IME: Um Modelo e Gerador de Editores de Atividades de Aprendizagem em IMS *Learning Design*[#]

Aladir Ferreira Silva Júnior[1,2], Leandro Roberto Silva[2], Clovis Torres Fernandes[1]

[1]Divisão de Ciência da Computação – Instituto Tecnológico de Aeronáutica (ITA) – Laboratório de Aprendizagem e Interação (LAI)
CEP 12228-901 – São José dos Campos – SP – Brasil

[2]Coordenação de Informática – Instituto Federal de Educação, Ciência e Tecnologia de Goiás (IFG) – Câmpus Jataí
CEP 75804-020 – Jataí – GO – Brasil

aladir@gmail.com, leandroroberto.br@gmail.com, clovistf@uol.com.br

Abstract. *Many editors have been developed to assist teacher in creating learning activities using the IMS Learning Design standard. However, such editors have some limitations, especially in supporting users to structure digital teaching materials based simultaneously on more than one pedagogical technique. Instructional Model Environment (IME) is a proposed model and generator that allows developers to create a family of LD editors, which aims to support the teacher in creating learning activities based on various pedagogical techniques.*

Resumo. *Fornecer auxílio ao professor na criação de atividades de aprendizagem com o uso de um editor de atividades de aprendizagem no padrão IMS Learning Design (LD), não é tarefa trivial dada a diversidade de estratégias pedagógicas para as quais se pode requerer apoio. O fornecimento restrito desse apoio a apenas uma ou outra estratégia e a impossibilidade de uso de mais de uma estratégia em uma mesma atividade são algumas das limitações dos atuais editores LD. Como resposta a tais limitações, propõe-se o IME (Instructional Model Environment), um modelo e gerador de editores que permite criar uma família de editores LD, que oferece apoio ao professor na criação de atividades de aprendizagem seguindo estratégias pedagógicas diversas.*

1 Introdução

No conjunto de softwares desenvolvidos para auxiliar o professor a estruturar material didático digital estão os editores de atividades e ações de aprendizagem usando o padrão de representação IMS *Learning Design* ou simplesmente editores LD. Empregar a LD [IMSLD 2014] para representar atividades instrucionais requer conhecimento técnico do usuário tanto em relação à própria representação computacional (viés tecnológico no uso da LD), no caso em *eXtensible Markup Language* – XML), como em relação às

[#] Agradecemos à FAPESP pelo apoio ao projeto #05/50621-2, programa TIDIA-Ae Aprendizado Eletrônico, em que a base que propiciou desenvolver este trabalho nasceu e prosperou!

estratégias pedagógicas a serem aplicadas na estruturação do material didático da atividade (viés pedagógico no uso da LD).

As estratégias pedagógicas ou instrucionais são técnicas que o professor pode usar para engajar o aprendiz e facilitar a aprendizagem [Dabbagh 2005]. O conjunto de tais estratégias é denominado modelo instrucional (MI). Pode-se comparar um MI ao "modo de fazer" de uma receita culinária. Um MI pode, por exemplo, auxiliar o professor a estruturar seu material didático guiando-o pelas fases "Introdução", "Tarefa", "Processo", "Avaliação" e "Conclusão", seguindo as estratégias instrucionais propostas pela técnica *WebQuest* [Dodge 2014]. O resultado dessa estruturação em LD, quer seja uma lição ou um curso completo, se intitula *Unit of Learning* (UoL).

Em função da diversidade de técnicas (estratégias) pedagógicas existentes e a existir, o fornecimento de apoio ao usuário por parte dos editores LD quanto ao viés pedagógico, não pode ser algo estático e se constitui em uma tarefa complexa. Talvez este seja um dos motivos de se ter à disposição diversos editores LD genéricos e poucos editores LD específicos. Um editor LD genérico, p. ex. RELOAD Editor [RELOAD 2014], fornece apoio ao usuário apenas em relação ao viés tecnológico no uso da LD, ao passo que um editor LD específico, p. ex. Recourse [Griffiths *et al*. 2010] e WQEditor [Camargo *et al*. 2010], adicionalmente fornece apoio quanto ao viés pedagógico no uso da LD, permitindo ao usuário criar uma UoL seguindo orientação pedagógica explicitada no editor por meio de MIs, onde os MIs são baseados em teorias pedagógicas consagradas ou em práticas bem sucedidas.

Apesar de fornecerem mais apoio aos seus usuários, os editores LD específicos apresentam limitações especialmente quanto ao apoio baseado em MIs. Alguns fornecem apoio em um único MI (p. ex. WQEditor);, outros fornecem apoio em diversos MIs, o que equivale a se ter um editor para cada MI, mas o usuário pode escolher apenas um editor para criar uma UoL (p. ex. Recourse). Dos três editores LD que permitem usar simultaneamente (por intercalação) mais de um MI/editor para uma mesma UoL, o Collage [Hernández-Leo *et al*. 2005] não permite criar novos MIs, e o LAMS [LAMS 2014] e o Open GLM Prolix [OGP 2014] exigem que seus usuários tenham que finalizar manualmente a intercalação de editores gerando re-trabalho.

Fruto desta pesquisa, que pode ser caracterizada como aplicada, se apresenta o *Instructional Model Environment* (IME), um modelo formal e gerador de editores LD, que permite gerar uma família de editores LD específicos para as diversas abordagens pedagógicas, como um possível caminho para superação das limitações apresentadas pelos representantes do estado da arte da autoria em IMS *Learning Design*. Na Seção 2, apresentamos o contexto desta pesquisa. O IME é apresentado na Seção 3. Na Seção 4, apresentamos as principais conclusões e possibilidades de pesquisa futura.

2 Contexto da Pesquisa

Apesar da diversidade de editores LD, constatam-se limitações em todos eles, especialmente quanto ao apoio aos usuários em relação à estruturação do material didático digital com base em técnicas pedagógicas, a saber:

a. A maioria dos editores LD não oferece apoio ao professor na estruturação de seu material didático quanto aos modelos instrucionais. Entre eles estão o Editor LD LAI [Fernandes *et al*. 2012] e o RELOAD.

b. Alguns editores LD foram desenvolvidos para atender a apenas um modelo instrucional (MI); ou seja, se usuário do editor necessitar alterar uma estratégia instrucional (fase do MI) ou incluir uma nova, tais necessidades não poderão ser atendidas. Um exemplo de editor com essa limitação é o WQE Editor.

c. Outros editores LDs oferecem ao professor vários MIs, porém não permitem seu uso de modo conjunto (intercalado) na estruturação de uma unidade de aprendizagem (p. ex. um curso). São exemplos os editores CADMOS e Recourse.

d. Uma limitação que se considera grave e atinge a grande maioria dos editores LD é não permitir reusar material já estruturado por outro editor, apesar de serem todos editores de atividades e ações instrucionais seguindo o padrão LD. Apenas os editores RELOAD, Recourse e Collage conseguem importar material feito por outros editores sem maiores problemas.

e. Por último, mas não menos importante, nos três editores LDs que dispõem de auxílio ao professor com vários MIs e permitem intercalação de MIs, ainda é notada uma falta de flexibilidade em não permitir ao usuário a criação de novos MIs (p. ex. Collage) ou, quando permite a criação de novos MIs, em exigir retrabalho manual por parte do professor em relação à forma de intercalação possível entre os MIs (p. ex. LAMS e Open GLM Prolix).

A falta de flexibilidade na criação de novos editores (MIs) é um mal que deve ser combatido a todo custo, haja vista que o professor precisa também definir o material didático de acordo com suas necessidades, sob pena de possível desmotivação na realização desta tarefa e com prováveis conseqüências negativas para os alunos. O professor poderá seguir alguma abordagem pedagógica baseada em teorias consagradas (p. ex. Aprendizagem Significativa e Construtivismo), mas também poderá definir seu material a partir de uma abordagem baseada em sua própria prática.

Percebe-se pela revisão de literatura e pelo uso prático de tais ferramentas em nosso laboratório que, de modo geral, os editores LD da literatura não oferecem apoio pedagógico adequado aos usuários, especialmente em relação à flexibilidade na estruturação e composição baseada em modelos instrucionais. Chamamos a isso neste trabalho de falta de apoio pedagógico flexível. Dentre outras, as limitações acima apontadas têm o seguinte como causa:

- A aderência dos editores de forma estrita à especificação LD, assumindo as limitações impostas por esta especificação, como as apresentadas por Burgos (2008) e, em especial quanto à possibilidade de se compor (intercalar) um material já estruturado com outro.
- A falta de definição de partes reusáveis no desenvolvimento dos editores LD, talvez por uso de uma abordagem tradicional de Engenharia de Software (ES) em seu desenvolvimento ou ainda por adotar um modelo *ad hoc* para esse fim.
- A falta de separação clara entre o editor LD desenvolvido e o MI que ele entende, o que significa mudar o editor LD quando se desejar fazer uso de um novo MI.

- O uso de arquivos adicionais para controle das facilidades ao usuário, especialmente quanto à parte de interface gráfica com o usuário, que não são reconhecidos por outros editores LD.

Duas abordagens para o desenvolvimento de um modelo de editor que pudesse superar as limitações do estado da arte até então apresentado foram analisadas: abordagem de Linhas de Produtos de Software e abordagem de Geração de Editores.

Definiu-se, com base na avaliação dos editores LD atuais e na abordagem de Geração de Editores, que os seguintes elementos precisam ser considerados quando da construção de um modelo para os editores LD com apoio pedagógico flexível: (i) o modelo deve possuir uma estrutura comum, que permita se reusar código-fonte, diminuindo assim o esforço no desenvolvimento de novos editores LD; (ii) o modelo deve evitar a compilação de código-fonte quando se precisar gerar um novo editor LD específico, pois isso poderia acarretar em mais uma etapa na geração de editores LD. Neste ponto, a interpretação de código-fonte LD, que é expresso em XML, é uma saída.

Adicionalmente (iii) o modelo deve prever, para a especificação dos editores LD a serem gerados, o uso de uma Linguagem Específica de Domínio (DSL – Domain Specific Language) para evitar que a especificação de novos editores seja uma atividade complexa. Sabe-se que a LD não permite intercalar MIs e, uma gramática abrangente, apesar da flexibilidade possível, poderia afastar o usuário devido à sua complexidade. Assim faz-se necessária uma LED para o domínio dos modelos instrucionais. A partir deste ponto se usará o termo intercalar/intercalação como sinônimo de compor/composição. A seguir se apresenta o modelo IME e o gerador de editores LD de mesmo nome.

3 IME – *Instructional Model Environment*

Nesta seção se apresenta um modelo formal denominado *Instructional Model Environment* (IME) e também um gerador de editores (ambiente IME) que segue o modelo IME. O ambiente IME é uma prova de conceito que permite a geração de uma família de editores LD específicos, sendo assim um representante do Nível 5, de acordo com a Classificação por Níveis de Autoria [Silva Júnior e Fernandes 2012]. Na Subseção 3.1, apresentamos o modelo IME. Na Subseção 3.2, apresentamos a estrutura e funcionamento dos editores IME (componentes do modelo IME) e, na Seção 3.3, apresentamos o ambiente IME.

3.1 O Modelo IME e a Linguagem de Modelos Instrucionais

O modelo IME se constitui em um orquestrador dos editores LD específicos e integráveis (editores IME). Os editores são ditos integráveis por sua capacidade de serem intercalados uns nos outros. Tais editores são constituídos por uma interface com o usuário, permissões de usuário e por uma representação do MI na Linguagem de Modelos Instrucionais (LMI).

A Linguagem de Modelos Instrucionais (LMI)

A LMI é uma linguagem que estende a especificação LD adicionando construtos que permitem, por exemplo, fazer a intercalação entre MIs e UoLs. Assim, ela conserva todos os construtos da linguagem LD com suas respectivas aplicações e restrições. A LMI é uma DSL aplicada ao domínio dos modelos instrucionais e sua representação

computacional foi, como na LD, também feita usando XML. Seu principal construto é o elemento "*learning-design-ref*", que possibilita a intercalação de MIs e UoLs, funcionalidade que permite resolver as limitações a., b. e c. apontadas na Seção 2!

O elemento "*learning-design-ref*" foi criado a partir da necessidade de se definir em cada fase do MI, representada pelo elemento *act*, uma ligação para um MI completo, representado pelo elemento *learning-design*. Com isso, tornou-se possível, a partir de uma fase de um MI (MI pai) fazer referência a um outro MI (MI filho). Voltando ao exemplo do "MI *WebQuest*", este poderia ser o MI pai, onde cada uma de suas fases seria composta de um MI filho respectivo, ou seja, um "MI Introdução", um "MI Tarefa" e assim por diante. A possibilidade de se intercalar MIs amplia a flexibilidade da estruturação de material instrucional usando o padrão LD.

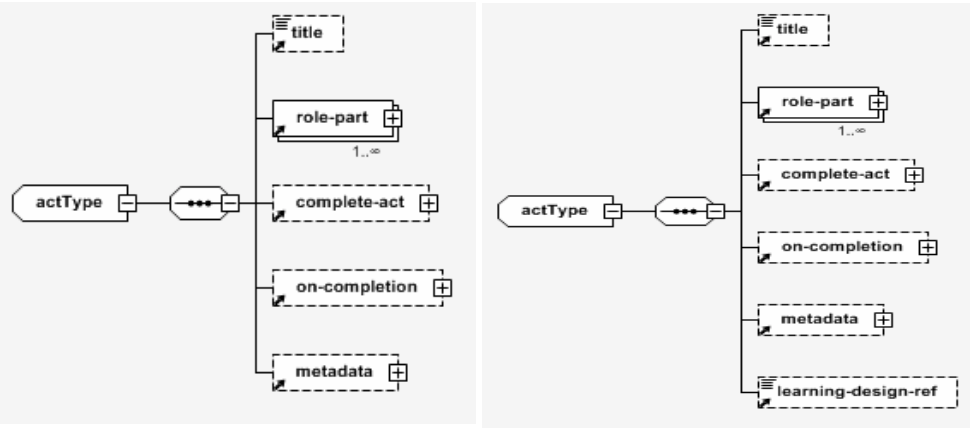

Figura 1 - actType IMSLD (esquerda) x actType LMI (direita).

```
    <ld:method>
      <ld:play identifier="play-1">
        <ld:title>Curso WebQuest</ld:title>
        <ld:act identifier="act-1">
          <ld:title>Introdução</ld:title>
          <ld:role-part identifier="rp-1">...</ld:role-part>
          <ld:learning-design-ref ref="ld-2"/>
        </ld:act>
        <ld:act identifier="act-2">...</ld:act>
        <ld:act identifier="act-3">...</ld:act>
        <ld:act identifier="act-4">...</ld:act>
        <ld:act identifier="act-5">...</ld:act>
      </ld:play>
    </ld:method>
  </ld:learning-design>
</cp:organizations>
<cp:resources>...</cp:resources>
<cp:manifest identifier="ldep-2">
  <cp:organizations>
    <ld:learning-design inherit-roles="true" identifier="ld-2" uri="" sequence-
    used="false" version="" level="A">
      <ld:title>MI - Introdução</ld:title>
      <ld:components>...</ld:components>
      <ld:method>...</ld:method>
```

Um elemento act referenciando um elemento learning-design

Figura 2 - Trecho de código LMI com referência entre MIs.

Para que fosse possível a intercalação de MIs, com a LMI alterou-se o proposto pela LD no tipo de dados *actType*, conforme se pode ver pela Figura 1. Foi feita a adição do construto *learning-design-ref* (Figura 1, à direita). Um trecho de código LMI com uma referência a outro MI é exibida na Figura 2.

O Modelo IME

Em termos funcionais o ambiente IME (*vide* Figura 3) é representado por um Construtor/Editor, por um Compositor, por um Importador, por um Exportador, pelo

Workspace e por um Modelo de Dados.O Construtor/Editor permite ao usuário criar um MI a partir do zero ou editar um MI já construído.

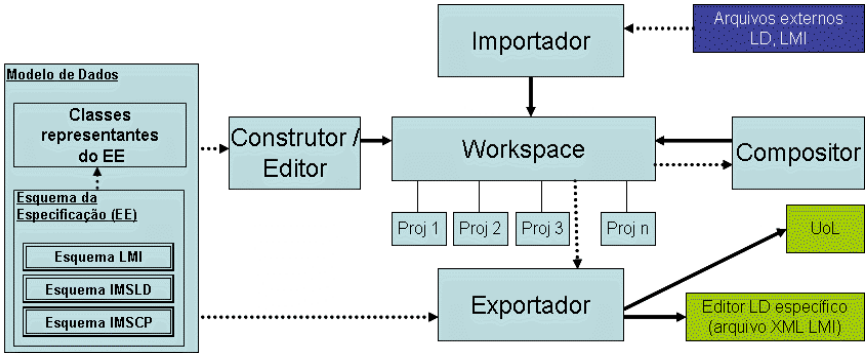

Figura 3 - Visão Funcional do modelo IME e seus componentes

Um MI neste contexto é a representação computacional feita em LMI, ou seja, um arquivo XML estruturado em uma seção de componentes (*components*) responsável por definir atividades (*activities*), papeis (*roles*) e ambientes (*environments*) e uma seção método (*method*) que permite definir a estratégia de realização de elementos parte-papel (*role-parts*) constituídos pela tríade (papel executa atividade em um ambiente).

A edição envolve, além da mudança da estrutura do próprio MI (i.e. adição ou remoção de elementos estruturais da especificação), a inclusão ou remoção de recursos, tais como arquivos de texto, vídeo e imagem, ao referido MI, transformando-o em uma Unidade de Aprendizagem (UoL). Assim, uma UoL é um MI contextualizado e, de modo inverso, um MI é uma UoL incompleta.

O Compositor permite incluir um MI (pronto) em outro MI (em construção). Permite também excluir essa intercalação, caso necessário. O Compositor atua juntamente com o Construtor/Editor quando da inclusão de MIs. Apesar de terem funcionalidades bem definidas, o Construtor/Editor e o Compositor estão representados na Estrutura Interna de Editores IME como um único módulo, o Manipulador/Interpretador, como pode ser visto na Figura 4.

O Importador é o módulo funcional responsável por obter arquivos localmente, armazenados em disco, CDs, DVDs e *pen-drives*, ou em um repositório remoto. Os arquivos devem estar no formato do padrão IMS *Content Packaging* [IMSCP 2014], empacotados ou ainda representados pelo arquivo "*imsmanifest.xml*". O exportador é responsável por gerar pelo Conversor LMI→IMSLD (*vide* Figura 4) os arquivos no formato LD para disponibilizar em um executor LD, tal como o do RELOAD.

O *Workspace* (área de trabalho) é o módulo funcional responsável por controlar todo o ambiente IME. É o núcleo do sistema IME e, como se pode ver pela Figura 3, todos os outros módulos são ligados a ele. O Construtor/Editor depende do *Workspace* para controlar o MI que ele está criando, o Compositor solicita ao *Workspace* o MI a ser incluído e devolve a ele após a intercalação terminar. O Importador obtém os dados externos e os fornece ao *Workspace* e, por fim, o Exportador solicita o MI a ser exportado ao *Workspace*.

O Modelo de Dados é composto da base de classes das especificações, que inclui a base de classes da especificação LD, a base de classes da especificação IMSCP e a base de classes da especificação LMI. Estas classes podem ser derivadas diretamente

da representação XML *Schema* (um arquivo com extensão *.xsd* que define e restringe o arquivo XML) de cada uma das especificações.

O ambiente IME é composto da base de classes do Modelo de Dados, de classes do ambiente (classes do sistema), e da base de objetos editores (arquivos LD válidos, arquivos LMI válidos, arquivos de UoLs válidos). As bases de classes do ambiente contêm as classes responsáveis pela parte estática do IME. Estas classes são representações no sentido de Orientação a Objetos (OO) e podem ser compiladas para criação do ambiente IME em uma linguagem de programação OO.

Essas bases de classes do IME que envolvem o Importador, o Exportador, o Construtor/Editor, o Compositor e o *Workspace,* juntamente com a base de objetos editores, representam um conjunto de funcionalidades do IME para resolver o problema da falta de flexibilidade na intercalação de MIs; ou seja, da intercalação de editores LD integráveis (os editores IME), cuja estrutura é discutida na próxima subseção.

3.2 Estrutura e Funcionamento dos Editores IME

Com a condição (i) de que o MI representado em LMI é uma UoL semi-estruturada, (ii) que a linguagem LD é genérica e permite estruturar material instrucional em uma variedade de MIs [Koper & Tattersall 2010], (iii) que um editor LD genérico permite ao usuário estruturar o seu material instrucional de acordo com (ii), e considerando os subsídios fornecidos pelas análises dos editores e das abordagens de Engenharia de Software, chegou-se ao Modelo dos Editores IME ilustrado na Figura 4.

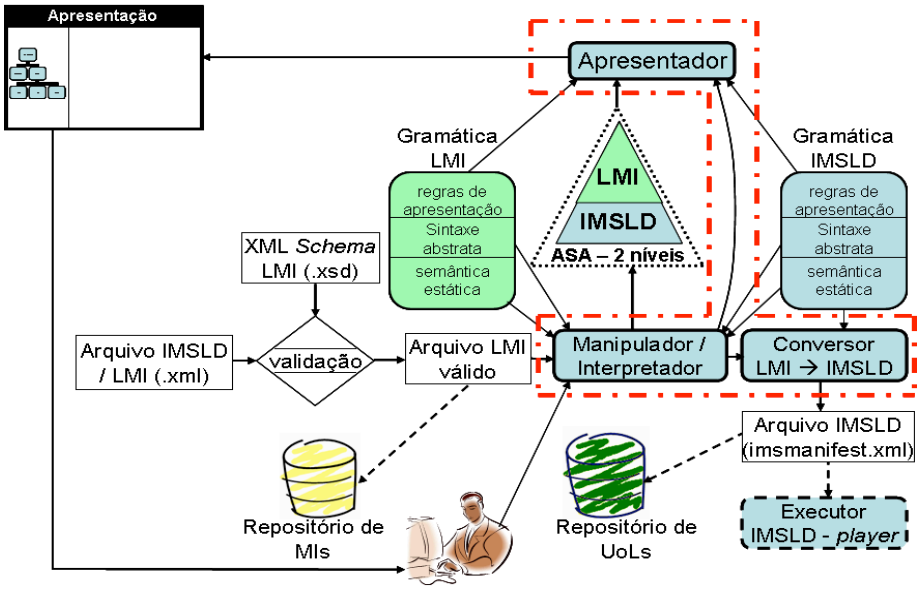

Figura 4 – Estrutura Interna de Editores IME e fluxo de arquivos

Um Editor IME é um Editor LD Específico Integrável e tem em sua estrutura interna uma evolução do editor LD genérico. O editor IME é composto por três elementos principais, a saber, Manipulador / Interpretador, Apresentador e Conversor LMI→IMSLD, representado pelo contorno tracejado na Figura 4.

Nos Editores IME, o Manipulador / Interpretador trabalha a partir de uma Árvore de Sintaxe Abstrata (ASA), sendo também responsável por manipulá-la em relação às modificações requeridas a partir da interface de usuário. A ASA possui dois

níveis, a saber, nível IMSLD e nível LMI. Para manipular essa ASA o Manipulador / Interpretador se apóia nas respectivas gramáticas de linguagem (IMSLD e LMI). Cada gramática de linguagem é composta de sintaxe abstrata (representada pelo código XML), semântica estática (representada pelo código XML dos arquivos de XML *Schema*) e regras de apresentação (associada à estrutura hierárquica de árvore do arquivo XML), de modo similar ao que acontece em um editor genérico de linguagens, conforme apresentado por Fernandes [1992].

A ASA de dois níveis apresentada pelo modelo IME tem a mesma forma do código-fonte das linguagens LD e LMI, pois ambas são representadas computacionalmente usando XML, que por sua vez, possui um mapeamento natural entre a ASA e o documento XML. O Apresentador também se beneficia do isomorfismo entre o código-fonte e a representação da ASA, para exibir ao professor um menu baseado em árvore. Essa característica resolve a limitação e. apontada na Seção 2, pois o professor, ao incluir outro MI não tem a necessidade de reorganizar o fluxo de atividades. Ao interpretar as regras de apresentação constantes na gramática IMSLD e LMI, o Apresentador pode personalizar a interface com o usuário.

Também por trabalhar com essa estrutura em árvore, o IME importa qualquer material estruturado no formato IMSLD, pois não necessita de arquivo adicional para controle de sua interface com o usuário, resolvendo a limitação d. da Seção 2. Por fim, o Conversor LMI→IMSLD, usado pelo Exportador, tem como função converter o código-fonte com construtos LMI para LD. Isso é necessário para que o material gerado pelos editores IME esteja disponível para uso em qualquer executor LD.

3.3 O Ambiente IME

O ambiente IME é um editor desenvolvido seguindo fielmente o modelo de mesmo nome. É uma aplicação *Web* (*vide* Figura 5) desenvolvida usando tecnologias atuais, tais como JAXB, servidor Apache Tomcat, HTML, JavaScript, jQuery e CSS.

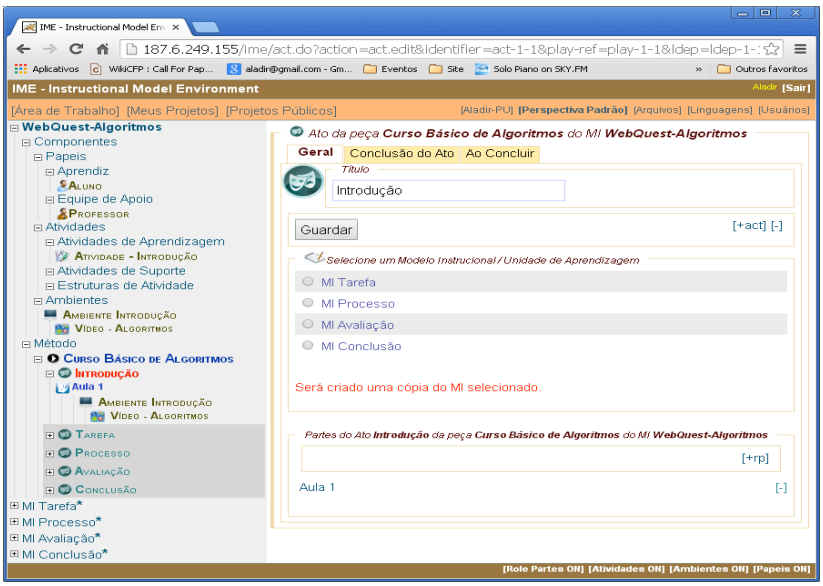

Figura 5 - Interface com o usuário do IME / Intercalação de MIs.

O IME apresenta uma interface amigável, constituída de uma tela com duas divisões, onde à esquerda é apresentado um menu em formato de árvore que, quando

clicado, apresenta formulários à direita de acordo com a opção de menu escolhida, conforme pode ser visto na Figura 5. Por exemplo, se o usuário clicar no lado esquerdo sobre o rótulo "Introdução", o formulário respectivo com a edição de atos aparecerá.

Suponha que se queira construir um MI *WebQuest*, como abordado anteriormente. Suponha também que já existam MIs para cada uma das fases da WebQuest. Neste caso, basta criar um MI Introdução (MI pai) e depois intercalar sequencialmente os outros MIs (MIs filhos), conforme mostrado na Figura 5.

Ao acessar esse mesmo MI gerado como um usuário do tipo professor-usuário, o usuário conseguirá apenas usar os editores IME para criarem UoLs e, neste modo, o professor terá acesso a um editor personalizado para aquele MI, como pode ser visto na Figura 6. Neste caso, o usuário estará acessando o mesmo MI da Figura 5, no entanto o IME exibe apenas o importante para o professor que quer agora estruturar seu material. É possível perceber também que na Figura 5 se tinha "Ato da Peça" como rótulo para a tela de atos e agora com o editor gerado se tem "Capítulo do Curso".

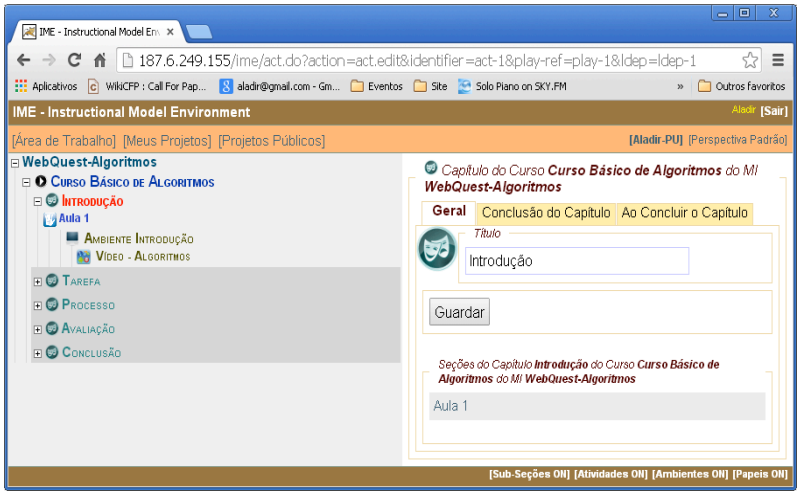

Figura 6 - Editor IME para um MI WebQuest.

3.4 Avaliação do Ambiente IME

Realizou-se uma bateria de testes exaustivos com intercalações de MIs tanto em extensão (um MI-pai com k MIs-filhos no mesmo nível) como em profundidade (um MI-pai com *k* MIs-filhos, *k* MIs-netos, seguindo até profundidade *n*). O material gerado nas simulações foi submetido a um executor (CopperCore Player) e executado com sucesso. Adicionalmente, realizou-se um estudo experimental com a participação de 18 professores/tecnologistas envolvendo a intercalação de MIs no Collage, no IME e no Open GLM Prolix. Neste estudo experimental, o IME superou os outros editores avaliados nos quesitos flexibilidade na intercalação de MIs, importação e exportação.

O modelo IME propõe uma abordagem baseada em geração de editores mais adequada para a criação de uma família de editores LD específicos. Por característica do modelo, grande parte da estrutura do software baseado neste é fixa e reusável, pois a geração se baseia na interpretação direta do código-fonte em LD. A diminuição de esforço no desenvolvimento de novos editores é acentuada, não sendo necessário manter uma equipe de desenvolvimento após se ter a estrutura fixa desenvolvida. A construção dos MIs pode ser feita por um ou mais professores, não havendo dependência da equipe

de desenvolvimento para criar editores LD específicos para os modelos instrucionais desejados, como é necessário nos outros editores LD da literatura. Isso é um ganho tanto no contexto educacional quanto no do desenvolvimento de softwares.

4 Conclusão

Este artigo apresentou um modelo, um gerador de editores e um editor que contorna problemas apresentados pelos atuais editores de LD, especialmente quanto ao apoio pedagógico aos usuários em relação aos modelos instrucionais. O IME permite a criação ilimitada de MIs. A intercalação entre eles também é ilimitada e feita sem a necessidade de ajustes manuais, o que resolve as principais limitações dos atuais editores LD.

Como pesquisa futura, pode-se se destacar o seguinte: (i) Uma pesquisa que foque na questão da usabilidade, talvez considerando uma possível representação em notação visual própria, que possa ser utilizada de modo associado na criação e intercalação de MIs; (ii) Verificar a possibilidade de aplicação do Modelo IME para outros padrões além do LD, como o SCORM, por exemplo; (iii) Ampliar a LMI para resolver outras limitações da LD ou até mesmo aplicá-la em outros contextos.

Referências Bibliográficas

Burgos, D. *What is wrong with the IMS Learning Design specification? Constraints And Recommendations*. 2008. Disponível em http://www.kde.cs.uni-kassel.de/conf/lwa10/papers/abis8.pdf.

Camargo, E. Z.; Fernandes, C. T. *WQE um Editor de WebQuests Versátil*. Anais do XXI Simpósio Brasileiro de Informática na Educação, 2010. João Pessoa-PB.

Dabbagh, N. *Pedagogical models for E-Learning: A theory-based design framework*. International Journal of Technology in Teaching and Learning, p. 25–44, 2005.

Dodge, B. *What is WebQuest?* Disponível em http://webquest.org/index.php. Acesso em 15/07/2014.

Griffiths, D.; Blat, J.; Garcia, R. et al. *Learning Design Tools*. Learning Design: A Handbook on Modelling and Delivering Networked Education and Training. p.109–135, 2010. Netherlands: Springer-Verlag Berlin / Heidelberg.

Hernández-Leo, D.; Asensio-Perez, J. I.; Dimitriadis, Y. *Computational Representation of Collaborative Learning Flow Patterns Using IMS Learning Design*. Educational Technology & Society, p. 75–89, 2005. Athabasca, Canada.

IMSCP. IMS *Content Packaging Information Model*. V. 1.2 Draft v2.0. Disponível em http://www.imsglobal.org/content/packaging/cpv1p2pd2/imscp_infov1p2pd2.html. Acesso em 15/07/14.

IMSLD. IMS *Global Learning Consortium - Learning Design Specification*. Disponível em: <http://www.imsglobal.org/learningdesign/>. Acesso em: 15/07/2014.

Koper, R.; Tattersall, C. *Learning Design*: A Handbook on Modelling and Delivering Networked Education and Training. 2010.

LAMS. *LAMS Project Website*. Disponível em: <http://www.lamsinternational.com/>. Acesso em: 18/07/2014.

OGP. *Open GLM Prolix*. Disponível em <http://sourceforge.net/projects/prolix-glm/>. Acesso em: 18/07/2014.

RELOAD. RELOAD - *Reusable eLearning Object Authoring & Delivery*. Disponível em: <http://www.reload.ac.uk/>. Acesso em: 18/07/2014.

Silva Júnior, A. F. DA; Fernandes, C. T. *Autoria em IMS Learning Design: uma proposta de classificação por níveis*. Anais do XXIII Simpósio Brasileiro de Informática na Educação, 2012. Rio de Janeiro - RJ.

Silva Júnior, A. F. DA; Silva, L. R; Fernandes, C. T. *Panorama dos Editores de Atividades de Aprendizagem em IMS Learning Design*. Anais do XXIV Simpósio Brasileiro de Informática na Educação, 2013. Campinas - SP.