Alguns princípios para situações de engenharia de sotwares educativos*

*Gilberto Lacerda Santos***

RESUMO

Este artigo trata das particularidades do processo de engenharia de softwares educativos, a partir da experiência de pesquisa e desenvolvimento acumulada pela equipe do Laboratório Ábaco, da Faculdade de Educação da Universidade de Brasília. Primeiramente, procura-se explicitar a problemática do desenvolvimento do software educativo. Em seguida, são apresentadas as metodologias e estratégias de engenharia de sistemas adotadas e adaptadas pelo laboratório: a modelagem da cooperação e a modelagem orientada a objetos. As conclusões dizem respeito às medidas a serem tomadas em toda iniciativa de concepção e desenvolvimento de softwares para ensino e aprendizagem.

Palavras-chave: Softwares educativos. Engenharia. Estratégias.

Sobre a problemática do desenvolvimento do software educativo

A descoberta de novas formas de ensinar e aprender por meio da informática é um desafio extremamente motivador, que implica e que demanda trabalhos de investigação voltados para a produção de meios e materiais específicos, na perspectiva do estabelecimento de uma nova relação com a aprendizagem (Carvalho, 2000). De fato, no meio escolar, o uso pedagógico do computador é apontado como fator que pode efetivamente contribuir para um avanço qualitativo dos processos de ensino e de aprendizagem, de modo que tanto as agências governamentais quanto

* Artigo recebido em 30/11/2008 e aprovado 13/05/2009.

** Professor da Faculdade de Educação da Universidade de Brasília (UnB). Ph.D em Educação, doutor em Sociologia, mestre em Tecnologias Educativas, engenheiro e analista de sistemas. E-mail: glacerda@unb.br

a iniciativa privada têm investido na adoção de programas de informatização do ensino e na produção de meios digitais e telemáticos para uso em educação. E o foco principal destas possibilidades é o software educativo. Portanto, é preciso que se invista no avanço de conhecimentos no campo da engenharia de programas para educação, a fim de que surja uma nova geração de sistemas educativos, mais integrados e relacionados com as possibilidades cognitivas humanas, de acordo com o que é anunciado por Maturana e Varela (1995).

A engenharia de softwares, enquanto disciplina, tem por objetivo a compreensão e o controle da complexidade inerente ao processo de desenvolvimento de softwares. Não se trata de um processo banal. De fato, desenvolver bons programas de computador é um desafio de peso para a indústria da informática. Tal complexidade é decorrente da necessidade de compreensão do problema que gera a necessidade do software educativo, da especificação do cenário demandante, da análise de possibilidades, da adequada arquitetura das informações disponíveis, do cumprimento de prazos e orçamentos, de levantamento claro e inequívoco de requisitos, de seleção de ferramentas computacionais adequadas e de recursos humanos qualificados. Em se tratando de softwares educativos, este processo de desenvolvimento tem que abraçar tanto o funcionamento do sistema propriamente dito quanto os mecanismos pedagógicos e didáticos que constituem a base de todo instrumento de ensino e de aprendizagem.

Um processo de desenvolvimento de softwares educativos tem especificidades que o distinguem bastante de um procedimento de desenvolvimento de aplicativos comerciais, bancários ou domésticos. O engenheiro de softwares educativos não tem diante de si um sistema fechado no qual usuários e proprietários de sistemas e subsistemas interagem entre si através de procedimentos pré-estabelecidos, previsíveis e perfeitamente traduzíveis em operações automáticas e informatizáveis. Pelo contrário, um sistema educativo, por mais simples que seja, traduz e delimita conhecimentos em processo dinâmico de comunicação e percepção. Consequentemente, o engenheiro de softwares educativos deve lidar com um conjunto de aspectos subjetivos que caracterizam o fenômeno educativo. Estes vão desde a consideração da natureza dos conhecimentos a serem veiculados e das estratégias mais adequadas para fazê-lo até a compreensão do próprio processo de ensino e das interações entre um

indivíduo em processo de aprendizagem e um saber de referência, por meio de um mediador informatizado (Santos, 2000).

Considerando o exposto, este trabalho tem por objetivo discutir algumas questões do processo de engenharia de softwares educativos, tendo como base a experiência acumulada nos últimos seis anos por meio dos trabalhos do laboratório de engenharia de softwares do Grupo de Pesquisas Interdisciplinares Sobre as Aplicações Pedagógicas das Tecnologias de Comunicação e Informação, da Faculdade de Educação da Universidade de Brasília (Ábaco). Primeiramente, procuraremos conceituar o processo de engenharia de softwares educativos, colocando em evidência suas especificidades, que tornam extremamente complexo o seu ciclo de vida. Em seguida, conceituar a estratégia global de modelagem adotada e adaptada pelo Grupo Ábaco, que envolvem, de um lado, a modelagem da cooperação e o estabelecimento de parâmetros ecológicos e econômicos nos sistemas baseados em conhecimentos e, de outro lado, estratégias de modelagem orientadas a objeto e sua aplicabilidade a softwares educativos. Por fim, elaboramos conclusões que enfatizam alguns princípios que devem nortear empreendimentos desta natureza.

A Engenharia de softwares educativos

A área da engenharia de softwares é bastante recente e acompanha a história da informática, desde o primeiro computador, o ENIAC, em 1946, com o desenvolvimento do seu software básico, que o fazia funcionar. No Brasil, essa história é mais recente ainda e tem início na década de sessenta, quando foram desenvolvidos os softwares básicos do *Zézinho*, o primeiro computador nacional, concebido no Instituto Tecnológico da Aeronáutica, seguido pelos do *Patinho Feio*, na Universidade de São Paulo (Moraes, 1996; 2000). Desde então, inúmeras técnicas e modelos de desenvolvimento do software vêm sendo importadas dos grandes centros produtores de tecnologia informática. No entanto, mesmo nesses centros, a preocupação com a engenharia de softwares para educação é uma área ainda embrionária. A própria aplicação da informática na educação é um campo de pesquisa e desenvolvimento de poucas incursões vitoriosas, tendo em vista justamente a complexidade do fenômeno educativo, a necessidade da promoção adequada do desenvolvimento de competências cognitivas, motoras e afetivas e a velocidade com que o ser humano trata informações, transformando-as em conhecimento.

Nessa dimensão mais ampla, que se refere ao uso do computador na educação, é importante sinalizar que inúmeras pesquisas revelam ter a informática potencial para dinamizar o processo de ensino e aprendizagem, tendo em vista que o software, educativo ou não, adequadamente empregado, estabelece relações qualitativamente diferentes com o erro, com o fracasso, com a autonomia dos atores da relação educativa e com o próprio conteúdo proposto para aprendizagem. Nessa perspectiva, é fundamental que o software educativo seja instrumento de liberação da relação educativa centralizada no professor e baseada na educação de massa, no ritmo uniforme, na avaliação quantitativa, no tempo controlado, da aprendizagem dicotomizada e ritmada, como nos *Tempos Modernos*, de Chaplin. É importante e crucial que o software educativo proporcione um deslocamento da ênfase do ensinar para o aprender, ceda espaço à aprendizagem por livre descoberta, à aprendizagem colaborativa e construtiva, realimentando e redimensionando a prática do professor e permitindo que a escola extrapole seus limites físicos para interagir com o que se passa fora dela (Santos, 1993; Iunes, 2001; Fagundes; Sato; Maçada, 2001).

Em nossos estudos, temos verificado que o surgimento do software educativo é simultâneo à disseminação do computador na sociedade. No final da década de 50 e início dos anos 60, ponto alto da tendência tecnicista na educação, que tinha suas bases na teoria behaviorista de Skinner e no modelo pedagógico de instrução programada, baseada no estímulo-resposta, o computador passa a representar uma possibilidade promissora e eficaz de padronização dos materiais instrucionais. Nasce assim a **computer aided instruction** (CAI) que teve ampla disseminação, principalmente nos EUA. Até o fim da década de 70, a CAI representa o tipo padrão de software educativo. O surgimento de novas tendências pedagógicas nos anos 70 e 80, agregado aos estudos sobre os processos de construção do conhecimento humano desenvolvidos pela Psicologia, provoca um questionamento amplo sobre o papel do professor e do aluno, agora percebidos como sujeitos produtores do conhecimento, condutores do processo ensino-aprendizagem. Teoricamente, o professor não é mais aquele que transmite, repassa conhecimentos e informações ao aluno, mas aquele que é capaz de criar um ambiente de aprendizagem que facilite e estimule o aluno a construir conhecimento, na perspectiva anunciada por Moretto (1999). Simultaneamente aos avanços na área educacional, acontecem grandes avanços na área computacional, tanto no que diz respeito

a hardware como a software. Os computadores encolhem fisicamente graças à miniaturização dos componentes, provocando uma produção em escala cada vez maior, acompanhada de uma redução no preço. Isso faz com que a disseminação antes iniciada chegue até as residências. Seguindo a tendência, os softwares educativos passam a incorporar novas concepções educativas e surge certa preocupação com os modelos computacionais próprios para o seu desenvolvimento. Desde logo, percebe-se que a engenharia de softwares educativos é um campo complexo, com características próprias, que demanda um conhecimento amplo por parte do projetista, principalmente quando são consideradas as características básicas de um sistema computacional para uso pedagógico.

O SOFTWARE EDUCATIVO

O software educativo é, primeiramente, um espaço para proporcionar a construção de conhecimentos. Nesse sentido, qualquer software pode ser considerado educativo, como um software aplicativo (um tratamento de textos ou uma planilha de cálculos), um software lúdico (um jogo, um simulador) ou um software de autoria (uma meta-linguagem de programação). Até um software básico (como o Windows ou o DOS ou, ainda, o Linux pode ser utilizado com finalidades educativas. Entretanto, o software educativo propriamente dito é aquele desenvolvido com a finalidades educativas explícitas demandando, para subsidiar sua produção, procedimentos específicos, relacionados a um conhecimento aprofundado dos processos cognitivos humanos, seja ele de natureza lúdica (um jogo educativo) ou de conteúdo escolar (um software para o ensino de Química, por exemplo), seja ele estático (em cd-rom) ou distribuído (para a Internet). Aliás, tendo em vista a evolução constante do conhecimento, a necessidade de dinamismo nos processos interativos e a impressionante expansão da Internet (MORAES, 2001; JOHNSON, 2001), há uma forte tendência à concentração de esforços no desenvolvimento de softwares educativos distribuídos, que podem evoluir conforme a evolução de necessidades de formação e de aprendizagem e ser alterados conforme demandas dos usuários, avanços no conhecimento, redimensionamento de aspectos avaliativos etc. De qualquer forma, e conforme indica a experiência acumulada do Grupo Ábaco, o software educativo, concebido em uma perspectiva pedagogicamente interessante, precisaria ser desenvolvido a partir da consideração de critérios tais como:

a) objetivos educacionais consistentes e condizentes com o contexto mais amplo, com a organização do trabalho pedagógico do professor, com a missão da escola e com as demandas da sociedade;

b) adequabilidade curricular para subsidiar tanto a construção de conhecimentos formais propostos pela escola quanto a valorização dos conhecimentos produzidos pelo próprio aluno;

c) possibilidade de integração de diferentes linguagens de comunicação;

d) valorização e potencialização do aluno e das múltiplas inteligências do indivíduo;

e) integração entre, de um lado, interfaces psicológica e ergonomicamente viáveis e, de outro, conteúdos pedagógica e socialmente coerentes;

f) possibilidade de subsidiar trabalhos colaborativos;

g) possibilidade de apoiar e de integrar o trabalho do professor, sem funcionar como camisa de força que impeça o uso criativo do software e limitando suas possibilidades;

h) grau de complexidade que permita a manifestação da singularidade do processo de aprendizagem, que varia de indivíduo para indivíduo;

i) integração de modelos de avaliação condizentes com abordagens educativas mais flexíveis e menos quantitativas;

j) interação do aluno com o software, fator importante para facilitar a aprendizagem, exigindo posicionamentos conceituais capazes de provocar conflitos cognitivos, remetendo a reflexões, questionando saberes já instituídos, retirando o aluno da passividade.

Evidentemente, não se trata aqui de pleitear o desenvolvimento do software educativo perfeito, que abarque todos estes princípios, além de muitos outros que poderiam ter sido elencados; trata-se, sim, de explicitar que há conhecimentos específicos que precisam ser considerados em todas as fases do ciclo de vida do software educativo que, conforme já mencionado, diferenciam seu processo de concepção do de um software convencional. E é justamente com o objetivo de contribuir para o avanço da discussão sobre modelos computacionais para a engenharia

de softwares educativos que abordamos, em seguida, à luz da experiência acumulada em três projetos de desenvolvimento de sistemas para educação, algumas considerações sobre a estratégia da modelagem da cooperação, associada à modelagem baseada em objetos aplicada ao processo do software educativo.

A modelagem da cooperação

O ponto de partida para o desenvolvimento do software educativo deve ser a compreensão exata da situação de aprendizagem para a qual o produto se destina e da natureza do conteúdo a ser tratado bem como do público-alvo. Nesse sentido, é necessário que seja feita uma análise de requisitos computacionais e, sobretudo, educacionais, que permita ao projeto tomar forma e cumprir seus objetivos, tanto lógicos (de programação) quanto didáticos (de educação). Em algumas experiências recentes da equipe do Laboratório Ábaco da Faculdade de Educação da Universidade de Brasília, voltadas ao desenvolvimento dos softwares educativos, *O Dado de Contos* (Santos, 1998), *Hércules e Jiló* (Souza e Santos, 2001) e *Hércules e Jiló no Mundo da Matemática* (em desenvolvimento), os três destinados a apoiar o trabalho pedagógico em fase de início de escolarização, foi empregada uma técnica de análise de requisitos chamada *Modelagem da Cooperação* (Breuer e Greef, 1993). Essa técnica é capaz de explicitar, ao mesmo tempo, os requisitos computacionais e os requisitos educacionais de sistemas com finalidades educativas.

A modelagem da cooperação serve fundamentalmente para delimitar os procedimentos necessários para que o software educativo seja operacional, identificando seus atores, o relacionamento entre eles, seus papéis e os produtos a serem gerados pelo software, traduzíveis na forma de conhecimentos, habilidades e atitudes, quando for o caso. Trata-se de um procedimento que tem por objetivo traduzir o funcionamento do sistema visado sob a forma de uma rede hierárquica de tarefas a serem executadas pelos diferentes atores implicados. Essa rede hierárquica de ações e de comportamentos deve cobrir todas as dimensões do sistema e a totalidade das interações entre este último e seus usuários, permitindo a obtenção de uma espécie de maquete ou visão de conjunto, suscetível de orientar o procedimento de desenvolvimento do software propriamente dito. A modelagem da cooperação é um instrumento para a arquitetura da informação, das interfaces, das interações e do conhecimento a ser

trabalhado no âmbito do software. Seu produto é ponto de partida para a modelagem do sistema, atividade que abordamos por meio da orientação a objetos, como veremos mais adiante. Em alguns casos, pode-se inclusive pensar na inclusão de conhecimentos a serem trabalhados fora do âmbito do software, mas gerados e fomentados pela intervenção pedagógica que teve o software como apoio principal.

A modelagem da cooperação compreende três etapas sucessivas e complementares: a decomposição de tarefas, a interrelação das tarefas e sua distribuição entre o sistema e seus usuários (fig. 1). Tais etapas permitem a chamada "modelagem das tarefas" que, por sua vez, conduz à modelagem do conhecimento e à modelagem da interface.

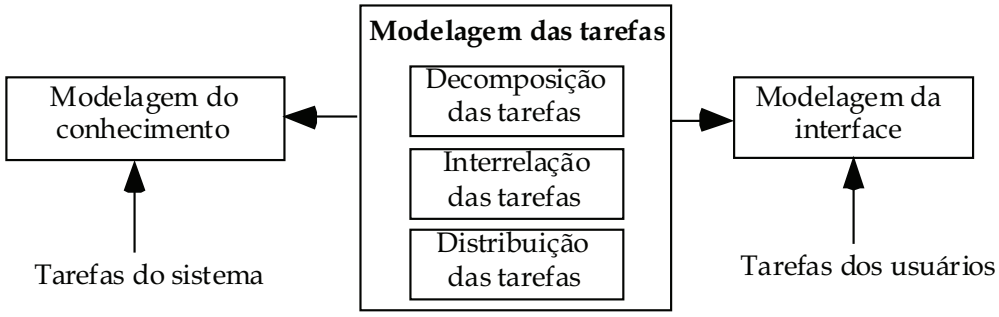

Fig. 1: A modelagem da cooperação (Adaptado de Breuer e Greef, op. cit., p. 50)

A decomposição das tarefas consiste na identificação dos diferentes sistemas e subsistemas que caracterizam o funcionamento do software educativo e em sua organização hierárquica. Por exemplo, no caso de *O Dado de Contos*, o próprio software foi percebido como um sistema em interação com o sistema "aluno", mediado pelas intervenções do sistema "professor". Esse tipo de representação fornece diversas informações fundamentais para o prosseguimento do processo de engenharia do software. Indica ao projetista a natureza das interações que darão suporte ao processo de ensino-aprendizagem, a distribuição de tarefas entre os atores implicados no processo (professor e aluno) e, também, a dimensão da ação do software no gerenciamento dos conhecimentos em torno dos quais as interações ocorrerão.

A interdependência das tarefas consiste em se colocar em evidência as relações de dependência entre certas tarefas e entre os diferentes processos implicados no desenvolvimento das interações. Essa etapa serve também para indicar as tarefas que se repetem e cuja rotina é

semelhante, na medida em que ensinar é participar de um processo em parte previsível e em parte aleatório, imprevisível (Moran; Masetto; Behrens, 2000). Nos softwares educativos mencionados, muitas tarefas não eram "informatizadas", mas induzidas a serem realizadas fora do computador. Esta opção teve como parâmetro as características da clientela visada pelos softwares educativos e toda uma concepção de uso da informática nas séries iniciais do ensino fundamental, em que há uma forte necessidade de apoio à socialização, à descoberta e à livre expressão da criança. A decomposição das tarefas previstas, computacionais ou não, permitiu, nas experiências mencionadas, uma delimitação exata do escopo do trabalho de desenvolvimento e de concepção lógica e educativa que teria lugar a partir da elaboração do projeto do sistema. No caso dos softwares educativos em foco, a lógica foi sempre a mesma: quanto maior o conhecimento das tarefas informatizáveis e das humanas, mais consistente seria o projeto do sistema, a cartografia do software educativo, que permite que se localizem os pontos mais importantes sem perder de vista as relações entre eles (Franco, 1997). E, no caso de todos os sistemas mencionados, o espaço dedicado à ação humana foi grande. Além do mais, em situação de aprendizagem, a maior parte das ações humanas são imprevisíveis, tendo em vista a própria maneira como diferentes pessoas reagem a um mesmo conteúdo pedagógico, os diferentes estilos de aprendizagem e a experiência de vida de cada um. Considerando o exposto, a decomposição de tarefas é essencial para que se compreendam com exatidão o papel e a dinâmica do software educativo almejado, visando-se articular melhor a interação entre o subjetivo (a atividade do aluno e do professor) e o objetivo (a atividade do software), conforme sugerem Button et al. (1998).

Finalmente, a distribuição das tarefas consiste na identificação dos atores implicados no desenvolvimento de cada tarefa. No que diz respeito ao software educativo, estamos referindo-nos ao aluno, ao professor e ao conhecimento articulado entre ambos. Trata-se de uma etapa que permite colocar em evidência as tarefas próprias ao software educativo (o modelo do conhecimento), as tarefas próprias aos seus usuários (o modelo da interface) e a articulação que será dada ao conhecimento (limites e abrangência). Essa etapa permite a obtenção de uma aproximação das relações interativas entre o sistema e seus usuários e o avanço do processo de análise em direção a um procedimento tradicional de análise

do sistema, seja por meio de uma estratégia estruturada, seja por meio de uma estratégia orientada a objetos.

As repercussões da modelagem da cooperação sobre a ecologia e a economia de um software educativo

A principal vantagem inerente ao processo de modelagem da cooperação está na obtenção de um recorte funcional rigoroso do software educativo visado. Em outras palavras, quando o engenheiro de softwares educativos, durante o processo de compreensão de um processo pedagógico a ser informatizado, consegue explorar e evidenciar toda a estrutura interativa a ser implementada, ele obtém uma espécie de modelo do sistema. Tal modelo permite e apoia a tomada de decisões com relação à própria organização futura do sistema (sua ecologia) e, também, com relação à quantidade de processos e de subprocessos que dele farão parte (sua economia). Em outras palavras, quanto mais sinergética for a relação entre os subsistemas que delimitam o funcionamento do software (subsistemas humanos e computacionais), mais ecológico ele será. Por outro lado, sua economia será medida pela racionalidade de seus procedimentos e pela efetividade de seu funcionamento. A modelagem da cooperação é, na verdade, um processo de metacognição que, no caso de softwares educativos, deve anteceder os procedimentos de análise computacional. Através desse processo de metacognição, o engenheiro de softwares educativos distancia-se de seu objeto de pesquisa para analisá-lo do exterior, incluindo em sua análise não somente as ações automatizáveis como também as que não o são. Desse modo, obtém-se uma maquete da interface entre o sistema e seus usuários, definida a partir de uma compreensão exata do papel atribuído ao sistema e do atribuído ao aluno bem como do atribuído ao formador e, principalmente, da identificação da interdependência entre tais papéis, da qual depende o próprio sucesso ou o fracasso das interações. Por outro lado, essa separação de papéis informa o analista sobre o tipo de conhecimento que será gerenciado pelo software educativo e. sobretudo, sobre os limites da ação deste último como instrumento de ensino e como material didático dinâmico.

Trabalhos considerados referência na área do desenvolvimento de sistemas tutores, como o de Wenger (1987), deixam claro que um dos principais problemas nesse campo de pesquisa está, justamente, no fato de que não há ainda metodologias de engenharia de softwares que permitam

uma interação efetiva com a maleabilidade de uma situação de ensino-aprendizagem, caracterizada por processos cognitivos complexos, por reações imprevisíveis e particulares a cada indivíduo em aprendizagem assim como pelo tratamento de informações subjetivas de comunicação de conhecimentos (Santos, 1993). A estratégia da modelização da cooperação entre o sistema tutor e seus usuários pode contribuir para o avanço dessa questão na medida em que permite um verdadeiro mergulho na situação pedagógica, favorecendo a compreensão de seus objetivos, de seu funcionamento e de seus componentes.

A modelagem da cooperação serve, como já foi explicitado, de ponto de partida para a modelagem do sistema, etapa que pode, por exemplo, fazer uso de uma abordagem orientada a objetos, tal como foi no caso dos três softwares educativos mencionados.

Objetos dão o tom da integração no projeto do software educativo

A modelagem baseada em objetos, proposta por Rumbaugh et al. (1993), é uma poderosa ferramenta da engenharia de sistemas e tem-se mostrado igualmente poderosa no que diz respeito a sistemas educativos, justamente por causa de sua maleabilidade. Trata-se de um instrumento que permite a descrição dos sistemas do mundo real, associando-os a comportamentos e a informações, aspectos fundamentais para se compreender, de maneira mais aproximada, o funcionamento de um software educativo, cuja concepção deve extrapolar o objeto tecnológico propriamente dito e englobar o ambiente educativo em que este será empregado. Segundo Coad e Yourdon (1992), o ser humano emprega normalmente três métodos para analisar o mundo real: a *diferenciação*, quando identifica os diferentes objetos do mundo real; a *distinção* entre o todo e suas partes, quando decompõe um objeto em suas partes componentes, e a *classificação*, quando agrupa objetos semelhantes segundo algum aspecto comum. A análise baseada em objetos utiliza-se, para desenvolver seus modelos, desses mesmos paradigmas. Estes poderiam também ser vistos como paradigmas de uma abordagem de aprendizagem tornando-se interessante instrumento para a modelagem de softwares educativos suscetíveis de provocar reações insondáveis em seus usuários interagindo em torno de conhecimentos, em processo de incremento de sua capacidade cognitiva. Vejamos os principais princípios da análise

baseada em objetos, em uma perspectiva de adaptação de sua compreensão às situações de projeto de softwares educativos:

A *Abstração* é o principio segundo o qual se ignoram os aspectos de um assunto não relevantes a um determinado propósito. A *abstração* é importante para entender um problema quando é impossível considerar-se toda sua complexidade. No caso do projeto de um software educativo, quando a representação pode facilmente escorregar para a redução da complexidade do processo de aprendizagem, a abstração é um componente importante do projetista. Mas, é preciso que seja uma abstração orientada pela realidade, o que implica na necessária observação de situações de ensino e aprendizagem, a fim de se captar sua complexidade e a fim de se entenderem melhor suas facetas. Tal procedimento foi uma baliza importante nos quatro projetos de sistema mencionados, especialmente no caso de *O Dado de Contos* e de *Hércules e Jiló*. É importante enfatizar que este último foi concebido para ser empregado junto a crianças com deficiência mental, o que demandou uma interação extremamente efetiva junto ao público-alvo, antes de se avançar para a etapa do projeto propriamente dita.

A *Identidade* garante que todos os objetos sejam distinguíveis através de um identificador. Apesar de dois objetos poderem possuir ambos os mesmos atributos, o princípio da identidade permite distingui-los. No projeto de software educativo, deve ser discutido com proprietários de sistema, docentes e alunos, é fundamental que a representação gráfica do sistema permita uma identificação e uma auto-identificação imediatas e inequívocas. Por ocasião do procedimento de desenvolvimento de *O Dado de Contos*, software educativo bastante complexo tendo em vista sua abordagem lúdico-educativa e por tratar-se de um jogo a ser jogado. tanto no computador quanto fora dele, o projeto do sistema tornou-se, gradativamente, muito complexo. Após a modelagem da cooperação e a identificação das tarefas dos atores envolvidos, a representação gráfica do projeto do sistema, respeitando princípios de identidade da representação orientada a objetos, foi fundamental para que todos se localizassem e se reconhecessem no projeto. É importante explicitar que, em um projeto de software educativo, os objetos representados não são exclusivamente aqueles que serão informatizados. O software educativo não se encerra em suas fronteiras físicas. Ele se estende até as ações do professor e do aluno. Neste sentido, a consideração de critérios de interface adequados também é fundamental.

A *Classificação* é a habilidade de realizar agrupamentos em classes, que descrevem conjuntos de objetos com a mesma estrutura de dados e comportamento. No projeto de um software educativo, a classificação de objetos requer uma atenção rigorosa do engenheiro de sistemas, que precisa ter desenvolvido um conhecimento apurado da relação educativa e de seu modo de funcionamento, sob pena de se proceder a classificações equivocadas ou redutoras da complexidade do fenômeno pedagógico. Por essa razão, o desenvolvimento de um software educativo não pode se esquivar de uma equipe interdisciplinar, capaz de se "entreajudar" e de avançar na compreensão de um projeto de sistema que paulatinamente toma forma. O projetista não pode avançar sozinho, solicitando o aval dos proprietários dos subsistemas (professores e alunos) somente após ter concluído seu trabalho. Afinal, como classificar o que ele não conhece profundamente? Quem garante que dois procedimentos educativos, ou dois objetos para se utilizar a terminologia da abordagem empregada, podem ser classificados sob a mesma classe? Afinal, ainda que aparentemente semelhantes, dois objetos, no âmbito de um projeto de software educativo, podem gerar situações cognitivas distintas. Tal raciocínio está igualmente relacionado à *herança*, outro princípio da modelagem orientada a objetos, segundo o qual as características de uma classe são compartilhadas pelos elementos dela. A *herança* permite ao analista expressar características de uma classe uma única vez, e compartilhá-las com seus membros simplesmente identificando-as como membros desta.

O *Polimorfismo* é a habilidade de se ter uma mesma operação com comportamentos diferentes para diferentes classes. A implementação de uma resposta a uma operação é parte do próprio objeto. Desse modo, há uma hierarquia de funções assim como há uma hierarquia de dados. Em um projeto de software educativo, a identificação de situações de polimorfismo entre diferentes objetos só é possível por meio de uma adequada modelagem da cooperação, conforme foi anunciado anteriormente. Situações de polimorfismo são muito comuns em projetos de softwares educativos, tendo em vista a já enfatizada natureza imprevisível das interações cognitivas entre usuários e sistema. Para se avançar com segurança nessa direção, a opinião de especialistas em Psicopedagogia pode esclarecer a existência de comportamentos distintos em um mesmo objeto. Por exemplo, consideremos certo conteúdo pedagógico como sendo um objeto ou uma classe. É fácil perceber que diferentes indivíduos relacionando-se com esse mesmo objeto ou classe podem gerar situações

diferentes. Identificar casos de polimorfismo no projeto do software pode apoiar a tomada de decisão quanto à implementação do sistema, na medida em que certas tarefas podem ser informatizadas ou não.

A *Encapsulação*, ou a *ocultação de informações*, permite a separação entre aspectos internos dos objetos e os aspectos acessíveis por outros objetos. Este princípio impede que o sistema se torne excessivamente interdependente, evitando aumentar a sua complexidade e facilitando sua manutenção, o que raramente é pensado no caso de softwares educativos. Estes, considerados como espaços para a promoção da construção de conhecimentos, estão necessariamente fadados ao fracasso se não são maleáveis o suficiente para conseguirem articular diferentes situações e necessidades de ensino e de aprendizagem. A encapsulação, que pode ser traduzida sob a forma de um procedimento externo ao software, executado pelo usuário, permitindo uma simplificação do projeto a fim de que as relações educativas geradas pelo sistema sejam mais complexas, menos previsíveis e repetitivas. Isto é fundamental para garantir a longevidade do software educativo.

O ciclo de vida de um software educativo, como de qualquer outro, deve cobrir as fases de formulação do problema, análise, projeto, implementação, testes de operação e manutenção. A metodologia proposta por Rumbaugh (1993) consiste em se construir um modelo baseado em objetos do domínio da aplicação e adicionar-lhe detalhes de implementação, de modo que o mesmo modelo possa evoluir da fase de análise, para a de projeto e daí para a de implementação. O propósito da análise é definir e entender o problema e o domínio da aplicação. O projeto gera, sobre o resultado da análise, uma estrutura de alto nível do sistema, que servirá de base para a implementação. A implementação, fase final do processo, é a tradução do projeto em código. O sistema, na técnica da modelagem orientada a objetos, é representado por um modelo subdividido em três partes: modelo de objetos, modelo dinâmico e modelo funcional. Cada modelo representa uma visão ortogonal do sistema em projeto complementando-se para formar o sistema final. A mesma notação é usada durante todo o ciclo de desenvolvimento. Seu emprego nos procedimentos de desenvolvimento dos softwares educativos do Grupo Ábaco trouxe eficiência e eficácia ao processo e um domínio mais claro dos resultados.

Além do exposto, diferentes dimensões devem ser consideradas no processo de engenharia do software educativo. São as dimensões

interativa, didática, cognitiva, lúdica e física. A dimensão *interativa* demanda que o software educativo seja organizado em torno da associação de diferentes recursos (materiais, computacionais e humanos) para dar lugar a um processo dinâmico de ensino e de aprendizagem, com o objetivo de potencializar, ao máximo, o ato de aprender, seja por livre descoberta, através de atividades lúdico-pedagógicas, seja através de situações de trabalho cooperativo. De fato, uma das críticas mais importantes ao emprego de softwares educativos nas séries iniciais do ensino fundamental (1ª à 4º série) enfatiza que tais recursos didáticos isolam a criança do contexto formador proporcionado pela escola, contexto fundamentado na socialização e na interação constante com os pares. Há, portanto, uma importante demanda por softwares educativos que estimulem a socialização ao mesmo tempo em que reforçam conteúdos e apoiam processos de construção de conhecimentos em diferentes áreas de formação. Nesse sentido, o conceito de ambiente de aprendizagem multimediatizado indica uma direção de pesquisa bastante promissora no campo da engenharia de softwares educativos.

A dimensão *didática* refere-se à natureza do conteúdo proposto, à cientificidade do mesmo e ao modo como tal conteúdo é transposto de sua versão científica para uma versão didática adequada e válida, suscetível de ser tratada como matéria de ensino e como objeto de aprendizagem e, principalmente, suscetível de ser interiorizada pelo aluno sob a forma de um saber efetivo, de ser empregada na aquisição de conhecimentos novos e de ser transferida para situações externas ao contexto do ambiente multimediatizado de aprendizagem e do ambiente escolar. Tais considerações levam em conta que o chamado "saber efetivo" é fruto de um processo contínuo de interação do sujeito com informações ou com conhecimentos que lhe são propostos, perpassando interações com saberes anteriormente adquiridos, com representações e interpretações previamente construídas.

A dimensão *cognitiva* diz respeito à estratégia psicopedagógica subjacente ao modo de funcionamento do software, que deve levar em consideração ser a aprendizagem processo ativo de estabelecimento de elos de ligação entre novas informações e conhecimentos anteriores que requer uma reorganização constante de conhecimentos declarativos, procedurais e condicionais, a partir do emprego de estratégias e de metaestratégias. Portanto, as interações previstas devem corresponder a situações de comunicação de conhecimentos nas quais estes últimos

necessitam ser revestidos de sentido, de contexto, de valor, de utilidade, de pertinência, a fim de, mais facilmente, ou com o menor esforço cognitivo possível, serem compreendidos, integrados e assimilados pelo aluno, que deve ser considerado sujeito ativo, afetivo e social. Este aluno, ao longo do processo de construção de conhecimentos, constrói também uma auto-percepção, uma auto-imagem e uma auto-estima, indo de saberes coletivos (como os que são apresentados pelo software) a saberes individuais (como os que ele constrói a partir de sua própria experiência enquanto ser vivo e interativo) e vice-versa. A dimensão cognitiva do software educativo, na qualidade de instrumento de comunicação de conhecimentos, funciona segundo duas vias distintas e complementares em termos de interface. A interface cognitiva, que se relaciona com a apreensão do conteúdo, e a interface física, que se relaciona com a percepção do suporte. No que diz respeito à interface cognitiva, estudos em Psicologia Cognitiva têm demonstrado que cada conceito novo, uma vez assimilado e transformado em saber efetivo, serve de contexto e de estrutura para a aquisição de novos conceitos. Tal dinâmica gera um processo contínuo de reinvestimento do saber do indivíduo na aquisição de conhecimentos inéditos. Assim, o novo objeto de conhecimento será mais bem compreendido e mais bem situado na bagagem cultural já detida pelo indivíduo em aprendizagem, à medida que, de acordo com modelos propostos pelas Ciências Cognitivas a rapidez na apreensão de um novo objeto, qualquer que seja ela, depende muito mais do que o indivíduo já apreendeu do que da complexidade do objeto em si (figura 2).

Figura 2: Modelo da apreensão de novos conhecimentos

Quanto à dimensão *lúdica* de um software educativo, ela não depende unicamente do computador. Este último instaura o fio condutor das interações que, por sua vez, são inteiramente controladas pelos usuários do software, que têm a responsabilidade de manipula-lo e de avançar na tarefa proposta, munidos do livre arbítrio de continuar,

parar, retornar, repetir. É importante que seja adotada, nessa dinâmica, uma situação de responsabilização do aluno pela situação pedagógica instaurada entre ele e um conjunto de saberes comunicados através do recurso informatizado. Nesse sentido, este último deve ser visto unicamente como um suporte, como o coadjuvante de um processo de ensino e de aprendizagem gerenciados pelo professor, mas protagonizados pelos próprios usuários.

Conclusões

O desenvolvimento de sistemas baseado em objetos pode ser empregado com sucesso na grande maioria das aplicações. Pode-se esperar benefícios de um melhor gerenciamento da complexidade dos sistemas e do aumento da produtividade dos programadores como conseqüência da encapsulação e da reusabilidade dos códigos. A manutenção é facilitada, assim como a capacidade de expansão dos sistemas, reduzindo-se os custos também a longo prazo. A herança permite que se adicionem ou eliminem objetos sem comprometer a aplicação como um todo. No caso do desenvolvimento de softwares educativos, é fundamental que uma metodologia dê suporte a um trabalho bastante amplo, em que o "engenheiro de softwares" torne-se também um pesquisador acerca do processo de ensino e de aprendizagem, que devem ser vistos como entidades, como objetivos com os quais os relacionamentos são sempre dinâmicos e normalmente imprevisíveis. Neste sentido, é importante que o próprio conceito de software educativo seja ultrapassado, para que tais meios de comunicação pedagógica sejam percebidos em um contexto mais amplo, envolvendo também os atores que constituem a relação educativa que tomará forma com o suporte do software. Assim sendo, todo e qualquer software educativo deve ser percebido como um ambiente de aprendizagem e, de preferência, como material didático incompleto em relação ao qual diferentes alunos e diferentes professores funcionarão de modo distinto, em função do conhecimento que eles já detêm, de seus estilos de aprendizagem, de sua adaptabilidade ao material didático etc.

Considerando o exposto, no caso de nossos trabalhos, adotar estratégias de modelagem da cooperação entre todos os atores da relação educativa permitiu que avançássemos o conceito do software educativo estático, fechado, de uso delimitado para o de software aberto, mutável, com boa parte de suas atividades realizadas pelos atores humanos, que,

por sua vez, necessitam de diversidade e de adaptabilidade para que possam avançar na construção de seus conhecimentos.

ABSTRACT

This paper deals with particularities of the educational software development process, based on the experience of research and development accumulated by the Abaco Laboratory Team, at the University of Brasilia's Faculty of Education. Firstly, the problems and complexity of educational software engineering are explained. Then, the methodologies and strategies of engineering systems adopted and adapted by the Abaco team are presented: cooperative and object-oriented modeling. Conclusions point out that these methodologies and strategies should be adopted in all attempts at conceiving and developing software for teaching and learning.

Keywords: Educational software. Engineering. Strategies.

Referências

ALMEIDA, M. E.. *Informática e Formação de Professores*. Ministério da Educação, Secretaria de Educação a Distância, Brasília, Brasil, 2000.

BREUER, J.; GREEF, P. Modelling System-User Cooperation in KADS. In G. Schreiber, B. Wielinga et J. Breuker (editores.), *KADS – a Knowledge Approach to Knowledge-Based System Development* (p. 48-70). New York: Academic Press, 1993.

BUTTON, G. et al. *Computadores, Mentes e Conduta*. São Paulo: UNESP, 1998.

CARVALHO, P. S. de. *Interação entre Humanos e Computadores*. São Paulo: EDUC, 2000.

COAD, P.; YOURDON, E. *Análise Baseada em Objetos*. Rio de Janeiro: Editora Campus, 1992

FAGUNDES, L.; SATO, L. S.; MAÇADA, D. L. *Aprendizes do Futuro*: As inovações Começaram. Coleção Tópicos em Informática na Educação. Brasília: Ministério da Educação, 2001.

FRANCO, M. A. *Ensaio Sobre as Tecnologias Digitais da Inteligência*. Campinas: Papirus, 1997.

IUNES, S. *Contratos e Destratos entre Informática e Educação Matemática, 2001*. 196 f.. Dissertação (Mestrado em Educação) – Universidade de Brasília, Brasília, 2001.

JOHNSON, S. *Cultura da Interface*: Como o computador transforma nossa maneira de criar e comunicar. Rio de Janeiro, Jorge Zahar Editor, 2001.

MATURANA, H.; VARELA, F. *Árvores do Conhecimento*. as bases biológicas do entendimento humano. Trad. Jonas Pereira dos Santos. Campinas: Editorial Psy II, 1995.

MORAES, D. *O concreto e o Virtual*: Mídia, Cultura e Tecnologia. Rio de Janeiro: DP&A, 2001.

MORAES, R. de. A. *Informática na Educação*. Rio de Janeiro. DP&A. 2000.

_____. *A Política de Informática na Educação Brasileira*: Do Nacionalismo ao Neo-Liberalismo. 1996 , 290 f., Tese (Doutorado em Educação) Unicamp, Campinas. 1996.

MORAN, J. M.; MASETTO, M.; BEHRENS, M. A. *Novas Tecnologias e Mediação Pedagógica*. São Paulo: Papirus, 2000.

MORETTO, V. P. *Construtivismo*: a produção do conhecimento em aula. Rio de Janeiro: DP&A. 1999

RUMBAUGH et al. *Modelagem e Projetos Baseados em Objetos*. Rio de Janeiro: Editora Campus. 1993.

SAMPAIO, M. N.; LEITE, L. S. *Alfabetização Tecnológica do Professor*. Petrópolis: Vozes, 2001.

SANTOS, G. L. Proposta de uma estratégia holística para a engenharia de softwares educativos. *Revista Brasileira de Tecnologia Educacional*, nº 148. Rio de Janeiro: ABT, 2000, p.22-26.

_____. A integração de princípios de Comunicação Visual e de Psicologia Cognitiva na concepção de um software educativo para apoio à construção de representações acerca de questões ambientais junto a crianças em fase de início de escolarização. *Anais do X Simpósio Brasileiro de Informática Educativa (SBIE98)*, Fortaleza, Departamento de Computação, 1998.

_____A Engenharia de Softwares Educativos. Anais do IV Congresso Ibero-Americano de Informática na Educação (RIBIE98), em cd-rom. Brasília/; CYTED, 1998a.

_____. *Développement d'un savoir fonctionnel à l'aide d'un environnement de formation technique assisté par ordinateur intégrant une approche didactique adaptée*, 1995, 276 f. Tese (Doutorado em informática aplicada à educação -. Universidade Laval, Québec (Canadá), 1995.

_____. Cognitive Modeling and intelligent tutoring systems. *Revue des Sciences de l'éducation*, Montreal, Canadá, v. 19, n. 3, p.501-509, 1993.

SOUZA, A.; SANTOS, G. L. Hércules e Jiló: um software educativo de apoio à formação docente para atuar junto a crianças com deficiência mental. *Linhas Críticas*, Brasília, v. 7, n. 13, p, 27-41, jul./ dez. 2001.

TAKAHASHI. T. (org.) *Sociedade da Informação no Brasil – Livro Verde*. Brasília: Ministério da Ciência e Tecnologia. 2000.

VIRILIO P. *A Bomba Informática*. São Paulo: Estação Liberdade, 1999.

WENGER, E. *Artificial Intelligence and tutoring Systems:* computational and cognitive approaches to the communication of knowledge. Los Altos, California: Morgan Kayfmann Publishers. 1987.