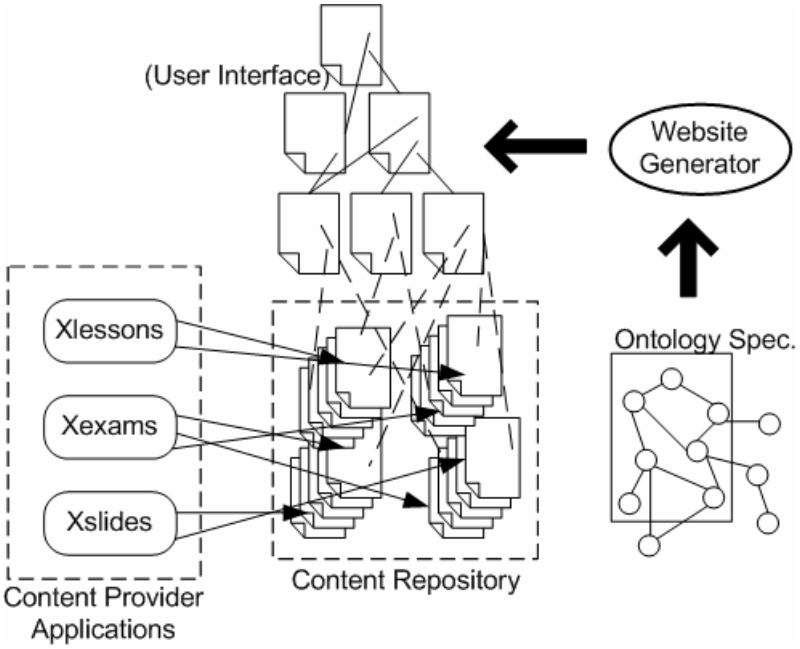

Fig. 1. Composição do ADRIAN

Na figura 1, podemos distinguir quatro grandes componentes:

- **Aplicacional (Aplicações para a produção de conteúdos)** - De momento este componente compreende três aplicações: *Xlessons* para a produção de aulas (aula teórica, guiões laboratoriais, folhas de exercícios, ...; *Xexams* para a produção de exames de vários tipos (múltipla escolha, verdadeiro e falso, progressivos, e de desenvolvimento) e *Xslides* para assistir o utilizador na produção de diapositivos.
- **Armazenamento** - Este componente é materializado no armazenamento físico dos conteúdos produzidos. Normalmente, é uma subárvore do sistema de ficheiros na máquina servidora da aplicação. Esta estrutura deverá estar pensada e criada antes de se começarem a produzir conteúdos. Está a ser desenvolvido um novo componente para gerir este repositório e que irá permitir alterar a sua estrutura dinamicamente. A estrutura que se está a usar para o repositório encontra-se representada na figura 2.
- **Ontológico (organização dos conteúdos)** – Este componente corresponde a uma organização abstracta e virtual dos conteúdos produzidos. Esta organização abstracta é dada através da especificação de uma ontologia. Neste momento, já existe uma pequena aplicação que trabalha sobre o repositório e calcula uma ontologia automaticamente baseando-se na hierarquia física do repositório. Está em desenvolvimento uma nova aplicação que permitirá ao utilizador acrescentar conceitos e relações à ontologia calculada automaticamente.

- **Interface (procura, navegação no repositório e visualização de conteúdos)** – Este componente é um conjunto de páginas Web geradas automaticamente a partir da ontologia e que vai permitir ter um acesso (guiado pela semântica expressa na ontologia) aos conteúdos.

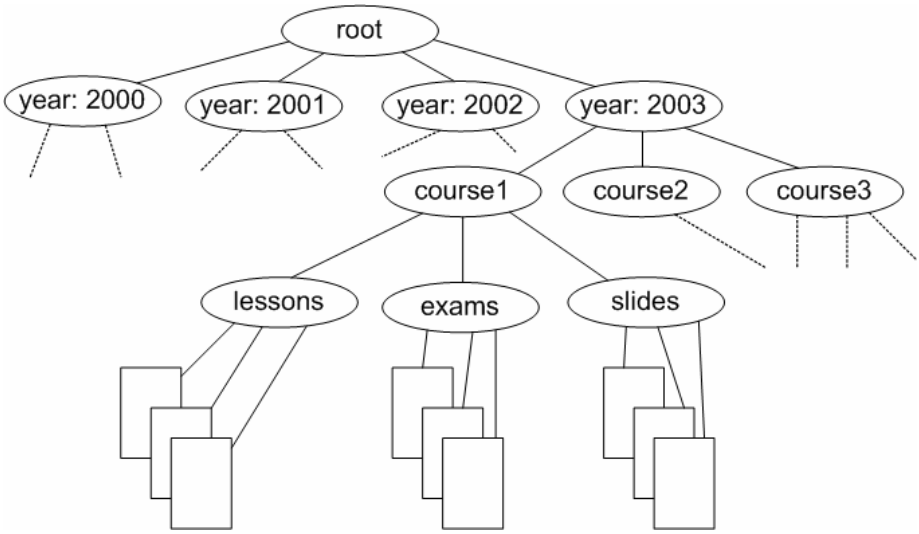

Fig. 2. Estrutura Física do Repositório de Conteúdos

Na secção seguinte descrevem-se com mais detalhe cada um destes componentes.

3 Produção de Conteúdos

Todas as aplicações para a produção de conteúdos partilham a mesma filosofia: baseiam-se nos princípios da documentação estruturada e utilizam anotação descritiva para estruturar os documentos que são produzidos. Esta metodologia tem grandes implicações. A primeira é a de que o formato não é livre, tem uma representação textual, a sua estrutura é completamente descrita através de anotações e está formalmente especificada numa gramática formal.

As vantagens na utilização da anotação descritiva estão bem documentadas na bibliografia [1], [2], [3] mas podemos realçar as mais importantes: portabilidade (uma vez que a anotação descritiva não tem nenhum siginficado operacional qualquer documento anotado descritivamente pode ser trocado entre plataformas e sistemas heterogéneos), reutilização, reconfiguração do aspecto visual (a anotação descritiva imlica uma completa separação entre conetúdo e forma o que torna possível a associação de várias e diferentes especificações de forma ao mesmo conteúdo) e longevidade (a informação está num formato neutro e facilmente processável).

Na tentativa de clarificar ainda mais esta escolha tecnológica, as próximas secções detalham alguns dos conceitos inerentes a este paradigma.

3.1 Anotação Descritiva

A ideia da utilização da anotação descritiva na publicação electrónica já vem dos anos 60. No entanto, só em 1986 surgiu o SGML [4], [5], como uma norma ISO (ISO 8879) para a publicação electrónica de conteúdos (posteriormente surgiu em 1998 o XML [6]). O SGML e o XML são metalinguagens com as quais se podem especificar linguagens de anotação descritiva de domínio específico. Apesar das vantagens referidas o SGML nunca foi verdadeiramente aceite devido à grande complexidade no seu processamento. Por outro lado, o XML sendo uma versão mais simples e fácil de processar impôs-se em praticamente todas as áreas da computação.

Um documento XML tem uma estrutura lógica à qual corresponde uma hierarquia de elementos. Cada elemento é diferenciado dos restantes através de anotações que são adicionadas ao documento. Nesta perspectiva, um documento é composto por dois tipos de informação: dados e anotações. O próximo exemplo mostra parte de um documento correspondente a um guião laboratorial:

Exemplo 1: Documento XML para um guião laboratorial

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<AulaPrática>
  <meta>
    <disciplina>Processamento Estruturado de  Documentos
    </disciplina>
    <data>2004.06.04</data>
    <objectivos>
       <para>O objectivo principal desta ficha ...</para>
       <para>Para atingir esse fim, ...</para>
    </objectivos>
    <recursos>
      <documento url="http://www.di.uminho.pt/~jcr/XML
                      /applications/album/">
          Ficheiros de trabalho para o álbum fotográfico
      </documento>
    </recursos>
  </meta>
  <corpo>
    <exercício>
      <título>Álbum Fotográfico</título>
      <enunciado>
        <para>Descarregue os ficheiros ...</para>
      </enunciado>
    </exercício>
  </corpo>
</AulaPrática>
```

Este exemplo mostra um documento XML composto por dois componentes. Um documento XML tem três componentes possíveis:

- Declaração XML – todos os documentos têm de ser iniciados por esta declaração:
  ```
  <?xml version="1.0" encoding="ISO-8859-1"?>
  ```
- DTD (Document Type Definition) ou Schema – Uma especificação gramatical que especifica que anotações são obrigatórias, quais são opcionais, onde é que são obrigatórias e onde é que podem ser opcionais. Esta gramática define a linguagem de anotação, ou seja, o dialecto XML específico.
- Documento – O documento propriamente dito, composto por texto, anotações e, opcionalmente, uma referência a um DTD ou Schema.

De um documento XML que esteja de acordo com as regras expressas num DTD ou Schema diz-se que é válido de acordo com esse DTD ou Schema. Um documento XML que não esteja de acordo com nenhum DTD ou Schema é considerado um documento bem-formado.

A figura 3 representa o ciclo de vida de um documento XML. De certa forma, esta figura ilustra a metodologia seguida para desenvolver as três aplicações produtoras de conteúdos que se estão a discutir nesta secção.

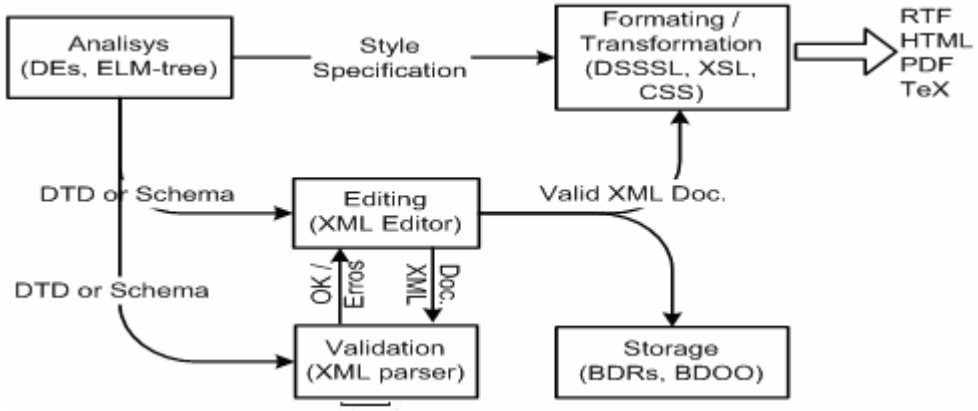

Fig. 3. Ciclo de vida dos documentos XML

A figura 3 apresenta o ciclo de vida dos documentos estrutrados composto por 5 fases: análise, edição, validação, armazenamento e formatação. A fase de análise corresponde ao estudo de um determinado tipo de documento. O resultado desta fase deverá ser um DTD ou um Schema que define completamente a estrutura de uma determinada família de documentos [2], [7]. Depois desta fase, há um pequeno ciclo entre outras duas fases: o utilizador cria/edita um documento e pede ao editor que teste a sua validade de acordo com um determinado DTD; se houver erros no processo de validação o utilizador irá corrigi-los editando de novo o documento, este processo é iterado até o documento passar no processo de validação. Esta fase (edição+validação) produz documentos XML válidos. Um documento XML depois de validado pode ser armazenado ou transformado/formatado para se obter um outro formato para um fim específico. Há várias soluções para o

armazenamento [8] que não serão discutidas aqui. O processo de transformação é normalmente especificado numa folha de estilo (*stylesheet XSL – XSL é uma sintaxe XML para especificar de uma forma declarativa transformações de documentos estruturados*) [9].

A seguir descreve-se de que forma estes conceitos foram aplicados nas aplicações produtoras de conteúdos.

3.2 Exames

Este é, talvez, o tipo de documento mais complexo no ambiente ADRIAN. Normalmente, no contexto do E-Learning referimo-nos a exames para disponibilizar na Web. Estes exames podem ser de dois tipos: estáticos (o exame é apenas mostrado numa página Web normal, situação partilhada pela maior parte dos outros documentos que se disponibilizam) e dinâmicos (a página Web através da qual o exame é disponibilizado está preparada para processar a interacção com o aluno quando este o tenta resolver e, normalmente, dá feed-back imediato ao aluno que o está a tentar resolver) . A aplicação para a produção de exames, *Xexams*, foi desenvolvida a pensar nos exames dinâmicos.

Os exames dinâmicos podem ser divididos nos seguintes tipos:

- Escolha múltipla – uma questão tem um conjunto de respostas; o aluno tem que indicar as que estão correctas.
- Verdadeiro e Falso – uma questão tem um conjunto de respostas; o aluno tem que indicar, para cada resposta, se esta é verdadeira ou falsa.
- Desenvolvimento – para cada questão, o aluno tem à sua disposição uma área em branco onde deverá desenvolver a sua resposta à questão.

A aplicação Xexams foi desenvolvida para suportar estes três tipos de exame.

Em termos funcionais podemos ainda subdividir os exames em três tipos:

- Tentativa única – o aluno tem apenas uma tentativa para resolver o exame (o correspondente à abordagem tradicional).
- Limite temporal – o aluno tem de resolver o exame dentro de um intervalo temporal.
- Aleatório – o sistema troca constantemente a ordem das questões; para o mesmo exame a ordem pela qual as questões são mostradas é sempre diferente.

Um exame não precisa de pertencer rigorosamente a um destes tipos, pode ser um misto dos vários tipos. Para se conseguir isto, cada questão é tratada individualmente tornando-se na unidade base para a composição de exames.

Para se dar uma ideia dum exame definido na aplicação Xexams mostra-se um pequeno excerto:

Exemplo 2: Excerto de um exame

```
...
  <body>
    <questions>
      <question numberQ="Q1">
        <description>
          <para>No Tratado de Roma assinado em 1957 pelos
  seis Estados-membros de então, ficava expresa a intenção em
  realizar:</para>
         </description>
         <choices>
           <choice answer="false">
             <para>uma união aduaneira;</para>
           </choice>
           <choice answer="false">
             <para>uma união aduaneira, um mercado comum e
  uma união económica;</para>
           </choice>
           <choice answer="true">
             <para>uma união aduaneira e um mercado
  interno;</para>
           </choice>
           <choice answer="false">
             <para>um mercado comum, uma união económica e
  uma união económica e monetária.</para>
            </choice>
          </choices>
        </question>
        ...
      </questions>
    </corpo>
  </exame>
```

Neste exemplo, define-se uma questão do tipo “Verdadeiro/Falso”. Ao criar a questão o docente coloca desde logo a resposta permitindo mais tarde ao sistema realizar uma correcção automática. De momento, a aplicação *Xexams*, após receber um exame definido desta maneira gera automaticamente quatro resultados: uma versão Web do exame, uma versão PDF para que quer alunos quer docentes o possam imprimir e, posteriormente à interacção com um aluno o sistema gera uma versão Web com as respostas do aluno corrigidas e outra versão PDF com o mesmo conteúdo. O processo de avaliação automática não será discutido aqui.

Nas secções seguintes, referem-se brevemente as duas outras aplicações.

3.2 Apresentações baseadas em diapositivos

Esta aplicação, *Xslides*, segue a mesma filosofia descrita na secção anterior. Foi desenvolvido um *XML Schema* para esta classe de documentos e foram especificadas duas

transformações: uma para gerar um conjunto de páginas Web que reproduz a apresentação desejada e outra para gerar uma versão PDF dos diapositivos.

3.3 Aulas e guiões laboratoriais

Esta aplicação foi usada no ano lectivo transacto, para suportar aulas práticas e sessões laboratoriais. Nestes cursos, os circuitos de papel utilizados anteriormente foram substituídos por conjuntos de páginas Web acessíveis de qualquer ponto com acesso à Internet (que pode levantar outro tipo de questões relacionadas com os direitos de autor).

No desenvolvimento da aplicação, foi seguida a mesma metodologia, tendo sido apenas desenvolvida uma transformação para gerar as páginas Web.

4. Armazenamento e gestão de conteúdos

Todos os documentos que são produzidos têm que ficar armazenados algures. Além disso, para cada documento XML que é criado há toda uma cadeia de transformações que tem de ser executada de modo a obter as versões Web e PDF. A solução ideal seria ter um gestor de conteúdos que permitisse automatizar todo este processo. Este projecto está a dar os primeiros passos e haverá novas evoluções em breve.

De momento, os documentos têm de ser armazenados numa subárvore de directorias no sistema de ficheiros da máquina servidora. O utilizador tem de conhecer a estrutura física desta subárvore e, por vezes, tem de fazer algumas operações de manutenção “à mão”.

5. Ontologia

Noutro projecto, que tem vindo a ser desenvolvido pela mesma equipe, *Metamorphosis* [10][11], utilizam-se ontologias para integrar e expôr na Web fontes heterogéneas de dados. A ideia principal é extrair um pequeno conjunto de metadados de cada fonte de informação, construir uma rede semântica de conceitos e relações entre conceitos e povoá-la com os metadados previamente extraídos das fontes. Mais tarde, esta base de conhecimento é utilizada para guiar semanticamente o utilizador na navegação até chegar aos dados que efectivamente lhe interessam.

Dos vários formalismos e modelos inerentes possíveis, foram seleccionados os Topic Maps (TM) devido ao seu elevado grau de abstracção e independência relativamente aos dados (o que os torna ideais para quem quer gerar ferramentas automáticas). Os TM são suficientemente abstractos para especificar qualquer coisa e suficientemente formais para que seja possível a criação de ferramentas de processamento e de navegação.

Nas próximas secções explicam-se alguns conceitos dos TM e dá-se uma ideia da interface Web que é possível gerar.

5.1 Topic Maps

Os TM [12], [13], [14], são um formalismo para a representação de conhecimento sobre a estrutura de um determinado sistema de informação. Qualquer conceito por mais abstracto ou concreto que seja representa um tópico. Os tópicos podem ter associações com outros tópicos e podem ter ocorrências (instâncias concretas do lado do sistema de informação). A informação associada a um tópica é calculada em função das associações e ocorrências desse tópico. Um conjunto de tópicos e respectivas associações e ocorrências é um TM.

Ao permitir criar um mapa virtual sobre as fontes de informação, esta tecnologia permite-nos obter diferentes vistas reflectindo diferentes organizações da informação sem que as fontes de informação sejam alteradas.

5.2 Interface Web

O sistema ADRIAN tem um gerador de interfaces que recebendo uma ontologia especicada com TM, produz um conjunto de páginas Web que permitem a navegação na rede conceptual especificada e aceder aos recursos de informação apontados pelas ocorrências no TM.

Assim o utilizador do sistema só necessita de especificar a sua rede conceptual de modo a obter a interface Web correspondente.

6. Conclusões

Como foi referido ao longo do artigo, o sistema ADRIAN ainda está a ser desenvolvido. Muitos componentes ainda estão na sua versão beta e falta integrar e automatizar algumas fases do processo.

Olhando para o estado da arte, podemos dividir a produção de conteúdos em duas áreas: metadados e conteúdo. Até ao momento, a comunidade tem-se preocupado com a metainformação associada àquilo que se convencionou designar por “Learning Object”. Foram desenvolvidas normas e algumas estão a ser utilizadas pelos grandes grupos de interesse nesta área: IMS EML [14] e LOM [15]. Estas normas foram desenvolvidas para toda a comunidade o que faz com que elas sejam grandes e complexas o que pode inviabilizar a sua utilização. No desenvolvimento das nossas aplicações, incluímos alguma metainformação mas centrámo-nos essencialmente no conteúdo. Está a ser iniciado um

novo projecto que visa tornar o ADRIAN compatível com aquelas normas sempre com o objectivo de manter a tarefa do utilizador (neste caso o docente) o mais simples possível.

Referências bibliográficas

1. Ramalho, J.C,, Henriques, P.R.: XML & XSL: da Teoria à Prática, 1ª edição, FCA, 2002.
2. Maler, E., Andaloussi J. : Developing SGML DTDs: From Text To Model To Markup, Prentice-Hall, 1998.
3. Megginson D. : Structuring XML Documents, Prentice-Hall, 1999.
4. Goldfarb C.: The SGML Handbook, Clarendon Press, Oxford, 1990.
5. Herwijnen E.V.: Practical SGML, Kluwer Academic Publishers, 1994.
6. Harold E.R., Means W.S.: XML in a Nutshell, O'Reilly and Associates, 2001.
7. Cagle K., Duckett J., Griffin O., Mohr S., Norton F., Ozu N., Rees I., Tennison J., Williams K.: Professional XML Schemas, Wrox Press, 2001.
8. Williams K.: Professional XML Databases, Wrox Press, 2000.
9. Tidwell D.: XLST, O'Reilly, 2001.
10. Librelotto G., Ramalho J.C., Henriques P.R.: Geração automática de interfaces web para sistemas de informação: Metamorphosis, in CoopMedia, 2003, ISEP, Porto- Portugal, 2003.
11. Librelotto G., Ramalho J.C., Henriques P.R.: TM-builder: Um construtor de ontologias baseado em topic maps, In CLEI'03, La Paz – Bolívia, 2003.
12. Biezunsky M., Bryan M., Newcomb S.: ISO/IEC 13250 – Topic Maps, ISO/IEC/JTC 1/SC34, 1999.
13. Park J., Hunting S.: XML Topic Maps, Addison-Wesley, 2003.
14. http://eml.ou.nl/eml-ou-nl.html, 2003.
15. http://ltsc.ieee.org/wg12/, 2003.