Um arcabouço para apoiar a generalização de conceitos através da pedagogia de representações externas

Diego Marczal[1], **Alexandre Direne**[1], **Gabriel Ramos**[1], **Andrey Pimemtel**[1], **Marcos Castilho**[1], **Fabiano Silva**[1], **Luis Bona**[1], **Marcos Sunye**[1], **Laura García**[1]

[1]C3SL – Departamento de Informática – Universidade Federal do Paraná (UPFR)
Caixa Postal 19.081 – 81.531-980 – Curitiba – PR – Brasil

{diego,alexd}@inf.ufpr.br

Abstract. *Initially, the paper presents the main aspects that justify the need for the design and implementation of a generic interactivity controller that can be applied and reused by a wide range os educational objects. The main objectives of the controller are detailed in terms of content access and the way it emphasises meta-cognitive aspects involved with typical learning tasks. The concept of abstraction though generalisation, as described by Ainsworth (2006), underpins the pedagogy associated with the learning process. At the end, the architecture and implementation of a framework called CARRIE (access Reflective and retroactive controller indexed by error) have their modules described and exemplified.*

Resumo. *Inicialmente, são apresentados os aspectos que justificam a necessidade de se projetar a implementar um controlador genérico de interatividade que pode ser aplicado e reutilizado por diversos objetos de aprendizagem. Os principais objetivos do controlador são detalhados em termos de acesso ao conteúdo e como ele enfatiza aspectos meta-cognitivos envolvidos nas tarefas típicas de aprendizagem. A abstração por meio da generalização, tal qual foi proposta por Ainsworth (2006), fudamenta a pedagogia associada ao processo de aprendizagem. Ao final, a arquitetura e a implementação do arcabouço denominado controlador de acesso reflexivo e retroativo indexado por erros (CARRIE) têm seus módulos principais descritos e exemplificados.*

1. Introdução

Atualmente, vários objetos de aprendizagem (OA) e simuladores têm sido desenvolvidos para apoiar tarefas de visualização de conceitos e solução de problemas em diversas áreas do conhecimento humano (*e.g.*, Matemática e Física). Por iniciativa coordenada de governo federal (MEC/SEED), boa quantidade desses objetos está disponível em bases de acesso público e gratuito. Esses OA possuem muitas características em comum, sendo que as principais estão relacionadas com a interface gráfica de acesso aos seus conteúdos educacionais. Todavia, a principal diferença entre eles está na ausência de um controlador genérico de interatividade que se aplique a todos os OA de maneira reutilizável e reconfigurável pelo autor do material de simulação e de solução de problemas.

Assim sendo, o papel do controlador não deve ser apenas o de acesso fácil ao conteúdo mas também o de tornar disponíveis os aspectos meta-cognitivos envolvidos

com as tarefas típicas de aprendizagem. Alguns autores [Puntambekar et al. 2003] caracterizam os aspectos meta-cognitivos como um conjunto de elementos da interface por meio dos quais é possível ao aprendiz realizar atividades reflexivas que demonstrem algum grau de consciência e controle sobre o seu processo de solução de problema. Exemplos de tais atividades são as de marcar, sublinhar, colecionar e a de desfazer (ou reverter) alguns passos já realizados em direção à referida solução. Todavia, com a invenção de técnicas cada vez mais modernas de interatividade dos OA, crescem também os desafios de criação e generalização das tarefas de natureza meta-cognitiva correspondentes. Tais desafios se manifestam de maneira particularmente acentuada quando o ambiente de execução inclui dispositivos manuais e de acesso móvel que permitem a aprendizagem em qualquer lugar, a qualquer hora [Sharples and Beale 2003]

Com isso, um exemplo de evento meta-cognitivo envolvido com o controlador é a possibilidade de o aprendiz retroceder ao contexto de erros cometidos no passado, uma vez que ele perceba em um dado instante a razão genérica por trás de sua incompreensão. De acordo com estudos recentes [Bull and Kay 2007], isso ajuda a promover condições para a reflexão sobre os passos adotados na solução de problemas e faz com que os mesmos caminhos de solução, ou novos caminhos, sejam tentados.

Porém, desenvolver software com esses potenciais não é um desafio simples, ainda mais quando eles são voltados à educação. Esse tipo de sistemas exige muita integração entre profissionais de diversas especialidades. Um exemplo de profissional é o programador, responsável pela implementação propriamente dita. Outro é o autor de conteúdo, que tem a tarefa de decidir a forma como esse conteúdo será apresentado.

Dificilmente um programador possui amplo domínio do conteúdo que será apresentado pelo sistema. De maneira semelhante, também não se deve esperar que o especialista do conteúdo saiba programar. Todavia, o programador, com seu conhecimento e experiência, pode propor um arcabouço genérico no qual os autores podem desenvolver seu conteúdo com pouco ou nenhum conhecimento de programação. Para tratar desse problema, surgiu a área de pesquisa de arcabouços (*shells*) de destinação pedagógica [Murray 1998].

2. Resenha literária

As linguagens de autoria podem ser classificadas como linguagens de programação peculiares. Elas são projetadas com o objetivo de proporcionar formas claras e de rápida assimilação para a construção de software educacional. Essas formas claras, na verdade, são Múltiplas Representações Externas (MRE) [Ainsworth 2006], conhecidas por elevarem o grau de abstração e de cobertura da descrição de conceitos de um domínio específico. Como tais MREs, as linguagens de autoria são muito mais compreensíveis a um autor de material eletrônico que, em geral, também é leigo em Ciência da Computação.

As linguagens de autoria mais típicas são as empregadas no desenvolvimento de sistemas educacionais baseados em estruturas de hipertexto ou hipermídia. Um representante clássico dessa categoria de software é o HyperTalk. O público-alvo do HyperTalk foi constituído por pessoas habitualmente chamadas de autores. Esses profissionais são capazes de identificar fragmentos simples em processos complexos (*e.g.*, didáticos) e de transformá-los em programas simplificados, chamados de “scripts”. Um script é semelhante a um texto escrito em língua natural mas possui estrutura lógica que lembra a de

linguagens da programação.

Criadas a partir de uma linguagem rica em MRE, as ferramentas de autoria são a implementação de compiladores ou interpretadores acoplados a ambientes de editoração. Em geral, tais ferramentas são utilizadas por meio de técnicas de programação visual em um WYSIWYG de tal maneira que o autor não precise usar nenhum tipo de programação por linhas de comandos. Em outras palavras, a ferramenta deve ajudar o autor a construir uma aplicação específica com recursos predominantemente gráficos e de manipulação direta dos objetos presentes no espaço de trabalho. Apesar de essa abordagem ser mais interessante do que escrever o código linha por linha, ela provoca o surgimento de muitas limitações à sua aplicação [Batista et al. 2003]. Uma delas, por exemplo, é que o usuário precisa se adaptar ao projeto do sistema, tendo em vista que várias decisões já foram tomadas pelo construtor do aplicativo de autoria.

De forma resumida, os sistemas de autoria traduzem escolhas do autor do conteúdo enquanto um usuário de linguagens de programação toma todas as decisões e gera o código linha por linha. Como exemplos de sistemas de autoria voltados à criação de STI (Sistemas Tutores Inteligentes), é possível citar os ambientes EON [Murray 1998], RUI [Direne 1997] e SIMQUEST [van Joolingen et al. 1997].

Alguns ambientes de autoria do passado, como o RUI, tiveram o objetivo de ensinar apenas conceitos de sub-domínios visuais especializados (*e.g.*, Radiologia médica). Devido a isso, eles implementam apenas os modelos de domínio e pedagógico da arquitetura funcionalista interna de um STI. Outros foram mais genéricos, tal como o EON, e implementaram todos os quatro modelos especialistas da referida arquitetura de STIs, incluindo os modelos do aprendiz e da interação. O SIMQUEST, por sua vez, foi destinado à criação de simuladores baseados em ambientes exploratórios de aprendizagem.

Apesar das facilidades que esses sistemas de autoria oferecem, eles apresentam extrema rigidez gramatical em certas situações de editoração. No ambiente RUI, por exemplo, há a necessidade do autor conhecer sentenças em Lógica de Predicados, o que significa uma dificuldade para leigos. Adicionalmente, pouco foi relatado sobre o funcionamento do mecanismo essencial da *shell* interpretadora do modelo de aprendiz do ambiente EON. Quanto ao SIMQUEST, nenhuma pesquisa evidenciou o potencial dos seus aspectos meta-cognitivos que afetam o controle que o usuário final (aprendiz) tem sobre o acesso aos conteúdos inseridos pelo autor das simulações.

O desenvolvimento de qualquer software é uma tarefa complexa e em se tratando de software educacional, ela se torna ainda mais difícil por ser multidisciplinar. Geralmente, essa dificuldade é acentuada na hora da criação dos aspectos interativos controlados pelo aprendiz. Se eles não forem bem projetados do ponto de vista pedagógico, podem fazer com que o software tenha pouco valor educacional [L. S. Fernandes and Benitti 2004]. Tudo isso acaba caracterizando problemas de baixa eficiência [Alves 2007] do ponto de vista de apoio à aprendizagem. Para que tais situações não aconteçam, deve existir equilíbrio entre atratividade e potencial educacional.

Em resumo, quando se constrói um software educacional (incluindo a interface), ele deve ser apoiado em alguma suposição de como as pessoas aprendem. Por isso, deve-se tentar desenvolver software como um recurso para promover o crescimento pessoal, não deixando-o apenas com a visão do projetista mas, principalmente, com a do aprendiz.

Assim sendo, a construção de interfaces voltadas à educação deve ter a preocupação de considerar, mesmo que simplificadamente, um perfil do aprendiz [O'Shea 1997].

3. Conceitos adotados na construção de AO

A *abstração* é o processo por meio do qual os aprendizes criam entidades mentais que servem como base para novos procedimentos e a formação de conceitos em um nível hierarquicamente mais alto de organização do conhecimento [Ainsworth 2006]. Nesse sentido, os aprendizes podem construir referências através das MREs para que a estrutura subjacente do domínio representado seja eventualmente exposta e tratada como um novo conhecimento, ainda mais genérico. Outro estudo do passado mostrou que a aprendizagem realizada com pares de representações externas leva a uma aquisição de conhecimento mais abstrato [Bull and Kay 2007].

Neste projeto de pesquisa e desenvolvimento, a construção de conhecimento aprofundado foi planejada para ocorrer por meio de abstração por generalização. A partir disso, as MREs selecionadas nos diversos software do projeto são sempre, pelo menos, em pares. Por exemplo, para o software de progressões geométricas (PG), elas são: (a) expressões analíticas organizadas passo-a-passo; (b) figuras da *geometria fractal*, as quais guardam correspondência com as várias versões de expressões do item anterior. É importante ressaltar que figuras de fractais não pertencem à geometria euclidiana clássica. As figuras de fractais possuem estágios de formação que podem ser chamados de passos. Em cada um desses passos, as alterações geométricas provocam crescimento ou decréscimo em medidas de tamanho do menor lado, perímetro, sub-áreas circunscritas e alguns outros aspectos com potencial para refletir regularidades matemáticas das propriedades de PG.

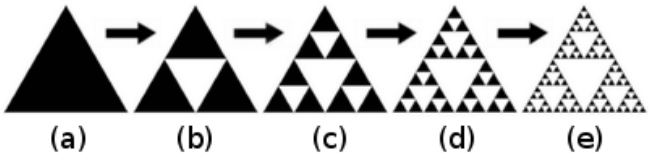

Figure 1. Triângulo de Sierpinsky

Como exemplo dos conceitos recém introduzidos, a Figura 1 apresenta o fractal denominado *Triângulo de Sierpinsky*, o qual está detalhado apenas em seus cinco primeiros passos de formação. O primeiro passo começa com um triângulo equilátero preenchido sobre uma área plana. No segundo passo, esse triângulo é dividido em quatro triângulos iguais por meio da ligação dos pontos médios de cada lado do passo anterior. Isso é então seguido da remoção da área do triângulo central. No passo seguinte, para cada triângulo gerado pelo passo anterior, repete-se o procedimento de divisão e remoção. Os passos de (a) até (e) da Figura 1 mostram as diferentes versões do referido fractal.

De maneira complementar, vale ainda dizer que a construção de um fractal não determina intrinsecamente um número fixo de passos. Devido a isso, esse número pode ser controlado por quem o constrói e, abstratamente, um aprendiz deve entender que a figura pode ter um número infinitos de passos. O que se espera disso é a extração de conceitos matemáticos abstratos sobre regularidades matemáticas por meio de processos indutivos da aprendizagem humana. De forma semelhante, relações complexas entre elementos visualmente simples (*e.g.*, um perímetro infinito que circunda a área finita do

PASSO		NÚMERO DE TRIÂNGULOS	TAMANHO DO LADO DO TRIÂNGULO (16)
	1	3	8
	2	9	4
	3	27	2
	4	81	1

Table 1. Passos de formação do Triângulo de Sierpinsky

fractal) podem ser formadas por processos dedutivos a partir de demonstrações baseadas em figuras da geometria euclideana clássica.

Com o uso de fractais como representação externa, pode-se estimular o aprendiz a integrar diversos conceitos matemáticos envolvidos na sua formação, dentre os quais podem ser citados: triângulos eqüiláteros, mediatriz, ponto médio de um segmento, perímetro, área, termo inicial e razão da progressão, além de muitos outros. Todavia, o aspecto mais importante para o desenvolvimento da capacidade de abstração está na maneira com que um aprendiz é levado a generalizar cada vez mais as expressões analíticas para que elas reflitam as grandezas de cada passo do fractal. Neste trabalho, argumenta-se que um software pode levar à boa prática do desenvolvimento da capacidade de abstração do aprendiz se este último for envolvido com o reconhecimento e a descrição analítica precisa de representações passo-a-passo, tais como as da formação de fractais.

Mais especificamente, o presente projeto investigou como a generalização por indução poderia ser atingida tomando-se os passos de formação de figuras fractais como sendo compostas de progressões geométricas. Por definição, uma PG é uma seqüência numérica em que cada termo, após o segundo, é igual ao produto do termo anterior por uma constante q. Por exemplo, a seqüência $(1, 2, 4, 8, 16, 32, \ldots)$ possui razão igual a 2. Com essa definição em mente, pode-se observar a Tabela 1, a qual apresenta quatro passos da formação do fractal Triângulo de Sierpinsky. Tanto a quantidade de triângulos formados nos diferentes passos como o tamanho do menor lado de cada triângulo formam PGs. Na verdade, mais elementos matemáticos desse fractal formam PGs ao longo das transformações entre os passos. O objetivo da Tabela 1 é mostrar parte do processo de indução do aprendiz por meio de uma aprendizagem passo-a-passo que pode ser precisamente guiada com a ajuda de um software.

Na referida tabela, é possível observar a formação de ao menos duas PGs: uma crescente e outra decrescente. A primeira está relacionada com a quantidade de triângulos e a segunda com o tamanho do lado do menor triângulo, ambas consideradas em cada passo. Observe que, se o tamanho do lado do triângulo inicial for definido com o valor genérico L, a divisão sucessiva dos lados em seus pontos médios leva gradualmente à sequência $(\frac{L}{2}, \frac{L}{4}, \frac{L}{8}, \frac{L}{16}, \ldots)$. Esse processo pode ser repetido infinitamente sendo que em cada passo, uma figura com triângulos cada vez menores será formada. Além disso, o importante mesmo do ponto de vista de generalização é que o aprendiz deverá descrever por si só o tamanho do lado no n-ésimo passo (passo de ordem n) como sendo $\frac{L}{2^n}$.

Essa representação passo-a-passo exige do aprendiz um desenvolvimento gradual de sua capacidade de abstração. Os conhecimentos matemáticos intrínsecos de um fractal fazem com que o aprendiz construa uma referência sobre os conceitos visualmente estruturados por bases analíticas. Isso parece tornar intuitivo o entendimento do assunto através do uso de MREs adequadas para serem manipuladas tanto por um software como por um humano, mantendo ainda a ligação direta com elementos concretos da rotina didática.

4. Arquitetura

Para consolidar os fundamentos da aprendizagem adotados até o momento no presente projeto de pesquisa e desenvolvimento, diversas ferramentas de software educacional estão sendo implementadas e validadas no ambientes escolar. A abordagem arquitetural de todas elas é baseada na existência de um núcleo comum de software, o qual é um arcabouço chamado aqui de Controlador de Acesso Reflexivo e Retroativo Indexado por Erros (CARRIE).

4.1. Organização funcionalista do CARRIE

Conforme representado na Figura 2, quatro módulos principais compõem a arquitetura funcionalista do CARRIE, sendo eles: Módulo de acesso ao conteúdo, Módulo indexador de erros, Módulo Facilitador e a Interface com o aprendiz. Todos esses módulos serão detalhados na sequência.

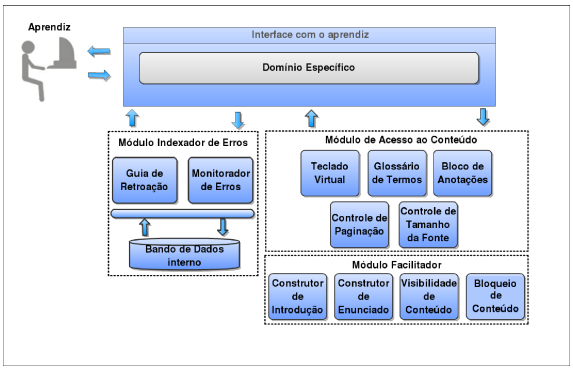

Figure 2. Arquitetura Funcionalista do CARRIE

Já foram implementados, completamente, dois OA que fazem uso do arcabouço CARRIE, sendo o primeiro para o domínio de Progressões Geométricas em Fractais e o segundo para o domínio de Funções de Primeiro Grau. Atualmente, diversos esforços têm sido realizados para a implementação de mais um OA que destina-se ao domínio da Matemática Financeira que também faz uso do CARRIE. Um total de quatro software educacionais farão parte da fase inicial de construção e aplicação de apoio computacional a atividades laboratoriais do ensino escolar de conceitos matemáticos do nível médio. A iniciativa se destina ao cumprimento de metas no âmbito do Projeto XXX [REF. OMITIDA PARA FINS DE AVALIAÇÃO].

Para a implementação do CARRIE, foi utilizada a plataforma Java seguindo as técnicas do paradigma de programação Orientada a Objetos. O software foi projetado para ser executado em qualquer navegador *Web*, independente de sistema operacional ou hardware. Além disso, o código é divulgado como código livre sobre licença GPL [LINK OMITIDO PARA FINS DE AVALIAÇÃO].

4.2. Módulo indexador de erros

Este módulo é composto por 2 partes principais, o monitorador de erros e o guia de retroação.

Monitorador de Erros: O monitorador de erros funciona com base em pontos de observação definidos pelo autor do conteúdo. Mais precisamente, o autor precisa necessariamente marcar, através da chamada de um método, onde o aprendiz comete um erro. Os parâmetros desse método são o título do erro, o texto explicativo do erro e uma variável booleana. Quando essa variável for verdadeira, uma janela *popup* será mostrada, alertando o aprendiz para o erro. A área visual do *popup* na tela tem como conteúdo o título e o texto explicativo passado como parâmetro para o método. Caso seja falsa, nada será mostrado.

No momento que esse método é chamado, o monitorador de erros recebe uma mensagem indicando um desvio conceitual cometido pelo aprendiz. Após o recebimento dessa mensagem, o monitorador salva o erro e o estado da aplicação em que o erro ocorreu em um banco de dados interno. Com essas informações salvas, o CARRIE estrutura o seu comportamento para que o aprendiz possa retroceder ao momento exato em que o erro foi cometido. Sendo assim, é apresentado ao aprendiz um guia de retroação, por meio do qual o aprendiz pode retroceder a qualquer erro cometido durante seu aprendizado.

Guia de Retroação: O guia de retroação permite que o aprendiz inspecione momentos exatos de erros passados e decida a quais deles vale a pena retroceder. Este guia faz a leitura do banco de dados interno de erros e cria um menu contendo um *link* para cada erro armazenado no banco de dados. Esse menu é mostrado ao aprendiz só depois de existir pelo menos um erro cometido. Quando o menu está visível, o aprendiz pode visualizar o título do erro e o seu texto explicativo. Passando-se o *mouse* sobre o *link* do erro, a qualquer momento pode-se retroceder ao erro cometido apenas clicando no *link*.

Quando o aprendiz retrocede ao erro, o CARRIE possibilita que ele refaça todo o exercício em que o erro foi cometido. Tal abertura de tarefas promove fundamentalmente o que é chamado de atividade reflexiva e permite que o aprendiz tente revisitar os aspectos que o levaram ao erro. Caso o aprendiz não queira refazer todo o exercício novamente (*e.g.*, por sentir que ainda tem dúvidas), ele também pode desistir da tentativa. Vale a pena ressaltar aqui que, ao contrário dos tradicionais recursos de desfazer simplesmente uma tarefa (*undo*, em inglês), o guia de retroação atinge um potencial meta-cognitivo muito mais adequado à finalidade pedagógica do CARRIE. Acredita-se que essa característica lhe confere um bom grau de originalidade em relação aos ambientes existentes.

4.3. Módulo de acesso ao conteúdo

É um conjunto de módulos responsáveis por apresentar o conteúdo ao aprendiz. Ele está dividido nos seguintes sub-módulos:

Glossário de termos: O CARRIE oferece um módulo para a criação do glossário de termos do domínio de uma maneira simples e clara. Quando o autor achar necessário que um glossário de termos seja disponibilizado ao aprendiz, basta definir um arquivo respeitando uma indentação pré-estabelecida. Quando o CARRIE identificar que um arquivo que segue essa estrutura foi criado, ele automaticamente disponibilizará a interface responsável pela apresentação do glossário por meio do *link* de acesso. Além disso,

o glossário ainda conta com uma busca por palavra-chave baseada no recurso de *auto-completar*.

Controle de tamanho de fonte: Um aspecto importante para os software atuais é a acessibilidade. Devido a isso, o CARRIE oferece um controle para que o aprendiz atue sobre o tamanho de letras de qualquer texto do OA.

Controle da paginação: O CARRIE tem como um dos objetivos fazer com que o aprendiz desenvolva seu conhecimento passo-a-passo para atingir a abstração por generalização. Para que isso ocorra, há um módulo que é responsável por oferecer paginação de todo e qualquer conteúdo a ser apresentado. Em outras palavras, o autor de material eletrônico não precisa se preocupar com esse tipo de código.

Teclado Virtual: Alguns OAs necessitam manipular uma entrada de dados do tipo numérico-analítica. Para isso, é necessário um estilo diferente de mecanismo de interação. Esse mecanismo é um teclado virtual, que é disponibilizado pelo CARRIE para o apendiz fornecer expressões analíticas tal qual essas são escritas em papel: em formato bidimensional.

Bloco de anotações: Um aspecto que pode ser considerado importante para o aprendizado é o bloco de anotações. Nele o aprendiz pode realizar anotações de todo o conteúdo que achar importante e, dessa forma, pode construir sua própria percepção do tema em foco. A partir disso, o CARRIE oferece um bloco de anotações ao aprendiz com recursos para destacar o texto em diversas cores.

4.4. Módulo Facilitador

Os sub-módulos contidos no módulo facilitador são destinados a tornar o desenvolvimento de um OA, com o uso do CARRIE, mais simples e com menos preocupação em codificação. Para isso, o CARRIE oferece três submódulos: (a) construtor de introdução; (b) construtor de enunciado; (c) bloqueio de conteúdo e visibilidade de conteúdo.

Os construtores de introdução e enunciado permitem o autor do conteúdo ampliar a formatação de imagens e textos através da codificação em html. Além disso, também permitem ao autor definir links de palavras, que quando clicadas abrem o glossário de termos com a sua respectiva definição em foco.

Por outro lado, certos conteúdos podem ser bloqueados ou escondidos pelo autor para que o aprendiz os acesse somente quando for apropriado. Isso oferece controle indireto do professor sobre os aprendizes espalhados no laboratório por meio de um módulo que garante que um exercício mais fácil é resolvido antes de outro substancialmente mais difícil.

4.5. Interface com o aprendiz

Uma interface com o aprendiz é pré-estabelecida pelo CARRIE e está dividida em três partes principais. A primeira é destinada ao título do OA, juntamente com o título da página atual. Em seu canto superior direito, também estão as opções para aumentar e diminuir o tamanho das letras. A segunda parte é destinada ao domínio específico. A outra é reservada às opções da dinâmica de controle oferecida ao aprendiz. Exemplos de tais controles são: paginar para frente ou para trás, botão de acesso ao glossário, botão de acesso ao guia de retroação e o botão de acesso ao bloco de anotações.

A Figura 3 mostra aspectos gráficos da interface em um certo instante da interação com o OA sobre Funções de Primeiro Grau definidas entre força e deslocamento de molas. No quadro estão ressaltados como as constantes elásticas das molas influenciam os gráficos das funções de primeiro grau de cada uma delas.

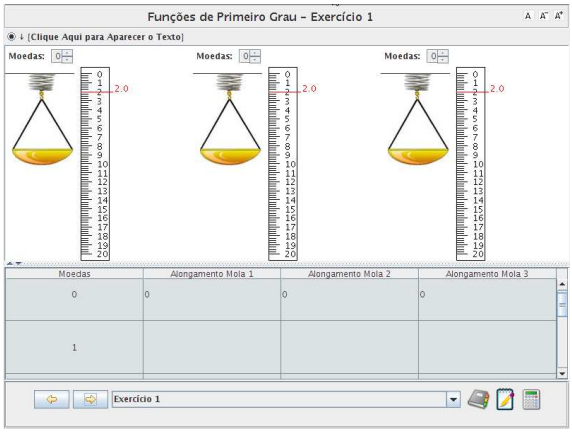

Figure 3. Interface do OA sobre funções de primeiro grau

5. Conclusão e trabalhos futuros

Destacou-se na resenha literária os aspectos que dizem respeito aos arcabouços de destinação pedagógica. Da mesma forma, as representações externas foram elucidadas como diretrizes fundamentais para o desenvolvimento de interfaces educacionais. Para demostrar o conceito de conhecimento aprofundado através da abstração por generalização, foi descrito o uso de fractais para o ensino de progressões geométricas e sua extensão para outros domínios por meio de uma arquitetura genérica.

Finalmente as perspectivas futuras apontam naturalmente para um maior aprofundamento das formas de retroação às situações onde o aprendiz cometeu erros. A ideia de registrar e restaurar os quadros de erro ainda é feita de maneira quase linear no arcabouço atual do controlador CARRIE. Adicionalmente, a iniciativa mais próxima de pesquisas do passado que ofereceram ao aprendiz uma visão de ambientes de aprendizagem com recursos para os usuários inspecionarem o que o software assumiu sobre eles foi chamada de modelos abertos de aprendizes (*open student models*) [Zapata-Rivera and Greer 2002]. Nessa categoria de sistemas tutores, o conceito de *skillometer* [Blessing 1997] como marcador de valor em uma escala de habilidade foi uma das formas com que os modelos abertos de aprendizes mais se projetaram no mundo de pesquisa.

No entanto, tais iniciativas não contemplaram a criação e o uso de linguagens de descrição da configuração com que os erros do aprendiz ocorreram para que tais erros pudessem ser revisados em outras condições no futuro. Essa fronteira de pesquisa continua totalmente inexplorada e se constitui em um campo relevante de interesse teórico e prático. Seus conteúdos cobrem desde aspectos epistemológicos da representação do conhecimento humano sobre os estados do mundo até detalhes de implementação que referenciam o chamado cálculo de situações no clássico mundo de blocos da Inteligência Artificial. Por si só, a abordagem integradora de tais conteúdos sob uma linguagem única representa um grande desafio de pesquisa cuja complexidade certamente contribuirá com conhecimentos originais para a Informática na Educação.

Agradecimentos. Os autores gostariam de agradecer ao financiamento do projeto CONDIGITAL, originário do convênio MEC/FNDE nº 825001/2007. Também se faz necessário reconhecer a importância de: (a) Jorge Bounassar pelo apoio operacional do LACTEC (www.lactec.org.br); (b) Elizabete dos Santos e Eziquiel Menta pela colaboração de produção do CETEPAR (www.diaadia.pr.gov.br/autec/); (c) Luciana Gastaldi, Lourdes Almeida e Márcia Cyrino pela orientação didática e pedagógica do material no Departamento de Matemática da UEL (www.mat.uel.br).

References

Ainsworth, S. (2006). Deft: A conceptual framework for considering learning with multiple representations. *Learning and Instruction*, 16(3):183–198.

Alves, G. (2007). Um estudo sobre o desenvolvimento da visualização geométrica com o uso do computador. In *Simp. Bras. de Informática na Educação*, pages 3–12.

Batista, L., Direne, A., Trindade, J., Gimenes, I., and Taha, O. (2003). Autoria de diretrizes pedagógicas destinadas ao treinamento das múltipla capacidades da perícia em conceitos visuais. In *Simp. Bras. de Informática na Educação*, pages 573–582.

Blessing, S. B. (1997). A programming by demonstration authoring tool for model-tracing tutors. *Int. Jrnl. of Artificial Intelligence in Education*, 8:233–261.

Bull, S. and Kay, J. (2007). Student models that invite the learner in: The SMILI open learner modelling framework. *International Journal of Artificial Intelligence in Education*, 17(2):89–120.

Direne, A. (1997). Designing intelligent systems for teaching visual concepts. *Int. Jrnl of Artificial Intelligence in Education*, (8):44–70.

L. S. Fernandes, A. L. A. R. and Benitti, F. B. V. (2004). Interface de software educacional: Desafios de design gráfico. *IV Congresso Brasileiro de Computação ¿ CBComp 2004, Informática na Educação*, (3):254–258.

Murray, T. (1998). Authoring knowledge based tutors: tools for content, instructional strategy, student model, and interface design. *Jrnl. of the Learning Sciences*, 7:5–64.

O'Shea, T. (1997). A typology for educational interfaces. In *CHI '97: CHI '97 extended abstracts on Human factors in computing systems*, pages 119–120. ACM.

Puntambekar, S., Stylianou, A., and Hübscher, R. (2003). How do students navigate and learn from nonlinear science texts: Can metanavigation support promote learning? *New Technologies And Their Applications in Education*, 1(4):674–684.

Sharples, M. and Beale, R. (2003). A technical review of mobile computational devices. *J. Comp. Assisted Learning*, 19(3):392–395.

van Joolingen, W., King, S., and Jong, T. (1997). The simquest authoring system for simulation-based discovery learning. In *World Conf. on Artificial Intelligence in Education*, pages 79–86.

Zapata-Rivera, D. and Greer, J. (2002). Exploring various guidance mechanisms to support interaction with inspectable learner models. In *Conf. on Intelligent Tutoring Systems*, pages 442–452.