Ambiente para Disponibilização de Cursos na *Web* Utilizando Autômatos Finitos com Saída

Graciela Cristina Bernardes Lima[1], Paulo Blauth Menezes[1], Renata Zanella[1], Júlio Henrique Araújo Pereira Machado[1]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

{gcblima,blauth,renataz,jhapm}@inf.ufrgs.br

Abstract. *This paper presents XHA (EXtensible Hyper-Automaton), an environment that supports the inclusion and organization of instructional materials and the presentation of instructional hyperdocuments as Web courses. XHA uses Finite Automata with Output as a structural model for organizing hyperdocuments, in a way that the course navigational structure is kept separated from the instructional material. XHA contributed to Software Engineering, Hypermedia, Computing in Education and Formal Systems areas developing a simple and efficient model of structuring hyperdocuments on the Web.*

Resumo. *Neste artigo é apresentado o XHA (EXtensible Hyper-Automaton), um ambiente que permite a inclusão e organização do material instrucional por parte do professor e a apresentação de hiperdocumentos instrucionais como cursos na Web para o aluno. O XHA utiliza Autômatos Finitos com Saída como modelo estrutural para a organização de hiperdocumentos, de maneira que a estrutura navegacional do curso permanece separada do material instrucional. O XHA contribuiu para as áreas de Engenharia de Software, Hipermídia, Informática na Educação e Sistemas Formais, com o desenvolvimento de um modelo simples e eficiente para a estruturação de hiperdocumentos na Web.*

1. Introdução

A *Web* vem se apresentando um meio cada vez mais interessante para o desenvolvimento de sistemas de ensino. Isso se justifica dado o fato que nesse meio é possível distribuir o conhecimento em larga escala, potencialmente para todo o mundo, reduzir os custos de distribuição, corrigir e atualizar mais facilmente o conteúdo, diversificar as técnicas de ensino com a utilização de recursos multimídia, além de facilitar o trabalho colaborativo entre alunos e professores.

Com base nisso foi desenvolvido o XHA [Machado e Menezes 2000, Machado et al. 2000, Machado et al. 2001, Machado e Menezes 2001], um ambiente que permite a inclusão e organização do material instrucional por parte do professor e a apresentação de hiperdocumentos instrucionais como cursos na *Web* para o aluno. O XHA baseia-se no **Hyper-Automaton**, que utiliza o formalismo de *Autômatos Finitos com Saída* como um modelo estrutural para a organização de hiperdocumentos [Machado et al. 1999, Menezes e Machado 2000]. Nesse modelo, cada autômato define um curso e referencia um conjunto de hiperdocumentos independentes, de maneira que a **estrutura navegacional do curso permanece separada do material instrucional**. Esse mecanismo estimula e facilita o reuso do material instrucional em diversos cursos, sob as mais diferentes formas, por diferentes professores.

No XHA a estrutura navegacional é armazenada em um banco de dados relacional, enquanto que os conteúdos instrucionais são armazenados em arquivos XML. Cada conjunto de

conteúdos (curso) tem associado a si uma *folha de estilos*. Ao gerar uma saída, o documento XML passa por um processo de *transformação*, ou seja, a folha de estilos do curso é aplicada ao documento XML, obtendo-se como resultado um documento com um *layout* apropriado para ser enviado ao *browser* e apresentado ao aluno. Com esse mecanismo, é possível utilizar os mesmos conteúdos, porém associando-os a uma folha de estilos diferente, e obter como resultado um curso diferente, tanto no estilo quanto na estruturação do *layout* de suas páginas.

O objetivo deste artigo é apresentar o XHA, dessa forma não serão efetuadas comparações com outros ambientes de ensino para a *Web*. Entretanto vale a pena ressaltar que a reutilização do material instrucional, da maneira que é proposta e implementada no XHA, representa um grande diferencial com relação a outros sistemas com os mesmos propósitos, como por exemplo, WebCT [WebCT 2004] e Eureka [Eureka 2004].

O XHA trouxe importantes contribuições para as áreas de Engenharia de Software, Hipermídia, Informática na Educação e Sistemas Formais, tais como: utilização de formalismos clássicos da Teoria de Autômatos aplicados a sistemas hipermídia de ensino na *Web*; desenvolvimento de um modelo simples e eficiente para a estruturação de hiperdocumentos na *Web*; desenvolvimento de implementações padronizadas; manipulação mais eficiente de documentos; facilitação da criação de documentos instrucionais por parte do professor.

Este artigo está organizado da seguinte forma: na seção 2 é apresentado o Hyper-Automaton, sistema no qual o XHA se baseia; na seção 3 são apresentadas as tecnologias utilizadas no desenvolvimento do XHA; na seção 4 são apresentados aspectos relativos à implementação do XHA; na seção 5 encontra-se a conclusão, com os comentários finais e trabalhos futuros e por fim as referências bibliográficas.

2. Hyper-Automaton

O Hyper-Automaton é um modelo que suporta o gerenciamento e apresentação de hiperdocumentos instrucionais como cursos na *Web*. O sistema utiliza o formalismo de *Autômatos Finitos com Saída* como um modelo estrutural para a organização de hiperdocumentos.

No Hyper-Automaton, cada autômato define um curso e referencia um conjunto de hiperdocumentos independentes (Unidades de Informação), tal que a estrutura navegacional do curso permanece separada do material instrucional. Esse mecanismo estimula o reuso do material instrucional em diversos cursos além de possibilitar a construção de cursos com diferentes abordagens e objetivos utilizando os mesmos hiperdocumentos. Algumas das vantagens desse modelo são apresentadas abaixo:

- Independência entre a base de hiperdocumentos e a estrutura de controle da aplicação hipermídia (autômato), cuja alteração não influi nas páginas e vice-versa. Assim sendo, é possível especificar vários conjuntos de *links* sobre um mesmo corpo de hiperdocumentos mantendo os objetos separados da estrutura de navegação.
- Suporte à criação de *links* de e para qualquer documento.
- Suporte à elaboração de seqüências instrucionais com objetivos específicos e abordagens diferentes, com possibilidade de oferecer estudo individualizado.
- Opção de limitação da liberdade do aluno ao explorar hipertextos, pois com o modelo de autômatos pode ser construído um material hipertexto fortemente hierarquizado, no qual os conteúdos são organizados em múltiplos níveis (definidos pela ordem dos estados do autômato).

3. As Tecnologias Utilizadas

Nos últimos anos as aplicações *Web* têm atendido um público maior, mais exigente e com necessidades específicas. O resultado disso é que essas aplicações acabaram se tornando maiores e mais complexas, o que por sua vez abriu espaço para que as tecnologias voltadas para a Internet sofressem uma sensível evolução. Esse cenário reforçou a idéia de efetuar a reestruturação do Hyper-Automaton, atualizando-o tecnologicamente, bem como flexibilizar as saídas geradas pelo autômato, utilizando XML e XSL (XML Stylesheet Language) ao invés de somente HTML (HyperText Markup Language).

3.1. XML (EXtensible Markup Language)

O HTML (HyperText Markup Language) foi e continua sendo o meio mais utilizado para publicação de conteúdo na *Web*. Entretanto o HTML possui sérios problemas: falta de extensibilidade; falta de validação e de uma estrutura claramente definida para os documentos; mescla de informações a respeito da formatação e do conteúdo dentro de um mesmo documento; fornecimento de poucos recursos para o controle da aparência das páginas.

Quando comparado ao HTML, o XML oferece várias vantagens, como por exemplo: melhor controle com relação ao *layout*; redução da carga de trabalho do servidor *Web*, dado que é possível transferir maior processamento para o cliente (*browser*); suporte a vários tipos de *links*; maior facilidade para exibir páginas longas; separação entre *layout* e conteúdo (a informação a respeito do *layout* é mantida em *folhas de estilo*); maior clareza e legibilidade nos documentos XML que, em geral, contêm informações a respeito de um domínio, que se encontram refletidas em um conjunto de *tags* criadas pelo próprio time de desenvolvimento ou então oriundas de um vocabulário amplamente documentado e difundido; conteúdo altamente estruturado e hierarquizado que pode ser aninhado em qualquer nível de complexidade; possibilidade de opcionalmente vincular regras gramaticais a um documento XML (por meio de DTD, XMLSchema, etc.) de modo que seja possível verificar sua validade ou não [Pitts-Moultis and Kirk 2000].

3.2. XSL (Extensible Stylesheet Language)

A idéia com XML é que o estilo e a apresentação do documento sejam fornecidos por meio de *folhas de estilo*. Com esse intuito foi desenvolvido o XSL, um mecanismo de folhas de estilo para o XML projetado e expresso em XML. O *processador de folhas de estilo* XSL recebe como entrada um documento XML e uma folha de estilo XSL e produz por fim um documento formatado de acordo com as especificações pretendidas pelo *designer* da folha de estilo [Adler et al. 2001]. Há dois aspectos envolvidos no processo de geração dessa saída: em primeiro lugar é necessário construir uma árvore de resultado a partir do documento XML fonte (*tree transformation*) e em segundo lugar interpretar a árvore de resultado para produzir a apresentação final formatada adequadamente (*formatting*). A grande vantagem do processo de transformação é a possibilidade de modificar significantemente a estrutura da árvore fonte produzindo uma árvore de resultado diferente da original. Isso significa que é possível obter várias visões de um mesmo conteúdo simplesmente rearranjando-o de maneira diferente.

3.3. Java

O XHA foi projetado para a *Web*. Assim sendo, dentro da proposta de atualizá-lo tecnologicamente, fez-se necessário optar por uma tecnologia que pudesse fornecer suporte a aplicações no lado servidor. A tecnologia *Java* foi escolhida por exibir as seguintes características: ambiente multiplataforma, isso significa que é possível executar um mesmo código em diferentes plataformas sem necessidade de recompilação; modelo de programação orientado a objetos; gerenciamento automático de memória; tratamento hierarquizado e

padronizado de exceções; rica diversidade de APIs (Application Programming Interface), altamente padronizadas além de extensamente documentadas e em geral fáceis de utilizar.

4. O Sistema Implementado

A máquina de estados do Hyper-Automaton gerencia rígidos fragmentos de HTML. No XHA essa máquina foi reimplementada utilizando as tecnologias XML e Java com o intuito de aumentar o leque de possibilidades com relação ao material disponibilizado, suportar recursos de *links* não convencionais, renderizar (interpretar e exibir) uma ampla gama de caracteres e fórmulas matemáticas, suportar arquivos multimídia e facilidades futuras de edição de conteúdo, estilos e regras.

A figura abaixo exibe a arquitetura do XHA:

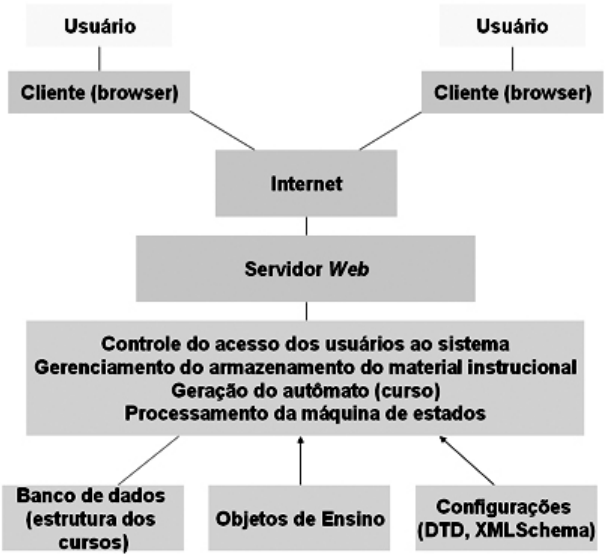

Figura 1. A arquitetura do XHA

Nas subseções seguintes são apresentados os principais elementos dessa arquitetura.

4.1. Objetos de Ensino

O XHA armazena os recursos instrucionais como *objetos de ensino*. Um objeto de ensino é um recurso instrucional (imagem, vídeo, animação, texto, *applet* Java, etc.) que pode ser utilizado para prover alguma experiência de aprendizagem para o aluno. Um objeto de ensino pode ser combinado com outros a fim de formar objetos mais complexos.

Uma questão de suma importância ao se projetar objetos de ensino é a noção de *granularidade*, isto é, como serão manipulados os objetos de ensino para a formação de objetos mais complexos. Atualmente há duas correntes de pensamento: a primeira especifica o nível de granularidade de um objeto de ensino conforme o *grau de composição* de objetos menores para compor objetos de ensino maiores; a segunda e mais recente, leva em consideração a *coesão* do objeto de ensino, ou seja, quanto mais centrado em um único conceito, menor a sua granularidade.

Acreditando-se que o reuso de material instrucional é potencializado pelo uso de objetos com maior coesão, foi definido o *objeto elementar de ensino*, ou seja, o menor fragmento de material instrucional que não seja definido em termos de outros objetos de ensino. Assim sendo, é possível medir com maior precisão a complexidade de um curso, já que se leva em conta a mensagem de ensino e não o número de objetos que compõem essa mensagem.

Os fragmentos de HTML utilizados na versão anterior do Hyper-Automaton já foram construídos segundo essa noção de coesão, mas por se tratarem de arquivos HTML, esses objetos constituíam o menor nível de abstração manipulável no sistema. Com o XHA a idéia era utilizar e manipular esses mesmos objetos de ensino de forma mais inteligente e com maior poder de reuso. Na nova proposta de implementação, esses objetos foram analisados internamente, de maneira que se pudesse identificar seus elementos constituintes, como textos, imagens, listas, definições, exemplos, etc. Com base nessa análise, foi elaborada uma *definição*, em *XMLSchema*, do objeto elementar de ensino, que traduz para XML os elementos antes em marcação HTML. A definição do objeto foi especificada de tal forma que seus componentes internos e externos (textos, figuras, arquivos externos) fossem facilmente identificados e manipulados.

A estruturação de objetos de ensino utilizando XML possibilita efetuar inclusões e modificações na definição de forma ordenada e controlada. Adicionalmente, é possível a existência de mais de uma versão de definição e nesse caso, cada objeto deve indicar qual versão deve ser usada na sua validação. Por fim, para que o XHA possa manter a compatibilidade com cursos legados (que não utilizam os recursos XML de estruturação de documentos) é possível que os tipos de fragmento sejam HTML.

4.2. Banco de Dados

Uma das características mais importantes do XHA é a separação entre a estrutura do curso e a base de hiperdocumentos, organizada sob o enfoque de objetos de ensino conforme descrito na subseção anterior. O autômato que define um curso é uma estrutura formal pré-estabelecida capaz de representar a *estrutura navegacional*. Esse autômato é armazenado em um banco de dados relacional. Já os objetos elementares de ensino, que fazem parte do curso, definem o *conteúdo* desse curso. Esses objetos ficam armazenados em arquivos XML. A essa altura é interessante lembrar que o XML tem uma estrutura hierárquica, e essa característica o torna substancialmente adequado e vantajoso, quando comparado a uma estrutura relacional, pois ele pode ser usado de uma maneira mais simples e direta para projetar estruturas de objetos de ensino.

4.3. Máquina de Estados

O sistema é responsável por processar a máquina de estados por meio de consultas ao banco de dados relacional. Posteriormente uma saída é devolvida para o *browser* (que tanto pode ser HTML quanto PDF (Portable Document Format) e até mesmo WML (Wireless Markup Language)). A saída é produzida ao se aplicar as transformações XSL a um arquivo XML. Como qualquer máquina de estados, seu funcionamento ocorre mediante parâmetros de entrada, que nesse caso são três: um curso, um estado e uma palavra de entrada. No exemplo abaixo o curso acessado é o de identificação 1, estado 3 e palavra de entrada 7.

*http://teia/webapps/hypno/hypno.jsp?**course=1&state=3&word=7***

O parâmetro *course* é sempre obrigatório, já que é a partir do seu valor que se define a função de transição a ser aplicada sobre as entradas.

É importante destacar que os cursos disponibilizados no XHA têm páginas iniciais independentes, com apontadores para as várias seções dos cursos, servindo como referência rápida aos estudantes. Esses pequenos portais de cursos são importantes já que não faz sentido que os usuários tenham que decorar longas URLs (Uniform Resource Locartor) com uma série de variáveis e parâmetros.

O processamento da máquina de estados pode ser dividido em quatro etapas:

1ª Etapa: inicialmente é efetuada uma verificação nos parâmetros recebidos do usuário. Em seguida, algumas informações pertinentes são coletadas, como por exemplo nome, tipo da

máquina (*Moore*, *Mealy*) e estado inicial do curso. Caso o estado não seja fornecido, a máquina vai automaticamente para o estado inicial e continua o processamento. Uma vez definido o estado atual, um documento XML virtual de saída é criado contendo uma seção para **metadados**, outra para os **objetos de ensino** e por fim uma seção para o **conjunto de palavras de entrada** (*links*).

2ª Etapa: o objetivo nesta etapa é descobrir quais objetos de ensino fazem parte da palavra de saída do autômato. Isso é feito através de uma consulta ao banco de dados. Esses objetos, que se encontram armazenados em documentos XML, são lidos e seus conteúdos anexados ao documento XML de saída criado na 1ª etapa.

3ª Etapa: o objetivo nesta etapa é descobrir as palavras de entrada que definem as transições para outros estados a partir do estado atual, ou seja, neste estágio serão descobertas as informações sobre a navegação do curso, como *links*, referências cruzadas, etc. Isso é feito através de uma consulta ao banco de dados. As identificações (palavras de entrada) são então anexadas ao documento XML de saída criado na 1ª etapa.

4ª Etapa: a última etapa do método é a que de fato diferencia a nova versão em XML da implementação anterior em *Perl*: uma vez gerado o documento XML de saída (com metadados, objetos de ensino e transições), ao invés de simplesmente apresentar os objetos de ensino concatenados na saída no formato HTML, um arquivo de transformação, **específico para cada curso**, é aplicado ao documento gerado, dando forma, sabor e personalização na apresentação.

Cada curso tem seu próprio arquivo de transformação XSL, que define a estrutura da página de saída: cabeçalhos, menus, cores, *layout*, etc. O responsável pelo curso poderá escolher entre vários *layouts* padronizados para seu curso e ainda terá a opção de desenvolver o seu próprio. Essa personalização não era possível na versão anterior pois a formatação e diagramação eram inerentes aos fragmentos HTML utilizados. Uma observação bastante interessante é que um outro curso pode utilizar os mesmos objetos de ensino mas com um arquivo de transformação XSL diferente obtendo conseqüentemente uma saída completamente diferente.

Na versão atual do sistema, toda a transformação XML e XSL é feita no *servidor*. Isso significa que o servidor deve retornar ao usuário as informações já em formato HTML. Hoje em dia vários *browsers* já suportam XML e são capazes de usar arquivos XSL para transformações. Isso significa que é possível diminuir a carga do servidor enviando para o cliente o documento XML gerado, acompanhado de seu arquivo XSL, deixando o processamento de transformação para a máquina do cliente.

5. Conclusão

A produção de material instrucional para a *Web* demanda esforço por parte do professor, assim sendo é desejável que esse material possa ser reutilizado em diferentes cursos, sob as mais diferentes formas, por diferentes professores. O XHA organiza sua base de hiperdocumentos possibilitando que o material instrucional seja reutilizado. Tal reutilização é possível devido ao fato do XHA apresentar as seguintes características: separação entre o conteúdo e a apresentação, o que possibilita que um mesmo conjunto de hiperdocumentos possa ser exibido de maneira diferente, simplesmente trocando sua folha de estilos; eliminação das limitações com relação à execução de aplicações proprietárias e caracteres incompatíveis em *browsers* (com a utilização de entidades XML); maior fragmentação da informação, o que conseqüentemente aumenta as possibilidades de reuso e manipulação; conteúdo instrucional construído sobre um padrão de elaboração (objeto elementar de ensino); possibilidade de inclusão de uma maior diversidade de recursos didáticos; possibilidade de estender a definição do objeto elementar de ensino para suportar mais elementos quando necessário.

O trabalho desenvolvido abre oportunidades que podem ser exploradas em trabalhos futuros, como por exemplo:

- Com a finalidade de facilitar a inclusão de material instrucional na base de conteúdos, pode ser desenvolvido um módulo que tome como entrada um documento, por exemplo um hipertexto, e o mapeie para a estrutura de armazenamento de objetos de ensino do XHA. Além disso, é viável a construção de uma interface simples, direcionada para usuários leigos, que facilite a criação do material instrucional por meio de *wizards* e *templates*, incluindo recursos para a edição de estilos.
- Criação de um mecanismo que possibilite exportar o curso de maneira que possa ser acompanhado *offline* pelo aluno. Essa alternativa é viável considerando-se a estrutura de armazenamento e organização dos cursos e material instrucional do XHA.
- Exploração dos recursos da MathML (Mathematical Markup Language), que é uma aplicação XML que objetiva habilitar o uso da notação matemática na *Web*. Essa alternativa é bastante interessante nos casos em que se deseja disponibilizar cursos de cunho tecnológico ou matemático.
- O sistema possibilita que através da aplicação de diferentes arquivos de transformação possam ser geradas saídas diferenciadas para cada curso. É interessante que se possa utilizar o mesmo curso nos mais diversos dispositivos de exibição, não somente em *browsers Web*, mas também em celulares, *palmtops*, etc. Atualmente o sistema permite associar apenas um arquivo de transformação para cada curso. É possível estender esse mecanismo de maneira que cada curso possua vários arquivos de transformação. Com isso o XHA irá identificar o dispositivo utilizado para o acesso e irá aplicar o arquivo de transformação adequado.

Embora o XHA apresente uma série de vantagens, como foi apresentado neste artigo, ele não possui nenhum mecanismo que possibilite avaliar o aluno. Com esse intuito, e visando dar continuidade ao trabalho iniciado, está sendo desenvolvido pelo mesmo grupo de pesquisa o *EASy* (Evaluation Automatic Generation System for Web based on Hyper-Automaton), que apresenta dois objetivos principais: suprir algumas funcionalidades desejadas em sistemas de avaliação via *Web*, como flexibilidade na autoria de avaliações, reuso de questões, recursos de edição para questões e avaliações, avaliações adaptativas e personalizadas; e prover um sistema de avaliação para o ambiente de ensino XHA.

6. Agradecimentos

Este trabalho está sendo parcialmente financiado pelo CNPq (Projetos HoVer-CAM, GRAPHIT, E-Automaton) e FINEP/CNPq (Projeto Hyper-Seed).

Os autores agradecem profundamente César Costa Machado e Leonardo Penczek que contribuíram diretamente no desenvolvimento do XHA.

Referências Bibliográficas

Pitts-Moultis, N., Kirk, C. (2000) "XML Black Book", São Paulo: MAKRON Books, p. 15-19.

Machado, J. P., Penczek, L., Morais, C. T. Q., Menezes, P. B. (1999) "Autômatos Finitos: um Formalismo para Cursos na Web", In: XIII Simpósio Brasileiro de Engenharia de Software, Florianópolis, p. 213-223.

Menezes, P. B., Machado, J. P. (2000) "Hyper-Automaton: Hypertext Framework with Categorial Operations", In: IV Simpósio Brasileiro de Linguagens de Programação, Recife, p. 29-47.

Machado, C. C., Menezes, P. B. (2000) “Características e Vantagens na Estruturação de Documentos XML na WWW para EAD”, In: I Workshop Informática na Educação, Passo Fundo, p. 99-105.

Machado, C. C., Machado, J. P., Grandi, R. H., Menezes, P. B. (2000) “Utilização do XML no Sistema Hyper-Automaton”, In: 3rd International Symposium on Knowledge Management/Document Management, Curitiba, p. 439-455.

Machado, C. C., Federizzi, G. L., Menezes, P. B. (2001) “Flexibility and Adequacy of the Output's Layout of the Content Displayed in The Hyper-Automaton System”, In: 2nd International Conference on Internet Computing, Las Vegas, p. 424-430.

Machado, C. C., Menezes, P. B. (2001) “Definition and Application of Rules for the Adequate Designing of XML Documents for the Hyper-Automaton System” In: Fifth International Query Processing and Multimedia Issues in Distributed Systems Workshop em conjunto com a 12th International Conference on Data and Expert Systems Application e IEEE Computer Society, Munique, p. 100-105.

Adler, S., Berglund, A., Caruso, J., Deach, S., Graham, T., Grosso, P., Gutentag, E., Milowski, A., Parnell, S., Richman, J., Zilles, S. (2001) “Extensible Stylesheet Language (XSL) Version 1.0, W3C Recommendation”, http://www.w3.org/TR/xsl/.

WebCT (2004) “WebCT – E-Learning System for Educational Institutions”, http://www.webct.com/.

Eureka (2004) “Eureka - Ambiente de Aprendizagem Colaborativa à Distância”, http://eureka.pucpr.br/.