1
Introdução

Sistemas interativos vêm cada vez mais se tornando populares e ubíquos, com o crescimento do uso da Internet e de dispositivos móveis. Se considerarmos o ponto de vista dos usuários, esses sistemas são percebidos através daquilo que eles podem diretamente ver e experimentar [Hix e Hartson, 1993], e portanto o sucesso (ou destaque) de um sistema está fortemente relacionado à qualidade da interação que é proporcionada por ele.

Assim, é crescente a preocupação por parte da indústria de software em oferecer aos usuários uma experiência interativa de qualidade. Porém, dentre muitos motivos, a falta de notações e ferramentas para apoiar a modelagem da interação do usuário com o sistema é bastante evidente [Seffah et al., 2005]. Esta dissertação endereça este problema, utilizando uma modelagem híbrida que representa tanto a interação humano-computador quanto a interface apresentada pelo sistema. Com isso, procuramos tornar claras as decisões de design tomadas ao se projetar um sistema interativo, explicitando seu comportamento, o modo como ele se apresentará e como usuário e sistema se comunicam, segundo a visão do designer[1].

Em seu livro *The Design of Everyday Things* [Norman, 1988], Don Norman diz que todos nós construímos modelos mentais para interagir com os objetos do dia-a-dia como carros, por exemplo, mesmo que não conheçamos o funcionamento da transmissão ou do motor. Simplesmente mantemos o nosso modelo mental em que girar o volante em sentido horário faz o carro ir para a direita, pisar o pedal do centro faz frear o carro, e assim por diante. Enquanto houver relação perceptível entre nossas ações e as reações advindas do artefato que usamos, tomamos nosso modelo como válido e confiamos nele – os modelos do sistema e do usuário são diferentes, porém **compatíveis** entre si. Assim, quando algo de errado acontece, o modelo mental que construímos é do que dispomos imediatamente para resolver a situação.

[1]Chamaremos de *designer* o projetista do software, também chamado de arquiteto de informação, user experience analyst, entre outros. Ou seja, a figura envolvida na concepção da solução interativa, e não necessariamente apenas quem o concebeu ou codificou.

Segundo Norman, um *design* eficiente deve propiciar que o usuário construa um modelo **compatível** com o modelo vislumbrado pelo designer, de modo que o sistema transmita claramente o seu estado, dada uma ação do usuário. No entanto, por se tratarem de artefatos intelectuais (e não físicos), softwares precisam que tanto o produtor quanto o consumidor usem a mesma linguagem para se comunicarem [de Souza, 2005].

Em outras palavras, isso significa que proporcionar uma interação eficiente implica a **comunicação** eficiente sobre como o usuário pode operar o software para atingir seus objetivos. Atingir isso não se trata meramente de aprendizado ou treino, mas de usuário e sistema "se entenderem" usando a mesma linguagem.

1.1 Motivação

A área de Engenharia de Software fez muito avanços ao longo do tempo em diversos aspectos da arquitetura dos sistemas computacionais, desde métodos mais rápidos para codificação e entrega de sistemas a estruturas mais eficientes e robustas para suportar a execução de aplicações.

Porém, ao considerarmos o projeto de sistemas orientado a objetos com *Unified Modeling Language* (UML) [OMG, 2007], um conjunto de linguagens amplamente adotado tanto pela indústria quanto pela academia, observamos que, do ponto de vista da Interação Humano-Computador (IHC), ela falha em prover recursos adequados para representar o comportamento do sistema, como ele será percebido ou experimentado pelo usuário.

Na UML, o componente humano geralmente é considerado como a "fronteira" do sistema, através dos diagramas de caso de uso, possivelmente por ser considerado imprevisível ou complexo demais para ser propriamente **modelado**. Não obstante é tratado pelos projetistas como uma componente que deve ser **treinada**, uma vez que o sistema esteja pronto. Como apontou Seffah (2005), é comum se encontrar termos como *usabilidade* como mais um item na lista de requisitos não-funcionais.

Porém, esta mentalidade vem mudando gradativamente com a atenção da indústria voltando-se para tópicos como *User-Centered Design* [Norman e Draper, 1986], que coloca as necessidades dos usuários como ponto central no projeto de soluções computacionais. A preocupação é crescente, dada a competitividade, principalmente na Internet, entre os diferentes sistemas que possuem propósitos semelhantes, mas proporcionam experiências

interativas diferentes, a exemplo de sistemas de correio eletrônico e transações bancárias, comuns hoje em dia.

Tradicionalmente, a **prática** em IHC tem considerado a utilização de cenários, *storyboards* e esboços, como propuseram [Carroll, 1995] e [Buxton, 2007]. Por ser de entendimento fácil, elas são utilizadas em muitas etapas, ajudando tanto na fase de elicitação de requisitos quanto em decisões acerca da interface gráfica concreta com a qual o usuário irá lidar. No entanto, uma vez que estas abordagens são informais, elas são imprecisas e frequentemente ambíguas [Seffah et al., 2005].

Algumas **pesquisas** em IHC caminham para abordagens mais formais ao projeto, abstraindo características que sejam consideradas relevantes em modelos. Um modelo pode ser definido como uma abstração de um sistema físico com um determinado propósito, considerando diferentes aspectos de um problema e colaborando para uma descrição limitada, porém confiável, da realidade [Hoover et al., 1991].

Podem ser citados alguns trabalhos que propõem arquiteturas orientadas a modelos para o projeto de interação e interfaces tais como o CAMELEON [Calvary et al., 2003], MECANO [Puerta, 1996] e WISDOM [Nunes e Cunha, 2001]. Dentre os modelos que se utilizam para representar a interação entre usuários e sistemas, o mais amplamente adotado é o CTT (ConcurrentTaskTrees) [Paternò, 2000], que é baseado em modelagem de tarefas, onde é feita a decomposição hierárquica de tarefas, realizada pelo usuário, pelo sistema ou por ambos. Por sua vez, a modelagem de tarefas é baseada em análise de tarefas [Diaper e Stanton, 2003], que foca a **performance** das tarefas executadas por seres humanos, mais do que a qualidade da experiência interativa.

Além disso, apesar de a modelagem de tarefas abordar o problema da interação com mais ênfase no usuário do que faz a UML, por exemplo, ela ainda não representa de forma clara aspectos importantes da interação tal como ela é percebida pelo usuário. Eventos importantes como a recuperação quando ocorre algum erro (quando um usuário recebe mensagens do sistema informando que algo não funcionou) não são claramente definidos – ou seja, a modelagem de tarefas especifica o comportamento como ele **deveria idealmente** ocorrer e não como ele **pode** ocorrer.

Assim, visando preencher as lacunas de outras modelagens e enriquecer a representação da interação, vem sendo desenvolvida uma linguagem visual para modelagem chamada MoLIC, "Modeling Language for Interaction as Conversation" [Paula, 2003; Silva, 2005; Araujo, 2008]. A linguagem é fundamentada na

Engenharia Semiótica [de Souza, 2005] e segue a metáfora de interação como conversa. Ela provê um mapa para a solução conceitual da interação projetada para uma aplicação, representando seu comportamento da forma que ele será comunicado pelo designer e percebido pelos usuários.

1.2 Objetivo

Este trabalho visa propor um uso estendido do diagrama de interação da MoLIC, na tentativa de aproximar a modelagem conceitual da interação a uma visão mais concreta do sistema. Mais especificamente, este trabalho estende a modelagem do diálogo usuário-designer com esboços de interface nela integrados.

A ideia de estender a MoLIC serve ao propósito de utilizar os esboços de tela para representar a interface preliminar em protótipos que tenham a interação guiada por um modelo. Ao mesmo tempo, o designer pode ter uma ferramenta em mãos que possibilite, iterativamente: modelar a **visão geral** do sistema a nível de comportamento; rascunhar a **representação visual** desse sistema; gerar **protótipos** para efetuar testes com os usuários de forma barata e, como consequência das observações, criticar e melhorar gradativamente a sua solução inicial.

Dessa forma, apresentamos uma abordagem que propicia o projeto da **interação** e também permite uma representação aproximada, porém concreta, em termos de **interface**, se o designer julgar necessário. Um estudo de caso exploratório foi conduzido visando avaliar a expressão da linguagem estendida com esboços de tela e investigar possíveis direções no futuro dessa abordagem.

Foi desenvolvida uma ferramenta para permitir tanto a elaboração gráfica de diagramas MoLIC quanto o aproveitamento de um modelo processável da linguagem – através do qual pode ser possível validar sintaticamente o diagrama, relacionar seus elementos a outros modelos, visando a interoperabilidade com outras ferramentas que porventura venham a ser desenvolvidas. Por fim, em consonância com os trabalhos de [Paula, 2003; Silva, 2005; Araujo, 2008], acreditamos que dispor de uma ferramenta computacional ajudará a popularizar a linguagem e como consequência também a Engenharia Semiótica entre pesquisadores e praticantes em IHC. Acreditamos ainda que isso venha a contribuir para a realização de um maior número de estudos de caso, o que poderá levantar novas questões de pesquisa sobre a MoLIC e sobre a Engenharia Semiótica.

1.3 Organização da Dissertação

Este trabalho está assim organizado: o Capítulo 2 apresenta os trabalhos que se relacionam com este, como linguagens utilizadas para modelagem de interação e também ferramentas que utilizam conceitos de *storyboarding* para modelar interação. No Capítulo 3 é apresentada a linguagem MoLIC e em seguida, no Capítulo 4, é apresentada a proposta de extensão da linguagem em sua integração com técnicas de *storyboarding*, numa abordagem híbrida para modelagem de interação e da visão geral da interface. No Capítulo 5 é apresentado o estudo de caso que envolve a investigação da viabilidade da integração da MoLIC com esboços de interface. No Capítulo 6 é apresentada a ferramenta desenvolvida para possibilitar a construção de diagramas. E por fim, o Capítulo 7 contém as considerações finais.