Mapeamento de Sequenciamento e Navegação de Pacotes SCORM para o Componente Tarefas de Aprendizado do Modelo 4C/ID

Janaína Schwarzrock[1]
Adilson Vahldick[1]

[1] Universidade do Estado de Santa Catarina
Centro de Educação Superior do Alto Vale do Itajaí

janainaschwarzrock@gmail.com, adilson.vahldick@udesc.br

Resumo
A especificação SCORM foi definida para criar uma padronização na estrutura dos objetos de aprendizagem visando a portabilidade entre ambientes de aprendizagem. Essa especificação é focada no autoaprendizado. Para que um objeto de aprendizagem seja eficiente na condução do aprendizado autônomo é preciso levar em conta alguns aspectos no processo de aprendizagem. O design instrucional aparece como uma política para coordenar o desenvolvimento desses objetos visando maximizar o aprendizado. 4C/ID é um modelo de design instrucional estruturado em quatro componentes. O presente trabalho apresenta as configurações que um objeto empacotado em SCORM deve atender para que o principal componente do modelo 4C/ID seja aplicado.

Palavras-chaves: Modelo 4C/ID. SCORM. Tarefas de Aprendizado.

Abstract
SCORM was defined to create some learning object structure pattern, aiming to portability among learning environments. SCORM focus is on self learning. To make that a learning object will be efficient to conduct the self learning are necessary to accounting some learning process aspects. The instructional design comes on development coordination policies for these objects to maximize the learning. 4C/ID is an instructional design model structured on four components. This work presents the settings that a SCORM package should satisfy that the mainly 4C/ID component will be apply.

Keywords: 4C/ID Model. SCORM. Learning Tasks.

1. INTRODUÇÃO

A informática está cada vez mais presente na educação, podendo ser percebida com a difusão dos cursos on-line, à distância ou até mesmos nas trocas de materiais e trabalhos entre professores e aluno. A postagem e acesso a estes materiais acontece por meio de sistemas de gestão de aprendizado (*Learning Management System* - LMS), também conhecidos como ambientes de aprendizagem.

SCORM (*Sharable Content Object Reference Model*) é uma especificação para criação dos materiais educacionais que podem ser renderizados em navegadores web. Esta especificação tem o objetivo de promover o reuso dos materiais instrucionais digitais, de forma que um material produzido não seja para uso de um LMS específico,

permitindo que este possa ser utilizado em qualquer ambiente de aprendizagem que adote este padrão (ADL, 2012).

O *Four Component Instructional Design* (4C/ID) é um modelo de design instrucional desenvolvido por Van Merrienboer na década de 90 (Merrienboer et al., 2003). Este modelo vem em concordância com a teoria da carga cognitiva, a qual prega que a capacidade de processamento da mente humana é limitada. Com base nesta teoria, o modelo 4C/ID busca reduzir a carga cognitiva passada ao aluno durante o ensino, permitindo que este adquira com mais facilidade aprendizados complexos, os quais integram conhecimento, habilidade e atitudes (Merrienboer et al., 2003). Esse modelo define uma arquitetura de objeto de aprendizagem dividida em quatro componentes[1]: Tarefas de Aprendizado, Informações de Apoio, Informações Processuais e Práticas Parciais.

Assim como qualquer material instrucional, um material produzido e entregue com auxílio de tecnologia também precisa ser organizado e estruturado de forma a facilitar o aprendizado do aluno. Outro ponto importante é o momento certo a ser entregue o material de acordo com o nível do seu conhecimento (Vahldick et al., 2007), o que é definido por algum modelo de design instrucional. Com esta motivação, este artigo tem como objetivo apresentar a possibilidade de aplicação do modelo 4C/ID aos materiais instrucionais produzidos no padrão SCORM.

O SCORM foi escolhido pela sua portabilidade, e por sua documentação ser de livre acesso. O modelo 4C/ID foi selecionado pelo escopo bem claro de cada um de seus componentes assim como a relação entre eles. Através das regras de execução dos objetos SCORM vislumbrou-se a possibilidade de aplicar esse modelo.

As próximas seções estão organizadas da seguinte forma: as três primeiras são a introdução e revisões de literatura, a quarta relata a metodologia usada, a quinta e sexta descrevem o resultado do trabalho e a última seção apresenta as conclusões e trabalhos futuros. Na revisão da literatura são apresentadas informações relevantes para esse trabalho sobre a especificação SCORM e detalhes do modelo 4C/ID.

2. SCORM

Segundo ADL (2012), SCORM é um modelo de organização de objetos de aprendizagem que executam em um ambiente web. Esse modelo é composto de três partes[1]: Modelo de Agregação de Conteúdo (CAM), Ambiente de Execução (RTE) e Sequenciamento e Navegação (SN).

Um pacote SCORM é um arquivo compactado (no formato ZIP) contendo arquivos que serão utilizados na exibição das páginas web, e mais um arquivo (imsmanifest.xml) para definir a estrutura, relacionamento e sequência de apresentação dessas páginas.

O CAM determina como o conteúdo deve ser estruturado no arquivo imsmanifest.xml (ADL, 2009a). No CAM são descritas as relações e agrupamentos dos arquivos (*assets*), criando unidades lógicas (*Sharable Content Objects* - SCO). Um *asset* é um arquivo de qualquer tipo que possa ser usado para apresentação no navegador como, por exemplo, arquivos HTML, imagens, som ou arquivos Flash, ou Applets Java. O SCO representa as atividades de aprendizagem que serão lançadas ao aluno pelo LMS. O SCO agrupa os *assets* e forma uma unidade indivisível, ou seja, o SCO representa a granularidade mínima possível no SCORM.

[1] Tradução livre dos autores.

O RTE determina como deve ser o módulo no LMS para que possa acontecer a comunicação com o SCO. Além de definir requisitos de implementação, o RTE também define uma API em JavaScript para que o SCO realize essa comunicação (JCA Solutions, 2012).

Por exemplo, as funções JavaScript *Initialize* e *Terminate*, devem ser executadas pelo SCO para informar ao LMS que está iniciando ou terminando uma seção de aprendizagem (ADL, 2009b). Já as funções *GetValue* e o *SetValue* são utilizadas pelo SCO para solicitar e enviar dados a serem registrados pelo LMS.

O RTE também define um modelo de dados, que são as chaves para os dados que podem ser gravados e lidos a partir do LMS (Rustici, 2009a). Por exemplo, a chave *cmi.learner_name* contém o nome do aluno que está interagindo com o conteúdo (ADL, 2009b). O SCO pode executar a função *GetValue("cmi.learner_name")* e exibir na página *Seja bem vindo <nome>*.

O SN define as regras de sequenciamento e navegação para as atividades. O SN permite organizar o fluxo e *status* do curso como um todo (Rustici, 2009b), determinando quais atividades estarão disponíveis para o aluno e quais as opções de navegação, entre elas, são permitidas. Estas regras são estabelecidas no arquivo imsmanifest.xml (ADL, 2009c).

O SN consiste de um conjunto de definições de regras como, por exemplo, (i) determinar quantas vezes o aluno pode executar uma determinada unidade (SCO); (ii) quanto tempo ele pode alocar para execução da unidade; (iii) se o aluno pode retroceder ou avançar unidades sem concluir uma determinada unidade; e (iv) se pode avançar para a próxima unidade depois de conquistar certa pontuação.

3. MODELO 4C/ID

O 4C/ID é um modelo de design instrucional para aprendizagem complexa, as quais visam à integração de conhecimentos, habilidades, atitudes e a transferência deste conhecimento para vida real (Merrienboer et al, 2003). Pode-se dizer que o modelo 4C/ID é uma forma de estruturar o material instrucional, ou seja, uma forma de organizar o conteúdo e as suas atividades.

Teorias de ensino que não controlam a carga cognitiva repassada ao aluno podem prejudicar a aprendizagem devido à capacidade limitada de processamento da mente humana (Merrienboer et al., 2003). O modelo 4C/ID visa reduzir a carga cognitiva através da redução na quantidade de informação a ser lembrada para resolver um problema.

Segundo Merrienboer e Kester (2005), o modelo 4C/ID sustenta a existência de quatro componentes para que a aprendizagem complexa possa ser realizada: Tarefas de aprendizado; Informações de apoio; Informações processuais; e Práticas parciais. A Figura 1 apresenta a arquitetura do modelo com esses quatro componentes e seu relacionamento.

Nas **Tarefas de Aprendizado** estão inclusos os estudos de casos, projetos, problemas, entre outros. São tarefas completamente baseadas em experiências da vida real e visam a integração de habilidades, conhecimentos e atitudes (Merrienboer; Kirschner, 2008). Segundo os autores, estas tarefas devem ser organizadas em classes, do nível fácil ao difícil. As tarefas classificadas como fáceis são as que oferecem bastante apoio ao aluno, e a cada classe de tarefa este apoio é reduzido, até chegar às tarefas classificadas como difíceis (com pouco ou nenhum apoio).

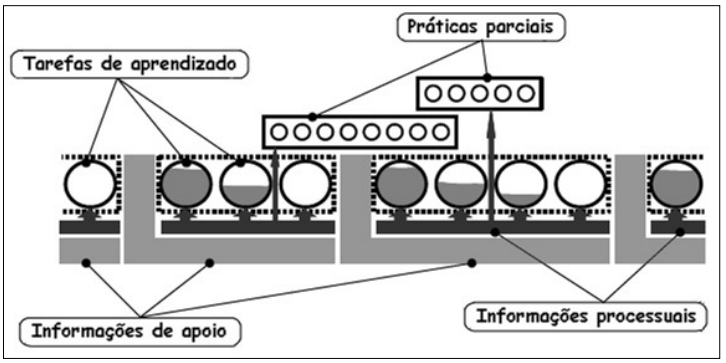

Figura 1 – Arquitetura do Modelo 4C/ID
Fonte: Adaptado de Merrienboer e Kester (2005)

As **Informações de Apoio** ajudam os alunos na execução de situações não-rotineiras das tarefas de aprendizado, as quais frequentemente envolvem raciocínio e resolução de problemas. São especificadas de acordo com cada classe de tarefa. Estas informações sempre devem estar disponíveis para o aluno, pois elas fazem a ligação entre o que o aluno já sabe e o que ele precisa saber para trabalhar nas tarefas de aprendizado (Merrienboer; Kirschner, 2008).

As **Informações Processuais**, de acordo com Merrienboer e Kirschner (2008), permitem que o aluno aprenda e realize as situações rotineiras das tarefas de aprendizado, e devem ser apresentadas ao aluno apenas quando necessário, para relembrar algum assunto. Estas informações fornecem um conjunto de regras e operações bem definidas e ordenadas de como executar estes aspectos rotineiros (Merrienboer; Kester, 2005).

As **Práticas Parciais** também se referem aos aspectos rotineiros das tarefas, mas aqui, em vez de informações, são oferecidos exercícios sobre estes aspectos, de forma a proporcionar um nível mais elevado de automação na execução das tarefas (Merrienboer; Kester, 2005). Estas práticas são necessárias apenas quando as Tarefas de aprendizado não fornecem uma quantidade de repetição suficiente.

4. METODOLOGIA

O objetivo do trabalho foi verificar se é possível aplicar o componente "Tarefas de Aprendizado", do modelo 4C/ID, em pacotes SCORM. Para mostrar essa aplicabilidade considerou-se a seguinte situação: "Existe, em um LMS, um material que possui um conjunto de tarefas sobre determinado assunto. O LMS seleciona aleatoriamente as tarefas e entrega ao aluno. Mesmo que a sequência de entrega é aleatória, os exercícios respeitam a ordem dos níveis de dificuldade das tarefas, sendo sempre do mais fácil ao mais difícil".

Para que a situação acima possa acontecer, o desenvolvimento dessas tarefas deve levar em conta a possibilidade da ordem aleatória de apresentação dos exercícios, e de acordo com o avanço do aluno os níveis de dificuldade devem evoluir. Um exercício deve ser implementado para que sua apresentação se adapte ao nível de dificuldade que se espera dele em um determinado instante.

Esta situação foi proposta pensando-se em uma futura automatização da aplicação do 4C/ID em materiais instrucionais por meio de uma ferramenta de autoria. Desta forma, com essa futura ferramenta espera-se aplicar o modelo 4C/ID de forma transparente ao *designer*. Assim, esse profissional desenvolverá os exercícios, e os níveis de dificuldade serão aplicados durante a execução do conteúdo no LMS.

Foram definidos os seguintes requisitos para atender a situação levantada:

REQ1: Existirão vários exercícios de forma que seja possível apresentar parcialmente as respostas de cada exercício;

REQ2: O nível de dificuldade do exercício é determinado pela quantidade de perguntas respondidas, onde o exercício mais fácil possui todas as perguntas respondidas, e no nível mais difícil não haverá nenhuma delas respondida;

REQ3: O sistema sorteará a ordem de apresentação dos exercícios, e como o nível de dificuldade é determinado por essa ordem, os exercícios devem ser criados sem nível definido, sendo este aplicado em tempo de execução.

Para encontrar uma solução para esses requisitos, foi necessário compreender as três partes do SCORM, e com base nisso, definir que elementos seriam usados. Foi utilizada a versão SCORM 2004 4th, e em cada módulo deveria ser definido:

CAM: como organizar os elementos para que fosse possível o sorteio dos exercícios;

RTE: que modelo de dados utilizar para determinar quantas respostas serão exibidas em determinado momento, ou seja, guardar o avanço no nível de dificuldade;

SN: em conjunto com o CAM, definir que regras usar para permitir o sorteio das atividades.

Após a definição de como mapear o 4C/ID com o modelo SCORM, desenvolveu-se um objeto de aprendizagem e utilizou-se este em um LMS compatível com SCORM 2004 4th.

5. PROPOSTA DE MAPEAMENTO

Conforme a lista de requisitos, os exercícios deveriam ser entregues ao aluno de maneira aleatória, e as respostas deveriam vir respondidas de acordo com o nível de dificuldade que o aluno se encontrasse: do mais fácil para o mais difícil.

O SCORM define as regras de sequenciamento em nível de SCO, e não de *asset*, ou seja, em nível de unidade e não de página. A primeira decisão tomada foi que cada tarefa de aprendizado representasse um SCO, e assim, uma tarefa de aprendizado pode ser composta de uma ou mais páginas, e todas as páginas correspondem ao mesmo nível de dificuldade, pois o nível é da tarefa e não da página.

As regras de SN selecionadas para atender o REQ3 foram o “Controle de Aleatoriedade” e o “Controle de Sequenciamento”. O “Controle de Aleatoriedade” possui duas configurações: se será usada ou não a aleatoriedade, e quantas vezes serão sorteadas (uma única vez, ou a cada vez que entra na atividade). Sobre o “Controle de Sequenciamento”, foi utilizada a opção para não permitir que o usuário selecione outra atividade enquanto está realizando alguma delas. Os LMS costumam mostrar no lado esquerdo da página uma estrutura de árvores contendo as unidades do objeto de aprendizagem. Desabilitando a saída de uma atividade, será possível obedecer que o aluno não “pule” níveis de dificuldade.

O Quadro 1 apresenta um trecho do arquivo imsmanifest.xml, ilustrando o mapeamento que será usado considerando três tarefas de aprendizado. O primeiro elemento XML representa o “Controle de Sequenciamento” e o segundo o “Controle de Aleatoriedade”.

```
<imsss:controlMode choice="true" choiceExit="false" flow="false"
     forwardOnly="false" useCurrentAttemptObjectiveInfo="true"
     useCurrentAttemptProgressInfo="true" />
...
<randomizationControls randomizationTiming="onEachNewAttempt"
     selectCount="3" reorderChildren="true" selectionTiming="once"/>
...
```

Quadro 1 – Trecho do mapeamento para sequenciamento e navegação

Para aplicar os níveis de dificuldade nos exercícios em tempo de execução é necessário basear-se na quantidade de tarefas que o aluno já realizou e na quantidade que deve realizar. Sabendo o valor destas duas variáveis é possível saber o nível a ser aplicado ao exercício atual. Para isto, foi necessária a utilização dos modelos de dados *adl.data._count*, *adl.data.n.id* e *adl.data.n.store* disponíveis a partir da 4ª edição do SCORM 2004. Esses modelos permitem criar uma área de memória compartilhada no servidor, e os SCOs podem acessar e armazenar dados nessa área. É importante salientar, que a especificação SCORM a todo instante lembra que cada SCO é considerado independente dos demais, ou seja, não existe troca direta de informações entre dois SCOs, e não existe a possibilidade de dentro de um SCO direcionar a navegação para outro SCO específico.

Os Quadros 2 e 3 apresentam os códigos JavaScript utilizados para obtenção e armazenamento de dados na área de memória compartilhada, utilizando-se dos métodos *GetValue* e *SetValue* da API do RTE juntamente com o modelo de dados. Essa área de memória compartilhada será usada para armazenar o valor da quantidade de tarefas que o aluno já realizou.

```
function GetDataMap(id) {
  var count = GetValue(“adl.data._count”);
  for (var i = 0; i < count; i++) {
     var resId = GetValue(“adl.data.”+i+“.id”);
     if (resId == id)
       return GetValue(“adl.data.”+i+“.store”);
}}
```

Quadro 2 – Obter um valor da área de memória compartilhada

```
function SetDataMap(id, store) {
  var count = GetValue(“adl.data._count”);
  for (var i = 0; i < count; i++) {
     var resId = GetValue(“adl.data.”+i+“.id”);
     if (resId == id) {
       SetValue(“adl.data.”+i+“.store”, store);
       return;
}}}
```

Quadro 3 – Armazenar um valor na área de memória compartilhada

Para que a “Tarefa de Aprendizado” consiga decidir a quantidade de campos preenchidos de cada nível, decidiu-se criar no início de cada página uma matriz contendo um percentual de acordo com a quantidade de tarefas já executadas pelo aluno, conforme exemplificado no Quadro 4, considerando que existirão três tarefas.

```
var aPreen = new Array();
aPreen[0] = 100.0; aPreen[1] = 50.0; aPreen[2] = 0.0;
```

Quadro 4 – Matriz para controle da quantidade de campos preenchidos

Uma tarefa ao ser iniciada faz uma busca, por meio da utilização da função *GetDataMap*, na área de memória compartilhada para verificar quantas tarefas o aluno já realizou, e com isto saber o nível de dificuldade em que se encontra. Uma vez conhecido o seu nível, a página consulta a matriz *aPreen* para saber o percentual de campos que deve apresentar já respondido ao aluno. A tarefa também é responsável por incrementar a quantidade de tarefas realizadas pelo aluno e atualizar este valor na área de memória compartilhada por meio da função *SetDataMap*.

A Figura 2 ilustra esta interação entre a tarefa e a variável, representada por *pagAtual*, que está localizada na área de memória compartilhada e contém a quantidade de tarefas realizadas. No início da execução (Figura 2.a), nenhuma tarefa foi lançada ao aluno, por tanto a variável *pagAtual* encontra-se nula. O LMS seleciona uma tarefa aleatoriamente e lança ao aluno. A tarefa selecionada (Figura 2.b) consulta a variável *pagAtual*, atualiza seu valor para zero, relaciona este valor com a matriz *aPreen* e, por último, preenche os campos necessários. Estas ações também são executadas pela segunda (Figura 2.c) e terceira (Figura 2.d) tarefas lançadas pelo LMS ao aluno.

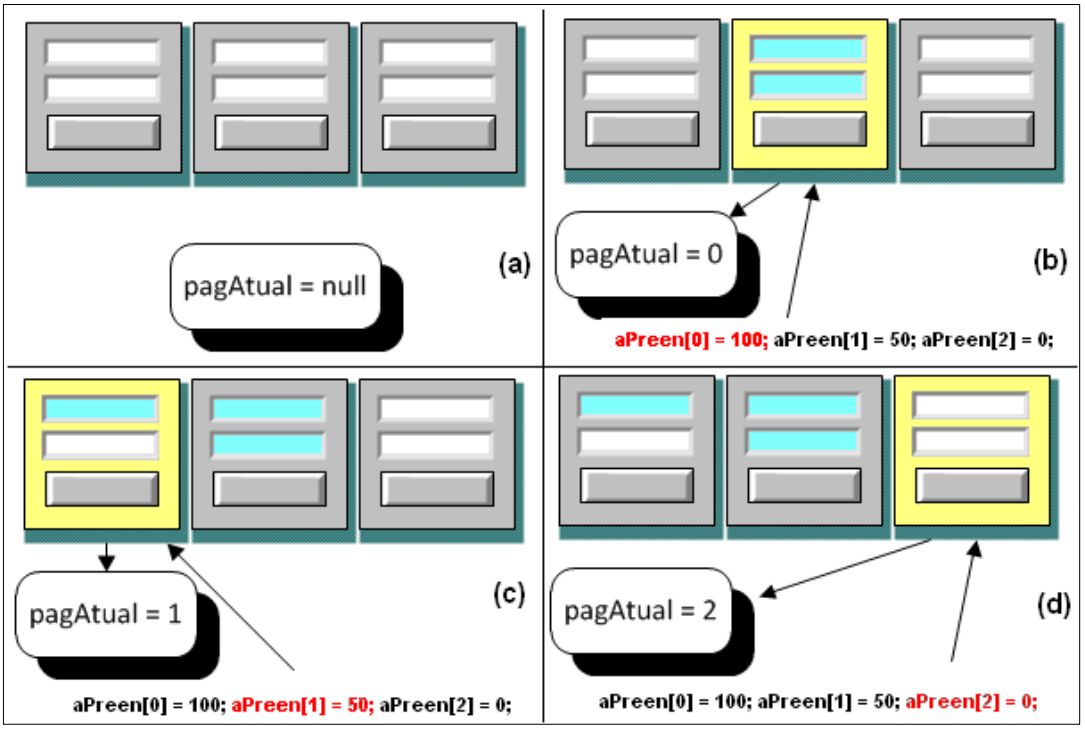

Figura 2 – Exemplo da relação entre a Tarefa e a área de memória compartilhada

Cada tarefa possui uma matriz de respostas. Esta matriz é utilizada pela tarefa tanto para o preenchimento dos campos, para adequar-se ao nível de dificuldade, quando para conferir se as respostas do aluno estão corretas. A matriz de respostas "resp" é ilustrada no Quadro 5.

```
  var resp = new Array();
  resp['campo1'] = '2'; resp['campo2'] = '3'; resp['campo3'] = '4';
...
<div id="campos">
   1 + 1 = <input type="text" id="campo1" size="5"/> <br/>
   1 + 2 = <input type="text" id="campo2" size="5"/> <br/>
   1 + 3 = <input type="text" id="campo3" size="5"/> <br/>
</div>
```

Quadro 5 – Definição das respostas de cada campo

A Figura 3 ilustra três páginas com as tarefas montadas para testar o mapeamento do modelo 4C/ID com SCORM. Pode-se observar que as páginas não possuem a mesma quantidade de campos entre elas. Todos os campos das tarefas criadas são do tipo digitação. Partes do seu código foram apresentados nos Quadros 2 a 5. Pode-se observar, além do botão [Validar] utilizado para conferência das respostas, os botões [Tarefa Anterior] e [Tarefa Posterior] que executam as ações de navegação, usando a função *SetValue* da API JavaScript do RTE, com o modelo de dados *adl.nav.request*, e os valores *previous* e *continue*, respectivamente. Caso o aluno retorne às tarefas anteriores, os campos já virão preenchidos com as respostas informadas anteriormente, não podendo mais realizar qualquer interação.

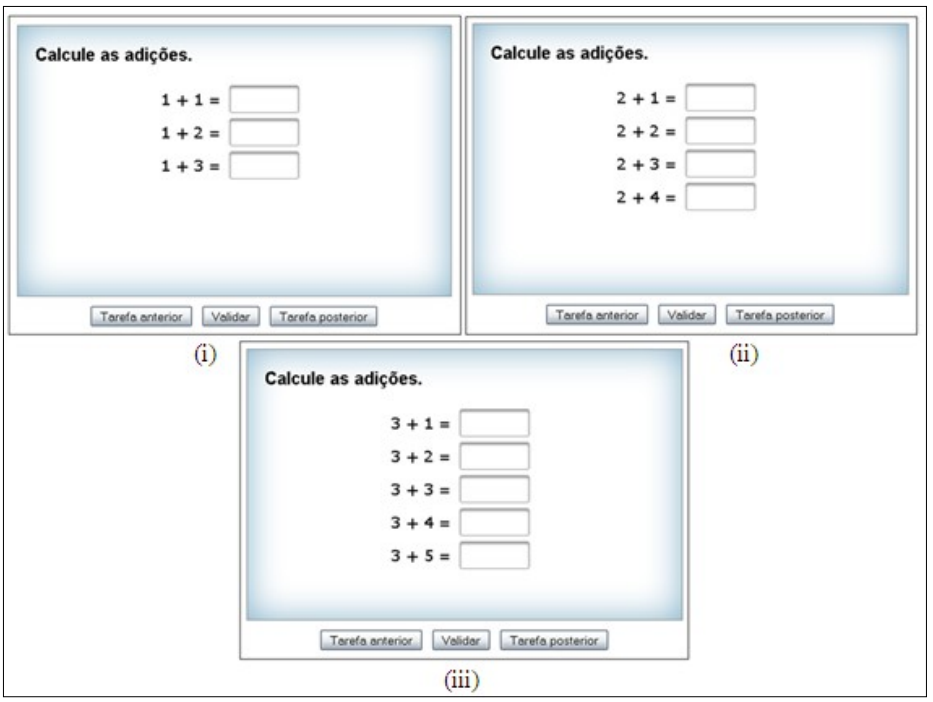

Figura 3 – Visualização das páginas com exercícios: (i) com três campos, (ii) com quatro campos e (iii) com cinco campos

6. EXECUÇÃO DO OBJETO GERADO

Para testar a execução do objeto, foi implantado o pacote no LMS BRUCE (Vahldick, 2008), o qual é compatível com a 4ª edição do SCORM 2004. BRUCE é um ambiente de gestão e execução de objetos de aprendizagem no formato SCORM, desenvolvido em uma dissertação de mestrado para validar um componente que executa e adapta objetos SCORM. A Figura 4 exibe a ordem (lembrando que é aleatória) de execução das três tarefas.

O BRUCE sorteou a ordem das tarefas, e nesse exemplo a tarefa com cinco campos foi a primeira, depois de quatro e por último três campos. A primeira página exibiu todos os campos preenchidos. A segunda página exibiu metade dos campos, e na terceira página todos estavam em branco.

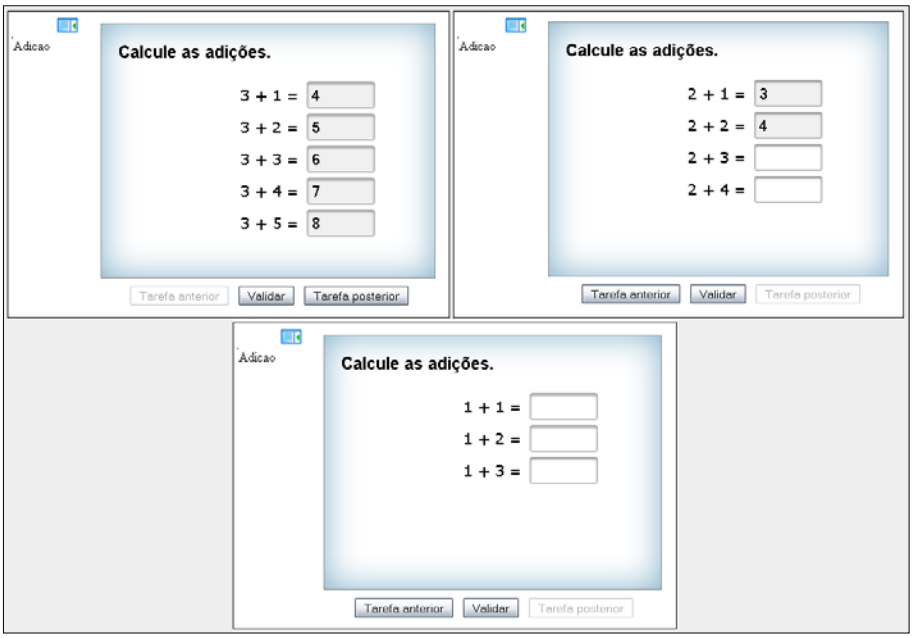

Figura 4 – Três tarefas sendo executadas por um LMS

7. CONSIDERAÇÕES FINAIS

O objetivo deste artigo foi propor o mapeamento dos recursos da especificação SCORM para que pudesse ser aplicado o componente "Tarefas de Aprendizado" do modelo 4C/ID.

No modelo SCORM foram utilizados os seguintes recursos:

CAM: cada tarefa de aprendizado deve ser um SCO (uma unidade);

RTE: foi utilizado os modelos de dados *adl.data._count*, *adl.data.n.id* e *adl.data.n.store*, disponíveis a partir da 4ª edição do SCORM 2004;

SN: aplicaram-se as regras Controle de Aleatoriedade e Controle de Sequenciamento.

A aplicação destas regras permite que o conteúdo seja entregue ao aluno de forma aleatória, além de controlar as requisições do aluno não permitindo que este entre em outra atividade sem ter terminado a atual. Com os modelos de dados citados do RTE, criou-se uma área de memória compartilhada entre os SCOs, permitindo que estes se localizassem quanto ao nível de dificuldade dos exercícios entregues ao aluno.

Além disso, criaram-se rotinas em JavaScript para que as páginas possam conhecer o seu nível de dificuldade, e preencher a quantidade de campos de acordo com seu nível.

Os testes com a execução do objeto mostraram a viabilidade do mapeamento proposto. O objeto continha somente três tarefas (uma página para cada tarefa), com quantidade de campos variados (todos os campos de digitação), mas a proposta é flexível para suportar qualquer quantidade de tarefas, bastando configurar o código apresentado nos Quadros 4 e 5.

Como trabalho futuro, espera-se desenvolver uma ferramenta de produção de objetos de aprendizagem, para que no momento do empacotamento, já se insiram as regras, rotinas e funções apresentadas nesta proposta, e assim naturalmente aplicar o modelo 4C/ID.

8. REFERÊNCIAS BIBLIOGRÁFICAS

Advanced Distributed Learning. [20--?]. Apresenta textos sobre SCORM. Disponível em: <http://www.adlnet.gov/capabilities/scorm#tab-projects>. Acesso em: 08 abr. 2012

Advanced Distributed Learning. **SCORM 2004 4th Edition: Content Aggregation Model**. 2009a. v. 1.1. 234p.

_____. **SCORM 2004 4th Edition: Run-Time Environment**. 2009b. Versão 1.1. 202p.

_____. **SCORM 2004 4th Edition: Sequencing and Navigation**.2009c. v. 1.1. 244p.

JCA SOLUTIONS. The SCORM Authority. [200-?]. Apresenta textos sobre SCORM. Disponível em: <http://www.scormsoft.com/scorm>. Acesso em: 08 abr. 2012.

MERRIENBOER, Jeroen J.G. Van; KESTER, Liesbeth. **The Four-Component Instructional Design Model: Multimedia Principles in Environments for Complex Learning**. New York: Cambridge University Press. 2005. 18p.

MERRIENBOER, Jeroen J.G. Van; KIRSCHNER, Paul A. Four Component Instructional Design (4C/ID). **SciTopics**. Set. 2008. Disponível em: <http://www.scitopics.com/Four_Component_Instructional_Design_4C_ID.html>. Acesso em: 08 abr. 2012.

MERRIENBOER, Jeroen J.G. Van; KIRSCHNER, Paul A.; KESTER, Liesbeth. **Taking the Load off a Learner's Mind: Instructional Design for Complex Learning**. Open University of the Netherlands - Educational Technology Expertise Center (OTEC), 2003. 30p.

RUSTICI, Mike. SCORM Run-Time Environment. **Rustici Software**, jan. 2009a. Technical SCORM. Disponível em: <http://scorm.com/scorm-explained/technical-scorm/run-time>. Acesso em: 02 abr. 2012.

_____. Sequencing and Navigation. **Rustici Software**, jan. 2009a. Technical SCORM. Disponível em: <http://scorm.com/scorm-explained/technical-scorm/sequencing>. Acesso em: 02 abr. 2012.

VAHLDICK, Adilson. **CELINE: um modelo para utilização e adaptação de conteúdo SCORM em ambientes inteligentes de aprendizagem**, no Estado de Santa Catarina. São José: UNIVALI. 2008. 139p. Dissertação.

VAHLDICK, Adilson; SANTIAGO, Rafael de; RAABE, André Luís Alice. Aplicação das Técnicas de Projeto Instrucional 4C/ID na Produção de Objetos de Aprendizagem em Conformidade com o SCORM Usando um Software Livre como Ferramenta de Autoria. In: **CICLO DE PALESTRAS SOBRE NOVAS TECNOLOGIAS NA EDUCAÇÃO**, X, 2007, Porto Alegre.