Dialogismo: a idéia de gênero discursivo aplicada ao desenvolvimento de software educativo

Flávia Peres[1], Luciano Meira[2]

[1]Departamento de Educação – Universidade Federal Rural de Pernambuco (UFRPE)
52171-900 – Recife – PE – Brasil

[2]Departamento de Psicologia – Universidade Federal de Pernambuco (UFPE)
CFCH – 50.670-901 – Recife – PE – Brasil

peres.flavia@gmail.com, luciano@meira.com

Abstract. *This article presents our use of the literary notion of "discursive genre" as a component of methods for educational software development. We observed designers' and engineers' activity in a Software House, followed the many phases a piece of software undergoes from development to usage in wider contexts, and analyzed users' activities with the software. We concluded that our knowledge of teachers' and pupils' dialogical practices is critical to educational software development, and that we would greatly profit from modeling those practices to software.*

Resumo. *Este artigo discute o uso da noção de "gênero discursivo" como parte de um método para o desenvolvimento de software educativo. Registramos e analisamos a atividade de engenheiros e designers em uma fábrica de software, seguimos os artefatos ali construídos e analisamos a atividade de seus usuários. Concluímos que é crucial conhecermos as práticas dialógicas em que se engajam alunos e professores, e que seria muito proveitoso modelar tais práticas nos softwares que desenvolvemos.*

1. Introdução

O pressuposto central em Bakhtin (1979), segundo o qual o modo de funcionamento da linguagem não pode ser apreendido pela observação das unidades da língua, mas somente nas relações dialógicas entre os enunciados, foi também a chave para todas as análises empreendidas nesta pesquisa. Assumimos que uma perspectiva dialógica é congruente com os trabalhos da psicologia sócio-cultural, originada no início do século XX (Van der Veer e Valsiner, 1999) por Vygotsky (2001) e Leontiev (2004). Desse ponto de vista, concebemos a cognição como *situada* e *distribuída,* tal como discutiremos a seguir, juntamente com nossa unidade de análise e os aspectos metodológicos que nos conduziram ao longo do trabalho. Estas discussões nos servirão de cenário para tratarmos a questão do engajamento de alunos e professores em fábricas de softwares especializadas no desenvolvimento de aplicações educacionais.

Como conseqüência de uma visão situada (Suchman, 1987) e distribuída (Hutchins, 1990) de cognição, a criação de máquinas mais "responsivas" deveria considerar os ambientes em que são utilizadas. Este tipo de pressuposto já representa um truísmo na área de IHC (Interação Humano-Computador), na qual se tem

argumentado em favor da interação designers-usuários e enfocado no modo como estes atores interagem na condução do desenvolvimento de interfaces. Tome-se como exemplo da aplicação desse pressuposto o "modelo escandinavo" (Spinuzzi, 2002), metodologia que emergiu no início da década de 80 e na qual os designers perceberam a importância de tornar o design de sistemas mais "democrático" e "participativo". Este movimento inspirou a defesa ampla da inclusão do usuário na atividade de design, como é o caso de perspectivas como a do *design participativo* (*participatory design*), do *cooperative prototyping* (Bodker, 1991; Bodker e Gronbæk; 1996), e do *design interativo* (Woodruff, Szymannski, Grinter, Aoki, 2002). Outros estudos, propostos como uma teoria em IHC, apontam para modelos eficientes de design, como a *Engenharia Semiótica* (De Souza, 1993; 2005). Como teoria, a Engenharia Semiótica tem várias ferramentas epistemológicas que servem não somente à pesquisa, mas também às atividades próprias ao processo de design.

A abordagem *dialógica* que propomos para o fenômeno da *Interação Humano-Computador* reforça, sob uma nova ótica, esta meta de continuamente reduzir a distância que separa desenvolvedores e usuários. Muitos estudos, por exemplo, já encontraram na *Teoria da Atividade* (Leontiev, 2004) uma base sólida para o desenvolvimento de suas pesquisas neste campo. A ênfase no entendimento das ações individuais no contexto da atividade coletiva de que fazem parte, ampliou os horizontes para uma melhor compreensão da relação entre pessoas e máquinas. Muitos desses estudos têm investigado o papel das ações realizadas durante o uso de determinado software, em um modelo de desenvolvimento que prevê a colaboração entre designers de sistemas de informação e usuários (Bodker e Grobaek, 1996; Engëstrom e Middleton, 1996). De nossa parte, tentamos ver a ligação desta teoria de base sócio-histórica com o *dialogismo bakhtiniano*. Trilhamos caminhos afins, mas combinamos essas diretrizes ao conceito de *gênero discursivo* (Bakhtin, 2003) e direcionamos nossos interesses para as implicações educacionais que esta perspectiva pode oferecer.

Sob o argumento de Suchman (1987), segundo o qual não é possível determinar a intenção subjacente a uma ação, no nível de uma causalidade estrita ou correspondência direta, nos opomos às perspectivas cognitivistas e, no lugar, propomos que a intencionalidade seja vista dentro de um horizonte de (proto-)tipificações, de modo situado. A partir desta visão, uma solução ao problema do design, em particular do design de ambientes educacionais informatizados, é seguir um entendimento alternativo sobre a natureza das intenções e suas relações com as ações. Sob este foco, identificar intenções para a aprendizagem de conteúdos específicos e conceitos científicos passa a ser uma realização largamente contingente, interacional e dialógica.

Suchman nos convida a olhar para o design de máquinas interativas, teorizarmos sobre tal "interação", e, alternativamente, pensarmos sobre o design de interfaces educacionais. Deste ponto de vista, o valor dos artefatos não está em suas estruturas intrínsecas, mas na integração dos mesmos com a prática de alunos e suas contribuições sociais e materiais a essas práticas, seu ambiente de uso. É com base nestas idéias de ação e cognição situadas que propomos a metáfora do diálogo, em contraposição à metáfora da interação, para as relações emergentes entre humanos e máquinas. E o diálogo num sentido tão amplo quanto o círculo de Bakhtin propôs, e que abarca eventos de conversação face-a-face, de autoria e leitura de textos, ou mesmo a reflexão solitária. O que acontece é um encontro de pessoas particularmente situadas com

"coisas" igualmente particulares, dinâmicas e culturais. Perspectiva esta que é bem-vinda à vygotskyanos e bakhtinianos, na dinâmica de vozes que precedem e alimentam uma máquina, bem como a sucedem numa cadeia dialógica ininterrupta.

1.1. Alternativa ao design de artefatos: a idéia de gênero discursivo

Sob o crivo da cognição situada, os estudos em design podem ser conduzidos ainda em outra direção, embora orientada a partir de delineamentos metodológicos similares aos das propostas referidas acima. Com propostas alternativas, Brown e Duguid (1996) acreditam que um artefato com um bom design pode providenciar pistas periféricas que sutilmente direcionam os usuários num caminho interpretativo particular, por evocar conhecimentos culturais e sociais (p. 131). Portanto, o essencial em tornar as coisas fáceis ao uso seria manter juntos contexto e conteúdo. Para isso, torna-se fundamental o estudo das práticas sociais em que os objetos estão situados.

Para Brown e Duguid, o conceito de *gênero* passa a ser um importante conceito no design de software porque: (i) em qualquer forma de comunicação os gêneros emergem do conhecimento socialmente compartilhado; (ii) a informação está sempre ligada a um gênero ou outro, e entender gênero é crucialmente importante na era da informação; (iii) a adequação de uma nova tecnologia requer novos gêneros, os quais emergem naturalmente e podem ser o caminho para o "design consciente". Design de gênero, tal como argumentam, seria o caminho para uma perspectiva inovadora em design (p. 144). Päivärinta e Tyrväinen (1998) também sugerem que a teoria dos gêneros pode oferecer conceitos importantes para o desenvolvimento de sistemas de informação, embora admitindo que muito trabalho ainda deve ser feito para fundamentar melhor esta perspectiva.

Esse parece ser realmente um caminho válido para o design de interfaces. Atualmente, muito se tem estudado sobre a emergência de gêneros digitais decorrentes das novas tecnologias (Marcuschi, 2004), entendendo-se que os softwares são suportes para variados gêneros discursivos. A compreensão das formas discursivas historicamente estáveis, embora em constante processo de mudança, as quais orientam os enunciados nas práticas de alunos e professores em interação no contexto escolar, podem ser uma excelente ferramenta de construção de aplicações educacionais mais adequadas.

2. Metalingüística bakhtiniana: método, *corpus* e unidade de análise

O objeto de estudo na *metalingüística* bakhtiniana (Bakhtin, 2002) é o enunciado, objeto lingüístico que dialoga retrospectivamente com os elos discursivos aos quais responde, e através do qual são produzidas as relações com as vozes dos outros que o precedem. Mas dialoga também prospectivamente com outros, através da antecipação responsiva de co-enunciadores. As concepções de "destinatário" são ligadas às práticas sociais específicas que dão contornos discursivos aos enunciados. Logo, os enunciados emergem de gêneros discursivos, sendo estes, portanto, anteriores àqueles, delimitando (ainda que não por completo) o campo dos enunciados que podem ser gerados, os que não podem, e a relação entre eles. Podemos dizer que os gêneros discursivos orientam as enunciações. Se os gêneros orientam as enunciações, é a partir da compreensão dos diferentes gêneros que emergem em diferentes práticas que se poderia delinear os

contornos dos enunciados que, ultimadamente, serão encapsulados nos softwares na forma de estados, ações etc. Consideramos que a língua vive concretamente apenas nas ações dos sujeitos em enunciação. Essas ações nunca ocorrem soltas num vácuo contextual. Ao contrário, pertencem a gêneros que, por sua vez, nelas se articulam.

O que nos interessou, portanto, em relação aos enunciados de desenvolvedores e usuários, foram suas relações de enderecividade e responsividade (marcas do sujeito em direção ao outro), as quais entram em jogo ao longo do processo de desenvolvimento-uso. Através de recursos da Análise Interacional (Jordan e Henderson, 1995; Goodwin, 2000) e da videografia, analisamos as atividades de desenvolvedores e usuários. Como as concepções de "destinatário" são ligadas a gêneros específicos e, logo, dão contornos discursivos aos enunciados, verificamos aí os impactos das práticas sociais sobre as atividades analisadas.

A construção do *corpus* aconteceu em dois momentos: processo de desenvolvimento e processo de uso. No primeiro momento, participamos de uma disciplina de *Engenharia de Software*. As equipes de desenvolvimento de software foram escolhidas a partir da disciplina Engenharia de Software (IN953), oferecida pelo programa de Pós-Graduação em Ciências da Computação do Centro de Informática da UFPE. Entre os objetivos da disciplina, um nos interessara particularmente: criar uma metodologia para desenvolvimento de software livre.[1] Para atingir os objetivos da disciplina, os alunos participaram de atividades de aula, leituras, práticas e experimentos reais. Durante o período letivo (4 meses), deveriam montar e executar "fábricas de softwares" (Aaen, I.; Botcher, P.; Mathiassen, L, 1997). Os softwares a serem desenvolvidos eram solicitados por clientes reais que apresentaram requisições para problemas reais. Isso levou as fábricas a lançarem propostas de projetos de desenvolvimento a fim de solucionarem os problemas colocados, comprometendo-se a entregar "protótipos" ao longo do período (uma vez por semana), especificando as funcionalidades e eficiências desses protótipos. Em suma, os alunos vivenciaram situações de desenvolvimento, as quais exigiam tomadas de posição e resolução de problemas típicos dos contextos profissionais de desenvolvimento de software, pois originados de problemas reais para os quais as equipes identificaram soluções, testaram, revisaram e transformaram códigos, documentaram o processo e apresentaram protótipos de softwares aos clientes.

A fábrica que analisamos neste artigo foi denominada por seus componentes de "Trend": esta fábrica trabalhou sobre um projeto já existente, cujo software *Project Management Knowledge*–PMK (Torreão, 2005) já possuía alguns protótipos e requeria desenvolvimento. A cliente que buscou a fábrica, portanto, possuía muito conhecimento do produto, por ser de sua autoria, e sua voz serviu-nos também como "voz de desenvolvedor", já que assim se posicionava num primeiro momento. PMK é um

1 Open Source Software (OSS) é uma inovação nas cenas de desenvolvimento de software, aparecendo em meados da década de 90 e atraindo até hoje um certo número de profissionais. Os OSS são softwares cujos códigos são abertos, com poucas limitações sobre possíveis modificações e uso por terceiros, permitindo inspeção e reuso de códigos de programação a partir de algum tipo de "licença open source" (Crowston e Scozzi, 2002). Vale ressaltar que embora os projetos analisados para esta pesquisa fossem de código aberto (Open Source), os produtos foram desenvolvidos nos moldes de fábricas de software tradicionais, ou seja, houve planejamento, processo de desenvolvimento, etc., e, mais importante, as fábricas possuíam clientes que definiram os requisitos do software.

ambiente de aprendizagem que se combina com um agente inteligente, o *Virtual Intelligent Companion for Tutoring and Reflection*–VICTOR, para prover suportes à aprendizagem de estudantes ou profissionais interessados no desenvolvimento de projetos.

O momento de uso foi focalizado em torno de atividades de usuários finais com os softwares cujas origens e desenvolvimentos acompanhamos nas fábricas. Para este segundo momento, escolhemos usuários cujo perfil se enquadrasse, em certa medida, ao perfil de "usuário final" pressuposto pelos desenvolvedores e para o qual tivesse sido desenvolvido o programa. Acompanhamos as atividades de uso e videografamos as mesmas, permanecendo no local do uso enquanto a atividade se dava.

3. Análise de um exemplo

Observamos que muitas vezes o usuário, ao marcar-se como *eu*, requer um *tu*, e nem sempre o software possui enunciados encapsulados suficientemente bem acabados para se aproximar da voz do desenvolvedor e se marcar como essa outra "pessoa" (tu) na interação. É aí, nesse momento de ruptura, que vemos o usuário direcionando-se a um outro que possa suprir-lhe respostas referentes ao uso ou ao programa, passando o software, assim, à posição de *ele.* Como veremos abaixo muitas situações de ruptura ocorridas na atividade de usuários finais do PMK apareceu nas atividades dos desenvolvedores. No entanto, lá no desenvolvimento, as questões eram pertinentes à atividade de autoria do software e não à atividade de uso, propriamente, o que tornam diferentes seus lugares e, logo, falta ao desenvolvedor um distanciamento necessário ao acabamento de futuras ações discursivas.

Como vemos abaixo (Ex. 1) essa mesma mensagem (linha 8; Figura 1) apareceu nas situações de uso em que os desenvolvedores se situavam como usuários (ver Ex. 2). No entanto, lá no desenvolvimento, as questões eram pertinentes à atividade de autoria do software e não à atividade de uso, propriamente, o que tornam diferentes seus lugares e, logo, falta ao desenvolvedor o distanciamento mencionado acima.

Ex.1	**Videografia de situação de uso**
1	/.../((preenchendo formulário eletrônico))
2	((seleciona área de conhecimento, entre as ofertadas <psicologia>))
3	- tem que preencher esse negócio" empresa" instituição" eu vou botar de lá
4	<XXXXXXXX> ((digita o nome da instituição da qual é integrante))
5	- nível de conhecimento sobre projetos, é" ((lendo o campo <nível de
6	conhecimento>
7	- <baixo>
8	((aparece mensagem de erro: <operação inválida>))
9	- ele não diz aqui que foi falha minha, não, mas o que foi" ele diz que foi do
10	sistema.
11	/.../

No exemplo a seguir (Ex. 2), a usuária não atenta para o enunciado "Em desenvolvimento" (Figura 2) que aparece na tela e tenta dar prosseguimentos às suas ações. Observamos que essas ações já aparecem com contornos específicos à prática com hipertextos, cujos movimentos do cursor do mouse sobre determinados elementos

da interface transforma e aterá a interface, levando a novas ações. Quando a usuária diz "então por que não tá aparecendo aquela mãozinha" (linhas 2 e 3), ela está fazendo referência aos limites e possibilidades de suas ações em direção à tela.

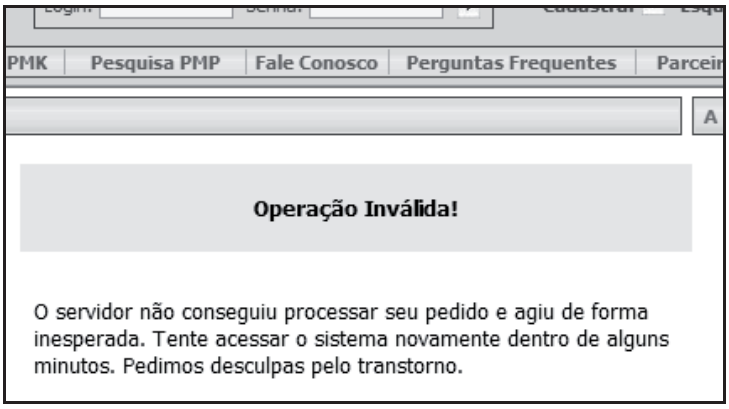

Figura 1. Detalhe da interface do PMK – mensagem de erro

Ex.2	Videografia de situação de uso
1	<em desenvolvimento>
2	- oxente! né pra clicar não" então por que não tá aparecendo aquela
3	mãozinha"
4	- ta em desenvolvimento, não está terminado ainda.
5	- A' poxa'
6	/.../

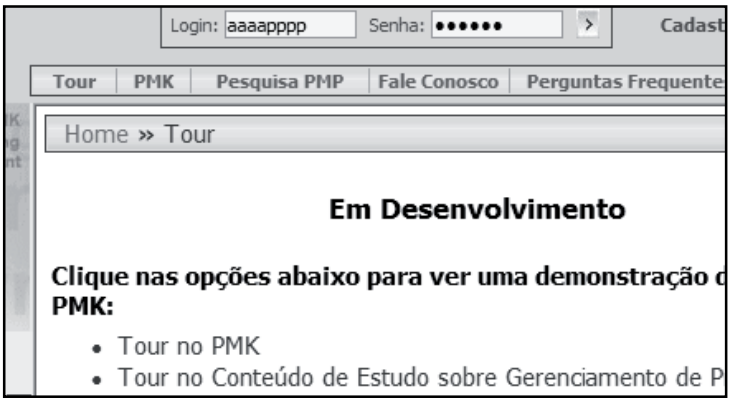

Figura 2. Detalhe de interface do PMK: página em desenvolvimento

No contexto de aplicação da pesquisa nas situações com usuários, este terceiro passou a ser os próprios pesquisadores, que ali permanecíamos no contexto de uso durante a videografia. Situávamos no contexto a fim de observar, videografar e tomar notas do que acontecia. Acontece que, por sermos "outro humano" situados no contexto de uso, esse terceiro buscado pelo usuário, representante de um humano que pudesse responder ativamente e intencionalmente aos seus apelos, éramos nós mesmos.

Ao acompanhar o processo de desenvolvimento de um software, é nítido o quanto o endereçamento para o usuário é fundamental para que sejam organizados os elementos na tela. No momento de uso, por seu turno, é também para um outro que o

usuário direciona seus atos. Acima, vemos que esse outro pode ainda vir situado em outro tempo e espaço – visto que há momentos em que usuários recorrem a outros ou por e-mail, ou em espaços que permitem interação síncrona, como salas de bate-papo virtual ou mesmo telefones. Curiosamente, este "outro" pode ser até um outro software ou site, como acontecia em pesquisas através de enciclopédias virtuais e sites de busca. Mas aí, também, a possibilidade de ruptura pode acontecer já que a atividade cognitiva está a se realizar entre um humano e um outro incapaz de construir intencionalmente o contexto: um sistema informatizado.

Pensemos sobre as caixas de diálogos. Estas são transposições para uma interface computacional de algumas regularidades que configuram os diálogos verbais, como pares adjacentes (pergunta-resposta; convite-aceitação ou recusa; etc.). Porém, este gênero sofre transformações de modo que as ações responsivas que se seguem às caixas de diálogos são realizadas por usuários em uma relação de uso e não de interlocução face-a-face. Assim, esses novos procedimentos de construção do todo do enunciado, seu acabamento e a pressuposição do outro, renova o gênero antecedente.

Pensando nas múltiplas possibilidades discursivas de um software, muito pode ser melhorado. Como exemplo, se pensarmos em desenvolvedores atentos às práticas escolares, observando suas atividades e colocando-se na posição de um outro situado no contexto, um gênero discursivo particular pode emergir: a conversação face-a-face, a qual servirá aos fins de design, que pode aperfeiçoar os limites da interface através de elementos implementados na mesma, ou pode ainda se transformar em um outro gênero na interface – as caixas de diálogos. As ações esperadas e mobilizadas diante de um ou outro gênero são diferentes. Enunciados em caixa de diálogos, os acabamentos podem se tornar mais precisos se houver a participação do desenvolvedor no contexto de uso de protótipos. Assim posicionado, lhe é garantido um "excedente de visão" com relação ao outro-social/usuário, o que lhe permite saber do outro o que ele não sabe de si e, por outro lado, depende desse outro para saber de si.

4. Considerações finais

Sugerimos que um modelo pode ser adaptado para o processo de desenvolvimento de software educativo, onde ao longo do processo, antes, durante e depois do desenvolvimento de um primeiro protótipo, o próprio desenvolvedor (talvez na figura do designer) possa ocupar o lugar do outro humano, inserindo-se nas práticas de sala de aula para delas construir uma visão. Assim, poderá o desenvolvedor capturar, do momento de uso, as situações em que há rupturas no fluxo discursivo. Estas rupturas, como vimos, apontam para descontinuidades entre as ações dos alunos e os procedimentos implementados no sistema (interface e programação). Estes são momentos em que o aluno coloca o software no lugar do *ele,* passando o *eu* (usuário) a clamar por um outro humano capaz de compartilhar o contexto e orientar-se pelas regularidades dos gêneros discursivos que emergem nesses contextos. Assim, poderá o desenvolvedor captar momentos em que a inserção de sua voz a partir de enunciados específicos, tornará mais responsivo o software. Isso porque os enunciados possuem contornos específicos relacionados aos gêneros dos quais emergem. Sendo assim, a compreensão dos gêneros discursivos próprios às práticas de sala de aula, em disciplinas específicas, permitirá a organização dos elementos nas interfaces de aplicações educacionais, tendo por base os contornos particulares que as práticas efetivam.

O desenvolvedor, que ali está na relação *eu-tu* com o usuário, poderá transpor aspectos desta relação para a interface, incluindo contornos discursivos mais apropriados na forma de *feedbacks* para os *inputs*, ou em reposta às ações dos usuários, nos termos bakhtinianos. O design de interface não é, portanto, um problema de simular a comunicação a partir da crença cibernética de "caixas de diálogo" (emissão-recepção de mensagens), mas de propor uma prática de desenvolvimento (em engenharia e design) capaz de redescrever (para o ambiente de software) as propriedades situadas da interação humana. Dessa forma, ao invés de *intencionalidade*, por exemplo, passamos a enfatizar as regularidades sugeridas pelos contextos da ação, e que orientam nosso trânsito entre variados gêneros discursivos. Estas regulações são recursos para a ação e não estruturas internas que possam ser modeladas. Tais recursos devem ser entendidos a partir das práticas discursivas de sujeitos em situações específicas e isso implica investigar contextualmente os processos de produção de sentidos.

Desenvolvedores e usuários são, ambos, autores. Interfaces, muitas vezes tidas como um código semiótico, não estão concluídas e apenas esperam, em seu vir a ser constante, a resposta de um outro. Por isso mesmo preferimos tratá-las não como códigos, mas como enunciados –no sentido bakhtiniano– que carregam em si uma dinâmica dialógica.

O desenvolvimento de softwares está inserido em uma rede na qual se entrecruzam softwares diversos. Nisso, o contexto de uso de softwares pelos desenvolvedores os tornam também usuários, e os enunciados encapsulados nos softwares que usam influenciarão o processo de desenvolvimento. Muitas vezes, esses enunciados são transpostos de forma muito similar do software que usam ao software em desenvolvimento. Mas, dadas as especificidades dos gêneros discursivos, a construção dos enunciados deveria respeitar as práticas em que os usuários estão engajados discursivamente, para que aquilo que lhe seja dado a ver na tela, escutar, ou qualquer ação responsiva, seja um enunciado lingüístico com contornos mais específicos aos seus específicos contextos de uso, os quais podem diferir, em muitos aspectos, dos contextos dos desenvolvedores. Mas ora, como verificamos, os acabamentos de estilo requerem sempre um outro para quem é endereçada a obra, no caso, o software. Assim, as facilidades de resposta que permitam aos usuários regular e corrigir suas interpretações requerem, dos desenvolvedores, conhecimento sobre as práticas e contextos de uso, a fim de interpretarem possíveis ações dos usuários, já que estes transitam entre diferentes gêneros que passam a lhes constituir em suas ações discursivas.

O desenvolvimento de software educacional ou as atividades de pessoas que pensam e projetam sistemas de informação com fins de aprendizagem de conceitos científicos pode melhorar se estas práticas passarem a: entender as práticas sociais de alunos e professores, e os gêneros discursivos que mobilizam as ações sociais nestas práticas; realizar um trabalho de "tradução" de tais gêneros para as interfaces que desenvolvem; permitir situações de compartilhamento e troca de informações não apenas entre os integrantes das equipes de desenvolvimento, mas também entre alunos e professores; participar dos contextos de uso dos protótipos a fim de capturar formas responsivas mais produtivas, considerando que alguns gêneros requerem, mais que outros, a co-participação de um outro enunciador.

Podemos concluir, de tudo o que acompanhamos neste trabalho, que o dialogismo pode oferecer algumas diretrizes para o processo de desenvolvimento de software. Reforçamos o argumento para o necessário engajamento e relacionamento de alunos e professores nos momentos de desenvolvimento, sendo esse engajamento realizado não apenas no trânsito dos usuários para as "fábricas de software", mas também o contrário. Isso porque os desenvolvedores de softwares devem cada vez mais buscar um engajamento nas práticas escolares para as quais será endereçado o software, pois os gêneros que orientam os tipos de enunciados podem ser transpostos para a interface que está sendo desenvolvida. Pensamos que uma metodologia baseada no dialogismo pode levar à transformação e emergência de gêneros discursivos, transformando a própria prática de desenvolvedores, alunos e professores.

Referências Bibliográficas

Aaen, I., Botcher, P., and Mathiassen, L. (1997). The software factory: Contributions and illusions. In: *Proceedings of the Twentieth Information Systems Research Seminar*. Scandinavia, Oslo.

Bakhtin, M. (1978). *Marxismo e filosofia da linguagem*. São Paulo: Hucitec.

Bakhtin, M. (2003). *Estética da criação verbal*. São Paulo: Martins Fontes.

Bakhtin, M. (2002). *Problemas da poética de Dostoievski*. Rio de Janeiro: Forense.

Beyer, H and Holzblatt, K. (1998). *Contextual design: defining custumer-centered systems*. San Francisco: Morgan Kraufman.

Bodker, S. (1991). *Through the interface: A human activity approach to user interface design*. Hillsdale, NJ: Lawrence Erlbraum.

Bodker, S. and Gronbæk, K. (1996). Users and designers in mutual activity: An analysis of cooperative activities in system designs. Y. Engeström and D. Middleton (Eds.), *Cognition and communication at work.* Cambridge: Cambridge University Press.

Brown, J. S. and Duguid, P. (1996). *Keeping it simple*. In: T. Winograd (Ed.), *Bringing design to software.* ACM Press.

Buchenau, M. (2000). Experience prototyping. *ACMI 2000. Proceedings of DIS 2000*. New York: Brooklyn.

Chin, G. and Rosson, M. B. (1998). Progressive design: Staged evolution of scenarios in the design of a collaborative science learning enviroment. *CHI 98,* Los Angeles.

Cole, M. and Engestrom, Y. (1993). A cultural-historical approach to distributed cognition. In: Salomon, G. (Ed.), *Distributed cognition: Psychological and educational considerations.* Cambridge: Cambridge University Press.

Crowston, K. and Scozzi, B. (2002). Open source software projects as virtual organizations: Competency rallying for software development. *IEE Proceedings Software*, 149(1):3-17.

De Souza, C.S. (2005). *The semiotic engineering of human-computer interaction*. Cambridge: MIT Press.

De Souza, C.S. (1993). The semiotic engineering of user-interface languages. *International journal of man-Machine Studies*, 39, 753-773.

Engestrom, Y. and Middleton, D. (Eds.) (1996). *Cognition and Communication at work*. Cambridge: Cambridge University Press.

Engeström, Y. and Escalante, V. (1996). Mundane tool or object of affection? The rise and fall of the postal buddy. In: B. Nardi (Ed.), *Context and consciousness: Activity Theory and Human-Computer Interaction (pp. 325-373)*. Cambridge: MIT Press.

Goodwin, C. (2000). Action and embodiment within situated human interaction. *Journal of Pragmatics*, 32, 1489-1522.

Heyes, P. and Reddy, D. (1983). Steps towards graceful interaction in spoken and written man-machine communication. *International Journal of Man-Machine Studies*,19, 231-84.

Hutchins, E. (1990). The social organization of distributed cognition. In: J. Levine and S. Teasley (Eds.), *Perspectives on socially shared cognition*. Washington, DC: American Psychological Association.

Jordan, B. and Henderson, A. (1995). Interaction Analysis: Foundations and practice. *The Journal of the Learning Sciences*, 4(1):39-103.

Leontiev, A. (2004). *O desenvolvimento do psiquismo.* São Paulo: Centauro.

Marcuschi, L. A. (1991). *Análise da Conversação.* São Paulo: Ed. Ática.

Marcuschi, L. A. (2004). Gêneros textuais emergentes no contexto da tecnologia digital. In: L. A. Marcuschi e A. Xavier (Orgs.), *Hipertexto e gêneros discursivos.* Rio de Janeiro: Lucernas.

Päivärinta, T. and Tyrväinen, P. (1998). Documents in information management: Diverging connotations of 'a Document' in digital era. In: *Proceedings of IRMA '98*, Boston, pp. 163-173

Spinuzzi, C. (2002). A Scandinavian challenge, a US response: methodological assumptions in Scandinavian and US prototyping approaches. In: *Proceedings of the 20th Annual international Conference on Computer Documentation*, New York, NY, 208-215.

Suchman, L (1987). *Plans and situated actions: The problem of human-machine communication*. Cambridge: Cambridge University Press.

Torreão, P. (2005). Project management knowledge learning environment: ambiente inteligente de aprendizado para educação em gerenciamento de projetos. *Dissertação de mestrado.* Recife: Centro de Informática, Universidade Federal de Pernambuco.

Van der Ver, R. e Valsiner, J. (1999). *Vygotsky: Uma síntese*. São Paulo: Loyola.

Vigotski, L. S. (2001). *A construção do pensamento e da linguagem*. São Paulo: Martins Fontes.

Woodruff, A., Szymanski, M., Grinter, R., and Aoki, P. (2002). Practical strategies for integrating a conversation analyst in an interactive design process. *Proceedings of the Conference on Designing Interactive Systems (pp. 255-264)*: ACM Press.