Em busca de uma proposta metodológica para o desenvolvimento de *software* educativo colaborativo

Patricia Scherer Bassani - Centro Universitário Feevale - patriciab@feevale.br
Liliana M. Passerino - Centro Universitário Feevale - liliana@feevale.br
Paulo R. Pasqualotti - Centro Universitário Feevale- ppasqualotti@feevale.br
Marcelo Iserhardt Ritzel - Centro Universitário Feevale -ritzel@feevale.br

Resumo
Apesar da existência de metodologias para especificação de sistemas computacionais consolidadas na área de Computação e Informática, o caráter diferenciado do *software* educativo vem impulsionado a investigação nesta área, em busca de uma proposta que contemple, além dos aspectos computacionais, também os aspectos educacionais, de interface e colaboração. O presente artigo apresenta uma proposta metodológica para o desenvolvimento de *software* educativo colaborativo. Este estudo considera a análise das especificidades dos *softwares* educativos em relação aos de aplicação comercial e industrial, bem como as diferentes metodologias de desenvolvimento propostas pela área de engenharia de *software*. Esta metodologia está sendo validada na construção do ambiente PALCO (Portal Acessível para Linguagem Colaborativa), atualmente em desenvolvimento pelo Grupo de Pesquisa de Tecnologias de Informação do Centro Universitário Feevale.

Palavras Chave: *software* educativo, desenvolvimento de *software*, colaboração.

A methodological proposal for collaborative educational software development

Abstract:
Although exist several methodologies for computational systems specification known in Computer Science and Software Engineering, educational software has own characteristic and need privates methodologies for development. This situation stimulated researches for methodologies that contemplate, beyond computer questions, and, also educational, interface and collaboration questions. The present paper presents a methodological proposal for collaborative educational software development. Considering, educational software differences in comparison with commercial and industrial software, approach educational contexts for collaborative software development and is presented a revision soon about some traditional software engineering approach. This methodology is validated as approach for PALCO (in Portuguese Accessible Gate for Collaborative Language) in development for Information Technologies Research Group of the University Center Feevale.

Keywords: educational software, software development, collaboration.

1. Introdução

Um dos aspectos que chama a atenção no desenvolvimento de *software* educativo é a carência de uma metodologia específica para projetá-los e desenvolvê-los. Enquanto a maioria dos *softwares* aplicativos comerciais enfatizam o processo, o fluxo funcional (entrada e saída), a execução de tarefas, o apoio à tomada de decisão, entre outros, o *software* educativo busca atender e promover a aprendizagem, a demanda cognitiva para

a aquisição do conhecimento e a construção de relações e conceitos. Assim, as decisões no projeto deste tipo de *software* não passam apenas por definições de tarefas/processos e dados que serão armazenados, diferentemente dos outros tipos de *software* cujos resultados se refletem no ambiente externo. As decisões no projeto de *software* educativo focam mudanças internas no usuário, muitas vezes pouco mensuráveis e cujos resultados devem ser de longo prazo.

No contexto deste trabalho, considera-se que *software* educativo é todo programa de computador desenvolvido especialmente para ser utilizado no contexto educacional. Evidentemente que esta definição é o bastante ampla para prever todos os níveis de ensino e todas as formas de ensino (formal e informal), mas restringe-se aos *softwares* que tem um objetivo educacional previsto desde o seu projeto e desenvolvimento. Segundo esta concepção, *software* de aplicativos gerais como editores de texto, imagens, entre outros, não estariam incluídos embora sejam amplamente utilizados nos ambientes educativos. Justificamos esta exclusão pelo interesse em propor uma metodologia adequada para a modelagem e o desenvolvimento de *software* educativo.

Apesar da existência de metodologias para especificação de sistemas computacionais consolidadas na área de Computação e Informática, o caráter diferenciado do *software* educativo vem impulsionado a investigação nesta área, em busca de uma proposta que contemple, além dos aspectos computacionais, também os aspectos educacionais, de interface e colaboração. Oliveira et al (2001) aponta 4 (quatro) parâmetros que distinguem um *software* qualquer de um *software* educativo: fundamentação pedagógica, conteúdo, interação aluno-*software* educativo–professor e a programação.

Assim, um projeto de *software* educativo precisa de definições de requisitos que vão além do contexto imediato de uso (que ainda assim deve ser considerado na concepção do *software*), mas perpassam decisões sobre conteúdos, envolvendo seleção, escolha dos tipos de conteúdos, seqüências, organização visual e didática assim como adaptação aos diferentes tipos de usuários. Nesse aspecto, queremos destacar que por tipos de usuários não estamos apenas nos referindo a faixa etária ou estilos cognitivos, mas também necessidades especiais que os mesmos possam apresentar: surdez, deficiência, visual, dislexia, déficit de atenção, entre muitas outras existentes na sociedade.

Nesta perspectiva, diversos autores (Oliveira et al., 2001, Begoña, 1997, Andrews e Goodson, 1991) têm apresentado algumas propostas metodológicas para o processo de desenvolvimento de *software* educativo. Entretanto, percebe-se que essas propostas tendem a contemplar aspectos educacionais e psicológicos, em detrimento dos aspectos computacionais, enfatizando o caráter descritivo.

Com base nos estudos realizados e buscando ampliar o escopo das pesquisas em andamento, este artigo apresenta uma proposta metodológica para o desenvolvimento de *software* educativo, que busca contemplar a integração dos aspectos computacionais e educacionais, no que se refere à interface e à colaboração.

2. Contextos educacionais importantes no desenvolvimento de um *software* colaborativo

Begoña(1997) considera que as teorias de *design* instrucional são importantes não somente por dar um suporte ao processo de projeto e desenvolvimento de *software* educativo, mas também como *links* entre as teorias de aprendizagem e as teorias de ensino, ou seja, um ponto em comum entre teorias de aprendizagem, que são de caráter descritivas (pois tratam de explicar processos internos de pensamento), e as teorias de ensino, que são do tipo prescritivo, focando nas estratégias de intervenção educativa.

As teorias de aprendizagem construtivistas levam em consideração o ambiente como sendo o cenário onde acontece o processo de interação, que promove e desencadeia o processo de aprendizagem que é, por definição, nessas teorias, um processo

exclusivamente interno. Assim, o cenário torna-se o elemento no qual os "atores" (professores e alunos) irão desenvolver suas "ações" que poderão levar a processos de aprendizagem.

Esses cenários têm alguns elementos importantes:

• Banco de informações ou conhecimento: trata-se de um repositório de fontes de informação, que os atores poderão utilizar, de acordo com as necessidades das atividades que estão desenvolvendo, seja para consulta, seja para incluir novas fontes. Tanto professores como alunos podem realizar todos os processos desde a inclusão, edição até a consulta e exclusão de informações no banco, pois na concepção construtivista de aprendizagem o aluno é um sujeito ativo que mantém o controle do seu processo de aprendizagem. Como o viés neste artigo é a colaboração, sem dúvida o uso deste banco de informações deve permitir a colaboração, seja esta de caráter síncrono ou assíncrono;

• Suportes simbólicos: são espaços para a construção e manipulação de símbolos, objetos e linguagens como diários, portfólios, editores gráficos e de textos inseridos no cenário. Como no caso anterior, o trabalho colaborativo deve ser considerado no projeto e desenvolvimento deste tipo de ambiente;

• Simulações: permitem a apresentação de informações, a observação de fenômenos e a manipulação de hipóteses por parte dos atores;

• Administração e organização do cenário: são elementos que funcionam como espaços de organização do trabalho, como agendas, cronogramas e atividades. Em ambientes centrados no ensino são exclusivas do professor assim como em ambientes construtivistas menos radicais. Nos ambientes construtivistas "radicais" estas funções devem ser negociadas pelo grupo e podem ser realizadas por qualquer um dos atores, não existindo necessariamente a figura do coordenador;

• Espaço de negociação: é constituído por ferramentas de comunicação para trocas de idéias e negociação das ações dos atores. Estas ferramentas podem ser síncronas ou assíncronas e em geral são dinâmicas, ou seja, que são alteradas por todos os participantes como as do tipo fórum, correio, bate papo, editores colaborativos de textos, entre outras.

Com relação à colaboração, Preece et al (2005), apresenta três categorias principais de mecanismos sociais utilizados para coordenar os trabalhos colaborativos. Estes são: mecanismos conversacionais, de coordenação e de percepção, apresentando uma relação de como os sistemas tecnológicos podem ser projetados para facilitá-los.

Os mecanismos conversacionais servem para facilitar o fluxo da conversa, enquanto os mecanismos de coordenação permitem que as pessoas trabalhem juntas e interajam, sendo necessários quando um grupo de pessoas atua/interage/trabalha junto. Estes mecanismos devem prever estratégias para a coordenação das ações dos usuários e prever "*uma política social para controlar a 'tomada da palavra'*" (Preece et al, 2005, p 143), de forma a evitar atualizações/alterações simultâneas que possam originar erros. Por último, mecanismos de percepção (*awareness*) são utilizados para que se descubra o que está ocorrendo, o que os outros estão fazendo e também, permitir que os outros saibam o que está acontecendo.

Considerando que "*a conversa e a maneira como ela é realizada constituem uma parte fundamental da coordenação de atividades sociais*" (Preece et al, 2005, p 130), o desafio tem sido o desenvolvimento de sistemas que permitam a comunicação entre pessoas geograficamente distantes, como se estivessem no mesmo lugar. Diversas tecnologias colaborativas vêm sendo desenvolvidas de forma a possibilitar a conversação, como *e-mail*, videoconferência, mensagens instantâneas, salas de bate-papo, ambientes virtuais de aprendizagem (AVA) e ambientes virtuais colaborativos (AVC). Conforme Preece at al (2005) alguns *frameworks* vêm sendo adaptados de

outras disciplinas, como Sociologia ou Antropologia, para o desenvolvimento de sistemas que possibilitem a comunicação entre pessoas que precisam trabalhar juntas. Estes evidenciam aspectos sociais, como é o caso do *framework* da linguagem/ação. A premissa básica deste é de que as pessoas agem por meio da linguagem e baseia-se na teoria dos atos da fala, que se preocupa com as funções que os enunciados desempenham nas conversações (Searle, 1969). Mas outras pesquisas, que levam em conta os aspectos sócio-históricos e de pragmática envolvidos na comunicação e na interação social também trazem aportes importantes (Wertsch, 1999, Pontecorvo, 2004, Watzlawick et al., 1967).

Incorporar características colaborativas a um *software* educativo vai além de integrar recursos de comunicação que permitam a comunicação entre dois ou mais usuários. Comunicar é essencial, pois é pela troca, pelo fluxo de informações e ações que emergem as descobertas e o aprendizado. Porém, somente comunicar-se não garante que ocorra a colaboração.

Panitz (1996) apresenta seus estudos e entendimento sobre aprendizagem ocorrida pela colaboração e pela cooperação. Suas conclusões definem "cooperar" como construir em conjunto, voltados para objetivos comuns, vinculados pela tarefa, pelo resultado. Por outro lado, "colaborar" refere-se ao processo pessoal de interagir, a necessidade de troca, de comunicar-se, de fazer parte de algo e de construir fazendo parte do coletivo.

Barros (1994) afirma que "*(...) colaborar está relacionado à contribuição enquanto cooperar envolve vários processos – comunicação, negociação, co-realização e compartilhamento*". Também Maçada e Tijiboy (2006) apresentam diversos estudos e levantamentos vinculados a essa discussão, entendendo que "*para haver colaboração o indivíduo deve interagir com o outro existindo ajuda - mútua ou unilateral. Para existir cooperação deve haver interação, colaboração mas também objetivos comuns, atividades e ações conjuntas e coordenadas*".

Neste estudo, entende-se que cooperar surge dos objetivos entre indivíduos em torno de uma tarefa ou ação comum. Cooperar é ajudar, é fazer algo, também junto, mas diferentemente de colaborar, cooperar está vinculado ao objetivo em comum, a mesma finalidade. Colaborar parte das ações de cada indivíduo, quando predisposto a construir para o coletivo. É ajudar a construir. É participar do fazer algo. E esse algo se refere à obra, ao trabalho, à atividade. Ao objeto ou espaço que está sendo construído.

Para que a aprendizagem se realize, é necessário que haja um processo de troca com o outro. A aprendizagem colaborativa apoiada por computador deve ser uma estratégia educativa em que dois ou mais sujeitos constroem o seu conhecimento a partir da discussão, do diálogo, da reflexão, da tomada de decisão, tendo o computador e o *software* educativo o papel de mediador dessa construção, somando-se dessa forma à atuação do professor.

Diversas atividades do ambiente colaborativo são norteadas pelo modelo de colaboração, apresentado na figura 1, segundo uma arquitetura baseada em componentes (Blois & Becker, 2002), cuja idéia central está baseada em comunicação, coordenação e cooperação.

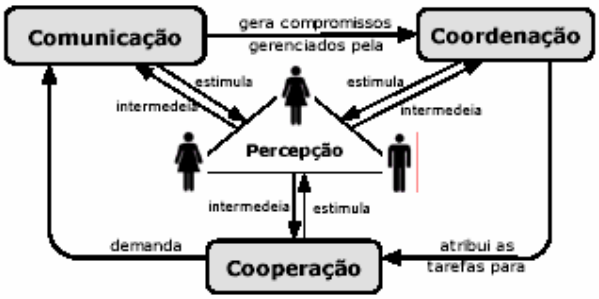

Figura 1 – Modelo de Colaboração (Blois & Becker, 2002)

Assim, considerando a aprendizagem colaborativa como uma atividade na qual os participantes constroem em conjunto um modelo de conhecimento (Souza, 2005). Então, é preciso que essa modelagem seja também levada em conta na metodologia de desenvolvimento de *software* colaborativo, pois o *software* colaborativo deve oferecer atividades nas quais os participantes possam co-construir partes do modelo de conhecimento, expressando, criticando, elaborando, compartilhando cada elemento desse modelo. Desta forma, o que precisa ser incluído no desenvolvimento do *software* é um modelo de interação social como elemento central para a criação de *comunidades de aprendizagem.* Nesse contexto observa-se a inserção da metáfora das comunidades de agentes, de modo a possuir uma visão geral do ambiente e de seus participantes atribuindo caráter mais real ao contexto.

3. Metodologias para desenvolvimento de *software*

A seqüência de atividades realizadas durante o processo de desenvolvimento de um *software* é conhecida como "ciclo de vida". O modelo tradicional de ciclo de vida do *software*, conhecido como modelo em cascata, envolve as seguintes atividades: levantamento de requisitos, análise, projeto, implementação, testes e implantação. Neste modelo, as etapas são realizadas de forma linear, entretanto, sempre que houver alguma alteração, todo o processo deve ser revisto.

Na etapa de levantamento de requisitos são realizadas as discussões acerca do *software* a ser desenvolvido, definindo-se as funcionalidades, a partir de um estudo exploratório das necessidades dos (possíveis) usuários e da situação do ambiente atual, caso este existir. A etapa de análise tem por objetivo modelar os requisitos delineados na etapa anterior, de forma a estudar como todos os componentes do sistema interagem entre si. A representação é feita por meio de modelos/diagramas, que representam o sistema a ser construído.Os diagramas variam conforme a abordagem utilizada, em geral opta-se por análise estruturada ou análise orientada a objetos. Na análise estruturada o foco centra-se nos processos a serem implementados e nos fluxos de informação entre estes. Destacam-se como diagramas principais desta abordagem o DFD (Diagrama de Fluxo de Dados), o modelo ER (entidade-relacionamento) e dicionário de dados (Gane e Sarson, 1987).

Por outro lado, a análise orientada a objetos contempla a modelagem de um sistema, a partir da definição/identificação de classes, objetos, atributos e operações. O objeto é compreendido como uma unidade autônoma, que contém seus próprios dados, é manipulado por processos definidos de forma específica para o objeto e interage com outros objetos de forma a atingir os objetivos (Bezerra , 2002). Sistemas computacionais modelados na perspectiva de orientação a objetos utilizam a linguagem gráfica UML (*Unified Modeling Language*). Ela descreve 13 (treze) tipos de diagramas oficiais, entretanto os mais comuns são diagrama de classes, diagrama de casos de uso e diagrama de interação.

Entende-se que estas metodologias tradicionais não se aplicam adequadamente à modelagem de *software* educativo, pois enfatizam essencialmente aspectos computacionais, a partir da identificação dos dados de entrada do sistema, especificação acerca do processamento destes e a apresentação de dados de saída.

3.1 Propostas metodológicas para o desenvolvimento de *software* educativo

Oliveira et al. (2001) propõe uma metodologia recursiva, que "*tem como fundamentação teórica a concepção interacionista e construtivista do conhecimento* (p. 97)" A proposta enfatiza o critério "coerência com os objetivos educacionais" e apresenta as atividades relativas a planejamento, desenvolvimento e avaliação de *software* educativo:

a) escolha do conteúdo;
b) análise dos conhecimentos prévios, necessários ao usuário, para utilização do *software*;
c) identificação dos conceitos estruturantes do conteúdo;
d) desenvolvimento do diagrama de fluxo, representado pelas telas do *software* e suas relações/interconexões;
e) desenvolvimento das telas, envolvendo layout e orientações para implementação;
f) implementação das telas;
g) desenvolvimento da documentação, incluindo instruções sobre instalação, características de hardware e manual do usuário;
h) utilização, avaliação e manutenção do *software* educativo.

Também Campos et al. (1998) entende que "*os produtos de software devem refletir os objetivos educacionais propostos e o ambiente de aprendizagem almejado, criando situações que estimulem o desenvolvimento das habilidades almejadas"*. Para tanto, sugerem que o processo de desenvolvimento de *software* adequado à hipermídia educacional é a prototipagem evolutiva acrescida de uma etapa inicial, que consiste na definição do ambiente de aprendizagem, isto é, a filosofia de aprendizagem subjacente ao *software*.

Gomes e Wanderley (2003), apontam que o desenvolvimento de aplicações educativas é "*há muito tempo considerado como sendo uma atividade quase artesanal*". Os autores reforçam a importância da identificação inicial de requisitos necessários à implementação do *software* educativo, enfocando que estes devem observar tanto aspectos do processo de aprendizagem dos alunos quanto aspectos de mediação promovida pelo professor.

Considerando as propostas metodológicas para desenvolvimento de *software* educativo advindas de estudos na área de Informática na Educação, percebe-se que estas enfatizam a etapa de levantamento de requisitos, apontando a importância da definição da abordagem pedagógica. Diferentemente da área computacional, que apresenta diretrizes específicas para a modelagem de *software* a partir de diferentes diagramas, as propostas relativas ao desenvolvimento de *software* educativo tendem a apenas apontar o que deve ser feito em cada etapa do processo, embasadas essencialmente em critérios qualitativos e/ou mapas conceituais e voltadas para a construção de hipermídia.

Nesta perspectiva, tornam-se relevantes os estudos na área de agentes computacionais. Atualmente, a arquitetura de sistemas educacionais mais utilizada é baseada em uma sociedade de agentes (Vavasori, 1998) (Sanchez, 1998) (Clara, 2002), ao invés de considerar arquiteturas tradicionais/convencionais.

Os princípios dos sistemas multiagentes têm apresentado um potencial bastante adequado ao desenvolvimento de sistemas educacionais, em função da natureza da questão ensino-aprendizagem ser mais facilmente resolvida de forma cooperativa e colaborativa.

Para ambientes educacionais baseados em uma sociedade de agentes, como metáfora ao comportamento social, é essencial delimitar as percepções, as ações e o objetivo de cada agente. Essas informações são importantes para o desenvolvedor possuir uma visão geral dos estados do ambiente, bem como das possíveis comunicações.

A modelagem da comunicação entre os agentes assume importante papel, pois é necessário possuir uma delimitação das ações dos agentes. Diante disso, a presente proposta metodológica propõe a utilização de AUML – *Agent Unified Modelling Language* – (Odell, 2001) como técnica de modelagem para os agentes envolvidos, cuja arquitetura modela o protocolo de comunicação (Finin, 1996) entre os agentes.

Existem poucas metodologias definidas para modelagem de agentes em ambientes educacionais (Webber, 2002) e, também, ferramentas de Engenharia de Software que auxiliem o desenvolvimento deste tipo de sistema. De fato, ainda não existe um consenso sobre metodologias para modelar este tipo de sistema (Bastos, 2000).

AUML foi criada a partir de um projeto desenvolvido por Odell (2001), visando adaptar a linguagem UML às características inerentes ao paradigma de Orientação a Agentes. A proposta inicial de AUML é representar protocolos de interação de agentes (PIA) que descrevam um padrão de comunicação, como sendo uma seqüência de mensagens permitida entre agentes e as restrições sobre o conteúdo destas mensagens. Nos diagramas da AUML, os fluxos de controle e informação são considerados atos de comunicação.

Como fundamenta Bastos (2000), o protocolo todo é tratado como sendo uma entidade, apresentando-se através de diagramas que representem o fluxo da mensagem entre os agentes (diagramas de seqüência, de colaboração, de atividade e de estados) dentro de um pacote, constituindo uma agregação conceitual de seqüências de interação.

Assim, pensa-se que a metodologia deva contemplar aspectos que possam expressar de forma mais real o contexto educacional em seu nível mais amplo, que permita identificar naturalmente especificidades associadas ao plano pedagógico/cognitivo do ambiente e de seus participantes, propondo e alterando aspectos metodológicos e estruturais de acordo com as percepções feitas a partir das interações sobre esse ambiente.

4. Em busca de uma proposta metodológica para o desenvolvimento de *software* educativo colaborativo

A partir dos estudos realizados, entende-se que uma metodologia adequada para a especificação e desenvolvimento de *software* educativo deve contemplar aspectos computacionais e educacionais, de interface e colaboração. Dessa forma, considera-se importante contemplar as seguintes atividades:

a) *levantamento de requisitos*: caracteriza-se pela especificação da abordagem pedagógica, definição do tipo de *software* (tutorial, exercício, jogo, etc.), indicação do público-alvo, profundidade e abrangência do conteúdo e perspectivas de colaboração. Propõe-se um projeto centrado no usuário desde sua concepção, e para tal, torna-se importante envolver todos os usuários/agentes participantes (alunos, professores, atores em geral) e identificar os possíveis cenários pedagógicos. E, nesse sentido, numa primeira aproximação, elabora-se um relatório descritivo em linguagem natural, apresentando as características gerais do *software* a ser desenvolvido.

Uma das finalidades desta etapa é delinear o *software* em termos de usabilidade, considerando proximidade com o usuário (necessidades, potencialidades e limitações) e seus modelos de aluno e a proximidade com o contexto de uso.

b) *análise*: esta etapa consiste em especificar detalhadamente o funcionamento do *software*, de forma a orientar o processo de prototipação. Como resultado desta atividade, tem-se a modelagem do *software* considerando diferentes pontos de vista: dos

processos a serem implementados, do *design* de telas, do *design* de interação, da usabilidade do *software* e do banco de dados/conhecimento. A seguir propõe-se um conjunto de técnicas a serem utilizadas de forma combinada. Contudo, o nível de complexidade do software será determinante para a escolha de quais técnicas serão aplicadas e em que nível de profundidade. As técnicas de modelagem indicadas nesta etapa são:

- *modelagem de cenários:* nesta etapa propõe-se uma prototipagem horizontal, ou seja, definir todas as funcionalidades do *software* em nível conceitual, ficando para a etapa seguinte a prototipação vertical, na qual se aprofundam as funcionalidades individualmente. Os cenários são importantes para representar as situações interativas, compreender o contexto, observar os elementos em interação (atores e objetos), a navegabilidade, estados do sistema e os processos de colaboração propiciados pelo *software*. Para a modelagem dos cenários propõe-se uma combinação das seguintes técnicas: diagrama de casos de uso, *storyboard* e descrição em linguagem natural (ver figura 2);
- *diagrama de fluxo de dados* (DFD), para visualizar os processos internos do *software* com descrição dos processos/funcionalidades em linguagem natural (descrição de cada processo/funcionalidade; dados de entrada e saída; interação com banco de dados; telas relacionadas do projeto de interface);
- *modelo entidade-relacionamento* (ER), para compreender os fluxos de informação e de conhecimento;
- *projeto de interface*, apresentando as telas do *software*, *design* de interação e a navegabilidade.

c) *Projeto/Prototipação*: Nesta etapa se desenvolve a prototipação vertical, que consiste em aprofundar cada uma das funcionalidades e cenários previstos na etapa anterior. Aqui, a presença do usuário é indispensável para permitir os ajustes necessários. A idéia central é "colecionar" dados gerando bases de dados, que analisados em conjunto podem contribuir para melhor conhecer o perfil dos usuários, a adaptabilidade do ambiente e de sua interface, bem como ajustar os requisitos de acessibilidade e usabilidade.

d) *Testes/Validação:* considerando que a finalidade de um *software* educativo é a aprendizagem, esta etapa adquire importância maior pois além de verificar se os requisitos de *software* estão contemplados, é necessário também que seja feito um estudo de caso envolvendo uma pesquisa empírica e, preferencialmente, em campo com situações reais de aprendizagem. Os resultados desta pesquisa poderão levar à novas versões do *software* ou a futuros desenvolvimentos de outros *softwares* a partir da identificação de novos problemas de pesquisa. Nesta etapa, sugere-se adotar uma metodologia de pesquisa quantitativa e qualitativa para estudos de caso, baseando-se, essencialmente, na observação e entrevistas como principais técnicas de pesquisa.

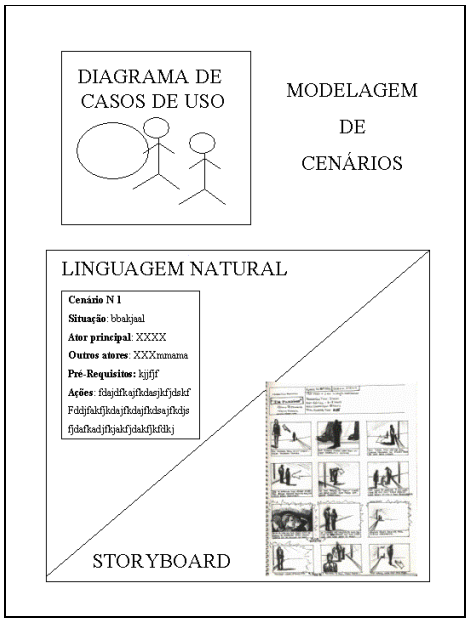

Figura 2: Modelagem de Cenários

O processo de modelagem, embora apresentado de forma linear, segue uma proposta recursiva, no sentido de que, a cada etapa, é possível retornar às etapas anteriores, para ajustes e maior refinamento.

5 Considerações Finais

A partir deste estudo o grupo encontra-se em fase de modelagem do ambiente PALCO (Portal Acessível para Linguagem Colaborativa), que visa apoiar o desenvolvimento da linguagem colaborativa através de um ambiente acessível que considere diferentes tipos de necessidades (auditivas, visuais e cognitivas), uma vez que a linguagem é vista neste projeto como um mecanismo de construção de contextos sociais. No senso comum, atribui-se o uso da linguagem a sua função mais direta: a comunicação. Mas, a linguagem não é apenas um meio para transmitir informações, é também, e principalmente, um mecanismo de construção de contextos sociais. A criação e recepção de mensagens passa por posturas ativas dos sujeitos, que se envolvem na comunicação atribuindo significados, "interpretando" as mensagens e "reconstruindo" um contexto social, que é particular e público ao mesmo tempo. Particular do ponto de vista de constituição de sentido e público enquanto significado. Este contexto dinâmico sofre alterações durante o processo de interação a partir das percepções e interpretações compartilhadas pelos sujeitos. A partir disso, toda comunicação, além de transmitir um conteúdo, define também uma relação. Nesse sentido, o ambiente PALCO visa estudar como promover o desenvolvimento da linguagem de forma que pessoas com diferentes níveis cognitivos, auditivos e visuais possam participar de um contexto social de construção de narrativas.

Referências bibliográficas

ANDREWS, D.H., GOODSON. *A comparative analysis of models of instructional design*. IN: Instructional Technology. Past, Present and Future(Ed). Englewood, Colorado:Libraries Unlimited, 1991. p. 133-155

BARROS, L. *Suporte a Ambientes Distribuídos para Aprendizagem Cooperativa*. Tese de Doutorado. Rio de Janeiro: COPPE, 1994.

BASTOS, R. M., OLIVEIRA, J. P. M. *A conceptual modeling framework for multi-agent information systems*, 19th International Conference on Conceptual Modeling (ER2000), pp. 9-12, October, 2000.
BEGOÑA, G. (coord). *Diseños y programas educativos: pautas pedagógicas para la elaboración de software.* Barcelona: Editorial Ariel, 1997.
BEZERRA, E. *Princípios de análise e projeto de sistemas com UML.* Rio de Janeiro: Campus, 2002.
BLOIS, A. P. T. B., BECKER, K. A. *Component-based Architecture to Support Collaborative Application Design*, 8th International Workshop on Groupware (CRIWG). LNCS Vol. 2440. Springer-Verlag, p. 134-146, 2002.
CAMPOS, F., CAMPOS, G., ROCHA, A. R.*Dez etapas para o desenvolvimento de software educacional do tipo hipermídia.* RIBIE, 1998.
CLARA, I. P.; JOSE,L. M.; JOSEP, L. de L. R. *Intelligent Agents in a Teaching and Learning Environment on the Web*. ICALT 2002 IEEE, Rússia, 2002.
GANE, C., SARSON, T. *Análisis estructurado de sistemas*. Bs.As.: El Ateneo, 1987
FININ, T. *UMBC KQML Web.* Lab for Advanced Information Technology, 1996. Disponível em www.cs.umbc.edu/kqml.
GAMEZ, L. *A construção da coerência em cenários pedagógicos on-line: uma metodologia para apoiar a transformação de cursos presenciais que migram para a modalidade de educação a distância*. Tese de Doutorado em Engenharia da Produção, UFSC, Florianópolis, 2004.
GOMES, A. S., WANDERLEY, E. G. *Elicitando requisitos em projetos de software educativo.* WIE, 2003.
ODELL, J. D., PARUNAK, H. V. D.; BAUER, B. S. V. *Representing agent interaction protocols in UML.* 22nd International Conference on Software Engineering (ISCE), pp. 121–140, 2001.
PANITZ, T. *A definition of collaborative vs cooperative learning*, 1996. Disponível em www.lgu.ac.uk/deliberations/collab.learning/panitz2.html (03/06/2006).
PREECE, J. *Design de Interação: além da interação homem-computador*. Porto Alegre: Bookman, 2005.
PONTECORVO, C. et alli. *Discutindo se aprende: interação social, conhecimento e escola.* Porto Alegre: ArtMed, 2004
SANCHEZ, J., AYALA, G. *User agents in digital libraries and collaborative learning environments*. 1998. Disponível em ict.udlap.mx/people/alfredo/cic98/taller-agents/ centia.html.
SEARLE, J. R. (1969) *Speech acts.* An essay in the philosophy of language. Cambridge: CUP.
SOUZA, R. R. *Contribuições das teorias pedagógicas de aprendizagem na transição do presencial para o virtual*. In: COSCARELLI, C. V. e RIBEIRO, A E. (org). Letramento Digital: Aspectos sociais e possibilidades pedagógicas. Belo Horizonte: Ceale, 2005.
TIJIBOY, A.V., MAÇADA, D.L. *Cooperação/Colaboração em Ambientes Telemáticos.* Disponível em www.niee.ufrgs.br/cursos/topicos-ie/ana/coop2.htm. (04/06/2006)
VAVASORI, F. B., GAUTHIER, F. A. *Proposta de ferramentas e agentes inteligentes para um ambiente de ensino/aprendizagem na Web*. In: IX SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO. Fortaleza/CE, Brasil. Novembro, 1998.
WATZLAWICK, P et alli. *Pragmática da Comunicação Humana: um estudo dos padrões, patologias e paradoxos da interação***.** São Paulo: Editora Cultrix, 1967.
WERTSCH,J. *La Mente en Acción***.** Buenos Aires: Aique, 1999.
WEBBER, C., PESTY, S., BALACHEFF, N. *A multi-agent and emergent approach to learner modelling*. ICALT 2002 IEEE, Rússia, 2002.