Comunicação Através de Diagramas de Casos de Uso no Desenvolvimento de Software – Uma Breve Análise de Sentido[1]

Carlos Eduardo Marquioni[2], M.Sc., PMP

Resumo:

O presente trabalho apresenta conceitos do diagrama de casos de uso, utilizado no desenvolvimento de software: trata-se de notação utilizada mundialmente durante o desenvolvimento de software para documentar necessidades dos usuários. A partir dos conceitos da notação é realizada uma reflexão do potencial comunicacional proporcionado pelo diagrama em questão, baseada principalmente na diferença de repertório dos envolvidos no processo de criação do artefato (técnicos) e dos leitores do artefato (não técnicos), utilizando como referencial teórico principal o estudo de diagramas da semiótica de C. S. Peirce e reflexões de percepção formuladas por Umberto Eco. Finalmente, é proposta a utilização do diagrama em conjunto com o design de interfaces como alternativa para facilitar a formação de sentido.

Palavras-chave: comunicação; semiose; requisito; software; diagrama.

[1] Trabalho apresentado ao II CONECO (Congresso de Estudantes de Pós-graduação em Comunicação) – GT05 (Tecnologias e estéticas da comunicação) realizado na PUC-Rio em Novembro/2007.

[2] marquioni@marquioni.com.br – Mestre em Comunicação e Linguagens (UTP/2008) e Bacharel em Análise de Sistemas (PUC-Campinas/1994).

1 INTRODUÇÃO[3]

Produtos de software são criados a partir de requisitos, que

> [...] são definidos durante os estágios iniciais do desenvolvimento de sistemas como uma especificação de o que deve ser construído. Eles são as descrições do comportamento do sistema[4], informações do domínio da aplicação, regras da operação do sistema ou especificações de uma propriedade ou atributo de um sistema (KOTONYA; SOMMERVILLE, 1998, p. 6).

Uma vez que "toda comunicação se realiza na medida em que a mensagem é decodificada com base num código preestabelecido, comum ao remetente e ao destinatário" (ECO, 2005, p. 305), desde o final da década de 1970 teóricos de software propõem e divulgam mundialmente – entre os produtores de software – notações[5] visuais para descrever requisitos, com o objetivo de tentar estabelecer comunicação eficiente entre técnicos de software e usuários[6]. Estas notações têm um papel determinante no processo de software, pois auxiliam na definição do sistema correto a ser construído antes de iniciar a construção dos programas propriamente ditos: "alcançar boa comunicação, associada a um bom entendimento do mundo dos usuários, é a chave para desenvolver bom software" (FOWLER; SCOTT, 1999, p. 09). Tecnicamente, o que ocorre ao longo de todo o

[3] Todas as traduções apresentadas no texto são de minha autoria. As citações a Charles Sanders Peirce referenciam os chamados Collected Papers (CP) quanto a seus volumes e parágrafos. Assim, quando apresentado, por exemplo, CP 2.228, trata-se de uma citação extraída dos Collected Papers, volume 2 da edição americana, parágrafo 228.

[4] Nota do autor: *sistema* deve ser entendido ao longo de todo o trabalho no sentido de sistema de software (ou sistema de computação); isto significa que os termos sistema e software são considerados sinônimos neste trabalho. A ressalva é importante para evitar confusão com a definição de sistema da Engenharia de Sistemas, que "[...] cobre o desenvolvimento de sistemas totais, que podem ou não incluir software" (CHRISSIS; KONRAD; SHRUM, 2003, p. 7).

[5] Para notações de software recomendadas no final da década de 1970, consulte: GANE, Chris; SARSON, Trish. **Análise Estruturada de Sistemas**. Rio de Janeiro: LTC – Livros Técnicos e Científicos Editora S.A., 1983; para notações de software recomendadas a partir de meados da década de 1980, consulte: YOURDON, Edward. **Modern Structured Analysis**. New Jersey: Prentice Hall, 1989.

[6] Podem ser considerados dois tipos de usuário. O primeiro deles é aquele que, durante o desenvolvimento do software, supostamente detém o conhecimento acerca do processo que será objeto de automação. O segundo tipo de usuário simplesmente utiliza um sistema já pronto, sem necessariamente ter envolvimento durante a concepção do sistema. Este segundo tipo de usuário normalmente é referenciado pela indústria de software como *usuário final*. A preocupação deste trabalho é em relação ao primeiro tipo de usuários.

desenvolvimento é o reescrever constante dos requisitos em função de seu leitor: em última instância, os programas são os requisitos redigidos em um *idioma* para leitura por máquinas (a linguagem de programação). A escolha de notações visuais nos estágios iniciais do processo de software é justificada pela mesma razão que se criam as plantas baixas na construção civil:

> Assuma por um momento que lhe foi solicitado que especifique todos os requisitos para a construção de uma cozinha profissional. Você conhece as dimensões do local, a localização das portas e janelas e o espaço disponível de paredes. [...] Para uma especificação completa você precisaria de um modelo significativo da cozinha [...] que mostre [...] a inter-relação entre [as partes desta cozinha]. A partir desse modelo, seria relativamente mais fácil avaliar a eficiência do fluxo de trabalho (um requisito de toda cozinha) e a aparência estética do local (um requisito pessoal, mas muito importante) (PRESSMAN, 2000, p. 255).

Em meados dos anos 1990[7], foi difundido mundialmente um conjunto de notações em diagramas para representação gráfica de requisitos de software que foi chamado de

> [...] **Linguagem de Modelagem Unificada (UML)** [...] [e que unificou] os métodos de [três teóricos de software:] Booch, Rumbaugh [...] e Jacobson [...]. A UML é uma linguagem de modelagem, não um método. A **linguagem de modelagem** é a (basicamente gráfica) notação que os métodos utilizam para formalizar *designs* [...] [e] é certamente um aspecto chave para comunicação. Se você deseja discutir seu *design* com alguém, é a linguagem de modelagem que ambos necessitam entender [grifos no original] (FOWLER; SCOTT, 1999, p. 01).

O *design* a que se referem os autores não é o *design* de interfaces, mas o *design* arquitetural de software: trata-se do equivalente à *planta baixa* da cozinha do exemplo anterior, aplicado à construção de programas. A notação UML definiu o conteúdo gráfico, os metadados e a sintaxe a utilizar para criar estas *plantas baixas* de software como um conjunto de diagramas (também chamados de

[7] Conforme comentado em nota de rodapé anterior, nos anos 1970 foi utilizada a notação da Análise Estruturada para criar modelos de software conceituais; nos anos 1980 foi adotada a notação da Análise Estruturada Moderna; nos anos 1990, surge a UML.

artefatos); um dos artefatos proposto é o Diagrama de Casos de Uso. Este diagrama tem como objetivo auxiliar "o cliente, os usuários e os desenvolvedores a obterem acordo no modo de uso do sistema" (JACOBSON; BOOCH; RUMBAUGH, 2001, p. 40). Trata-se da primeira transcrição técnica dos requisitos, um dos primeiros produtos criados para desenvolver ou atualizar software e modela "as ações de nossos usuários, e as respostas associadas do sistema [...] [, uma vez que] o que nós necessitamos que o software faça depende de como os usuários o acessam e o que esses usuários tentam fazer" (ROSENBERG; SCOTT, 2001, p. 35). Em um momento inicial, os casos de uso deveriam estabelecer comunicação entre os técnicos de software e os usuários; uma vez estabelecido acordo quanto ao comportamento esperado, estes diagramas seriam utilizados também para comunicação entre técnicos: os casos de uso constituem a referência primária para as demais abstrações – ou decodificações – dos requisitos nos artefatos propostos pela UML, até se chegar à linguagem de programação. Para representar o comportamento do produto e estabelecer comunicação, tanto entre técnicos e usuários, quanto entre os representantes do próprio corpo técnico, a notação do diagrama de casos de uso faz uso das convenções apresentadas a seguir.

1.1 O ATOR

O ator é um usuário externo ao sistema; ele não é parte do software e pode ser

> [...] qualquer coisa que tenha interface com seu sistema – por exemplo, pessoas, outros softwares, dispositivos de hardware, depósitos de dados ou redes. Cada ator define um papel específico. Cada entidade externa a seu sistema pode ser representada por um ou mais atores. Com isso, um indivíduo pode ser representado por muitos atores porque esse indivíduo assume vários papéis na utilização do sistema. Ou muitos indivíduos podem ser representados por um único ator porque eles todos assumem o mesmo papel em relação ao sistema (SCHNEIDER; WINTERS, 2001, p. 12).

Os atores trocam informações com o sistema enviando e recebendo mensagens através de uma associação de comunicação (descrita a seguir). O ator é representado segundo a notação apresentada na figura 1.

1.2 O CASO DE USO

Outra convenção do diagrama de casos de uso é o objeto caso de uso. Cada objeto caso de uso é representado graficamente com o símbolo de uma elipse e corresponde a

> [...] um comportamento do sistema que produz um resultado de valor mensurável para um ator. Casos de uso descrevem as coisas que os atores querem que o sistema faça [...] [e] deve ser uma tarefa completa, segundo a perspectiva do ator. [...] O conjunto de todos os casos de uso irá descrever a funcionalidade completa do sistema sob o ponto de vista dos usuários (SCHNEIDER; WINTERS, 2001, p. 14).

Tipicamente, quando o usuário do caso de uso for um ator humano e o sistema em questão fizer uso de ambiente visual – como o MS Windows ou o Linux –, o caso de uso será construído em termos de programação através de uma interface do tipo tela gráfica, pois é através dessas interfaces que os usuários tipicamente interagem com o sistema computacional. É possível então estabelecer um elo entre o *design* de interface e os casos de uso: este elo é debatido na terceira parte deste trabalho.

1.3 ASSOCIAÇÕES DE COMUNICAÇÃO

A representação gráfica das interações entre os atores e os casos de uso é convencionada pela UML através de linhas chamadas associações de comunicação, ou simplesmente associações. É importante destacar que cada associação entre um ator e um caso de uso representa todo o diálogo que se dá entre eles, até que um resultado de valor observável seja alcançado. Com isso, mesmo que haja várias interações entre o usuário e o sistema, uma linha entre o

ator e o caso de uso é suficiente para representar todas essas interações. Formalmente, as "associações mostram com quais atores o caso de uso se comunica [...]. A associação deve sempre ser binária, implicando um diálogo entre o ator e o sistema" (ERIKSSON *et al*, 2004, p. 66). A figura 2 representa dois atores associados a um caso de uso.

1.4 METADADOS EM CASOS DE USO

Uma vez que o comportamento esperado do software normalmente é complexo, há textos explicativos associados aos diagramas: casos de uso fazem uso de recursos gráficos e textuais para representar o comportamento do sistema. Estes textos são chamados tecnicamente de especificação, e sua estrutura de escrita define que

> [...] a narrativa de um caso de uso deve ser orientada à resposta de eventos, como, 'O sistema faz isso quando o usuário faz aquilo' [...] [logo,] o caso de uso deve capturar [e especificar] o que acontece 'na retaguarda' [após cada ação de interação do usuário com o produto de software] (ROSENBERG; SCOTT, 2001, p. 39).

O detalhamento textual dos casos de uso se dá através do conceito de fluxo de eventos, que "é uma série de estruturas declarativas que listam os passos de um caso de uso sob o ponto de vista do ator" (SCHNEIDER; WINTERS, 2001, p. 29). O fluxo de eventos é "dividido em duas seções: o caminho básico e os caminhos alternativos. [...] [O primeiro lista] [...] a seqüência de passos mais comum [enquanto os segundos listam] [...] alternativas de execução e exceções" (*id.*, p. 35). O anexo 1 contém um exemplo de especificação de um fluxo de eventos e a figura 2 é o diagrama de caso de uso correspondente.

2 A ANÁLISE DA NOTAÇÃO DE CASOS DE USO

"Não se cria uma imagem fiel do nada"
(GOMBRICH, 1995, p. 89)

2.1 O DIAGRAMA

Sob uma perspectiva comunicacional, o diagrama de casos de uso pode ser descrito como "um sistema de símbolos que, por convenção preestabelecida, se destina a representar e transmitir uma mensagem entre a fonte e o ponto de destino" (PIGNATARI, 2002, p. 23). A fonte seria o técnico de software e o ponto de destino um usuário ou um outro técnico de software. Utilizando definições da semiótica de Charles Sanders Peirce em relação a diagramas, casos de uso deveriam permitir uma "visão clara do modo de conexão de suas partes, e de suas composições em cada estágio de nossas operações sobre ele [...] [possibilitando que] cada pequeno passo do processo apareça distintamente, de modo que sua natureza possa ser entendida" (CP 4.533). Indubitavelmente os diagramas em questão possibilitam uma visão detalhada do comportamento do software. Contudo, sua eficiência no estabelecimento de diálogo entre técnicos e usuários pode ser objeto de análise, embora os autores da UML e do diagrama de casos de uso argumentem que

> [...] há muitas razões pelas quais os casos de uso são bons, se tornaram populares e universalmente adotados [...] [e uma dessas razões seria o fato que] eles oferecem um meio sistemático e *intuitivo* [grifo meu] de capturar os **requisitos funcionais** [grifo no original] [...] com foco no valor agregado ao usuário (JACOBSON; BOOCH; RUMBAUGH, 2001, p. 37).

Este suposto caráter intuitivo seria reforçado pela argumento que "usuários e clientes não necessitam aprender uma notação complexa [...] [e que a] linguagem natural pode ser utilizada na maior parte do tempo, *o que facilita a leitura das descrições* [grifo meu] dos casos de uso e propostas de mudanças" (JACOBSON; BOOCH; RUMBAUGH, 2001, p. 37). Outros teóricos da Engenharia de Software adotam uma abordagem mais cautelosa e dizem que

> [...] casos de uso são uma ferramenta de comunicação. Eles são efetivos apenas quando comunicam informação sobre a forma de funcionamento do sistema ao leitor. É importante considerar quem lerá os casos de uso [...]

> [pois os leitores do diagrama] devem ser capazes de entendê-los. Se isso não ocorrer, os casos de uso necessitam ser reescritos (SCHNEIDER; WINTERS, 2001, p. 32).

Esta segunda visão parece mais adequada em termos teóricos de comunicação, e evidencia a possibilidade de erros decorrentes de problemas de leitura dos diagramas. Umberto Eco (2005, p. 36) relata que um diagrama é uma representação simplificada, que nasce de um ponto de vista[8]. No caso dos diagramas de caso de uso, esta simplificação se faz considerando o aspecto da automação de atividades executadas no ambiente em que o software é posto em operação. O ponto de vista utilizado na elaboração dos diagramas foca aspectos técnicos associados à produção de software. Estes aspectos podem ultrapassar os limites do conhecimento do usuário, ou mesmo não serem pertinentes ou relevantes para este usuário. Em outros termos, a leitura pode não ser intuitiva devido às variações de repertório entre os envolvidos no processo de leitura destas abstrações. A percepção do leitor "*pode ser vista como um fato de comunicação*, como um processo que só se gera quando, com base em aprendizagem, conferiu significado a determinados estímulos e não a outros" (ECO, 2005, p. 101-102): se o leitor do diagrama não tiver um repertório que permita leitura correta dos aspectos representados, a comunicação pode ser imprecisa.

> A expressão visual significa muitas coisas, em muitas circunstâncias e para muitas pessoas. É produto de uma inteligência humana de enorme complexidade, da qual temos, infelizmente, uma compreensão muito rudimentar" (DONDIS, 2003, p. 2);
> [...] o *input* visual é fortemente afetado pelo tipo de necessidade que motiva a investigação visual, e também pelo estado mental ou humor do sujeito. Vemos aquilo que precisamos ver (*id.*; 133).

[8] O autor cita como exemplo que se quiser "estudar [...] corpos humanos sob o ângulo da estrutura esquelética, ou sob o ângulo da característica 'animal em posição erecta', ou 'bípede com dois membros superiores e dois inferiores'" (ECO, 2005, p. 36), poderia elaborar um diagrama como aquele da figura 3.

Cabe ainda relação com Umberto Eco, que comenta que a percepção do objeto representado ocorre após seleção de "*códigos de reconhecimento [...] com base em convenções gráficas*" (2005, p. 104). O leitor *entende* a partir do momento que possui códigos para entender: "chegamos a entender uma dada solução técnica como a representação de uma experiência natural porque em nós se formou *um sistema de expectativas* codificado" (ECO, 2005, p. 109) – talvez esta *experiência natural* seja o que os autores da notação tenham chamado de leitura intuitiva. Contudo, ela só se dá após estabelecida a convenção com o leitor. Note-se que o caráter intuitivo não precisa se referir necessariamente à fidelidade com a qual a representação se dá, já que a

> [...] representação não é [...] uma réplica [...] [assim como o] teste da imagem não é a sua semelhança com o natural, mas a sua eficácia dentro de um contexto de ação. Ela pode ser semelhante ao natural se isso for considerado como algo que contribui para a sua força, mas em outros contextos *o mais sumário dos esquemas bastará, desde que retenha a natureza eficaz do protótipo* [grifo meu] (GOMBRICH, 1995, p. 117).

O aspecto do protótipo será debatido a seguir, através de associação com o *design* de interfaces. Por ora é importante ter em mente que independente da semelhança que se estabelece entre o conceito representado e a representação propriamente dita, a compreensão do objeto no desenho do diagrama "no mais das vezes faz mais apelo a uma aprendizagem do que a um reconhecimento natural" (FONTANILLE, 2005, p. 103) e

> a comunicação não se define apenas por elementos [...] que determinem a eficiência ou não de seus processos [...] [mas] se efetiva de fato como efeitos de sentido interacionais de comunicabilidade ou incomunicabilidade, que dependem do grau de domínio dos sistemas simbólicos e das regras multivariáveis no jogo das relações sociais e interpessoais (CAETANO, 2002, p. 74).

2.2 A ESPECIFICAÇÃO (METADADOS)

A utilização de texto no detalhamento do diagrama é justificada pelas teorias da comunicação pois o desenho "nem sempre é tão claramente representativo quanto se crê [...] [por isso ele pode ser] *acompanhado de inscrições verbais* [grifo no original]" (ECO, 2005, p. 111). A argumentação dos autores da notação em relação ao uso de descrições em linguagem natural é pertinente em termos, pois pode ser analisada a redação da descrição do diagrama que eventualmente gera problemas comunicacionais em virtude de diferenças de repertório entre o redator e o leitor dos diagramas.

> Requisitos em linguagem natural podem ser ambíguos, obscuros e geralmente confusos. Problemas comuns são que: 1. os requisitos são escritos usando cláusulas condicionais complexas (se A então se B então se C...) que se tornam confusas; 2. a terminologia é usada de forma inconsistente; 3. os redatores do requisito assumem que o leitor tem conhecimento específico do domínio do sistema (KOTONYA; SOMMERVILLE, 1998, p. 19).

2.3 A IMPORTÂNCIA DA COMUNICAÇÃO COM USUÁRIOS

A notação de casos de uso, assim como a UML, foram divulgadas e tiveram ampla aceitação mundial pelos membros da comunidade de software. É importante destacar que para comunicação entre membros do corpo técnico, os casos de uso *podem* gerar comunicação eficiente[9]: o que está em análise a princípio não é a utilidade dos diagramas quando se trata de comunicação entre técnicos, mas entre técnicos e usuários. Enquanto a compreensão pelos usuários do produto que será construído é fundamental, é discutível a pertinência do aprendizado deste tipo de notação (mesmo que apenas para leitura) por pessoas que são apenas usuários de produtos de software, e que estão eventualmente envolvidas em projeto de

[9] Destaque-se que mesmo no caso de comunicação entre técnicos de software a convenção deve estar estabelecida para que ocorra compreensão. Contudo, uma vez estabelecida esta convenção, a comunicação tende a ser facilitada. Isto é justificado pois a forma de especificar casos de uso remete à programação (mas é de leitura muito mais fácil que o código fonte de um programa), tornando-os mais facilmente compreensíveis pelos profissionais habituados com os *idiomas* de software.

desenvolvimento de software como fontes de informação e validação de requisitos, mas não como construtores de programas. Ao invés de sugerir uma nova notação[10], pode ser mais interessante identificar alternativas que permitam estabelecer comunicação também com os usuários a partir daquelas existentes. Uma vez que "a forma denota a função só com base num sistema de expectativas e hábitos adquiridos, e portanto, com base num código" (ECO, 2005, p. 200), trata-se de identificar um "*código conhecido*" (ECO, 2005, p. 201) que seja significativo também para os não técnicos quando se trata de software. É neste sentido que avança esta análise.

3 A METÁFORA DO *DESKTOP*

> "[...] *mais do que modificar as mensagens, ou controlar as fontes de emissão, pode-se alterar um processo comunicativo agindo sobre as circunstâncias em que a mensagem será recebida* [grifo no original]"
> (ECO, 2005, p. 418)

Os diagramas de casos de uso representam a interação entre atores (usuários) e o caso de uso (o aplicativo de software propriamente dito). Quando se tratar de um sistema que utilize interfaces gráficas, a tradução última do caso de uso será em um programa associado a uma tela. Esta mesma tela talvez seja o objeto mais familiar ao usuário, pois é através dela que é estabelecida interação com o software e "qualquer mensagem exige, em primeiro lugar, um contexto [...] ao qual remete; em seguida, exige um código pelo menos em parte comum ao emissário e ao destinatário" (JOLY, 1996, p. 56). Há indícios de que o "modelo perceptivo do objeto" (ECO, 2005, p. 112), no caso de software, seja a tela. Isto permite estabelecer em relação ao protótipo da tela no software uma analogia com

[10] Conforme comentado anteriormente, foram várias as notações sugeridas pelos teóricos de software desde a década de 1970; esta variedade permite identificar como possibilidade de análise avaliar como a não manutenção de uma morfologia pode ter dificultado o aprendizado das notações pelos usuários. Por se tratar de mensagens visuais simbólicas, "alguma educação por parte do público se faz necessária para que a mensagem seja clara [...] [e há que se considerar a] penetração na mente do público para educá-la quanto ao seu significado" (DONDIS, 2003, p. 92).

a maquete na construção civil, que “talvez seja a única forma de fazer com que as pessoas de pouca sensibilidade para a visualização possam ver como uma determinada coisa vai ficar em sua forma definitiva” (DONDIS, 2004, p. 79-80). Estudos interdisciplinares entre as áreas de software e comunicação trabalham aspectos do *design* de interfaces, que incorporou ao ambiente dos computadores a organização espacial do mundo. Para o *design* de interfaces há relevância o trabalho do pesquisador Alan kay, que propôs a metáfora do *desktop* ao (JOHNSON, 2001, p. 39) sugerir a transposição da escrivaninha para a tela do computador. Assim, o usuário teria a impressão que manipulava papéis, com os quais estava habituado: era proposto um ambiente espacial, no qual o indivíduo tinha a sensação de entrar. A partir desta metáfora original da escrivaninha se questionou “por que não fazer [a interface] imitar o velho mundo analógico que iria substituir?” (*id*., p. 40). Os *papéis* de Kay citados por Johnson ficaram conhecidos como *janelas* (*windows*) nos ambientes gráficos de software, as metáforas se expandiram, e talvez as mais conhecidas, além da escrivaninha e das janelas, sejam a do gerenciador de arquivos (o antigo fichário) e a da lixeira (para os papéis que não mais interessam). É interessante notar que o que se desejou desde o início foi tentar minimizar o impacto sentido por quem fosse utilizar a interface: “se você sabia se sentar a uma escrivaninha e revirar papéis, podia usar a máquina” (*ibid*.). Evidentemente se trata de uma simplificação; contudo, o sucesso desta metáfora foi tamanho que os usuários utilizam ambientes visuais gráficos de computação sem grandes problemas.

Segundo esta perspectiva, o design de interface e os diagramas de casos de uso podem ser compreendidos como complementares. Enquanto os diagramas fazem mapeamento gráfico e textual do comportamento das interfaces do produto de software, a tela formaliza estes diagramas de maneira perceptível a leitores não técnicos. Apesar de a comunidade de software utilizar conceitos de design de interface através do envolvimento de designers nas atividades de produção de software, este envolvimento não está necessariamente associado à formalização de

requisitos, mas a aspectos de usabilidade[11]: não há procedimentos formais que relacionem de maneira efetiva e em escala os protótipos do produto à documentação técnica deste produto. É importante notar que a comunicação entre técnicos e não técnicos (e mesmo entre os técnicos) pode melhorar com a criação e especificação de diagramas simultaneamente ao desenvolvimento de protótipos de telas gráficas, baseadas na bem sucedida metáfora do desktop.

Os casos de uso e sua especificação podem ser utilizados pelos técnicos como subsídio adicional para explicar ou validar junto aos usuários o comportamento esperado da interface, mas estes artefatos técnicos deixam de ser o foco principal, e passam a ser apoio. Este uso adicional é justificado também porque "as melhores intenções do projetista não o tornam [o objeto de uso] manobrável pelo ingênuo [...] [que eventualmente pode não saber] *que determinadas formas significam determinadas funções* [grifo no original]" (ECO, 2005, p. 199-200). Com esta abordagem, o diálogo tende a ser estabelecido a partir de um objeto que é percebido por um não técnico de forma *talvez mais natural* que um diagrama elaborado segundo uma notação técnica. Destaque-se que não se sugere aqui que esta *naturalidade perceptiva* esteja associada a intuição[12], mas a uma convenção decorrente de uma metáfora que foi amplamente utilizada pelos ambientes gráficos e teve grande aceitação dos usuários não técnicos.

4 CONSIDERAÇÕES FINAIS

Uma vez que "a comunicação pressupõe sempre uma forma de eficiência comunicativa" (CAETANO, 2002, p. 62), mais importante que propor novas notações para documentação de software, ou simplesmente apontar problemas decorrentes das notações técnicas utilizadas, é fundamental sugerir alternativas

[11] Usabilidade é definida como "a capacidade que um sistema interativo oferece a um usuário, em um determinado contexto de operação, para a realização de tarefas, de maneira eficaz, eficiente e agradável" (ISO 9241).

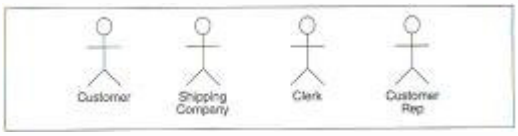
[12] Vale ressaltar ainda que "criar e compreender mensagens visuais é natural até certo ponto, mas a eficácia [...] só pode ser alcançada através do estudo" (DONDIS, 2003, p. 16).

para solucionar problemas práticos de comunicação que podem comprometer a qualidade dos sistemas computacionais desenvolvidos mundialmente.

Neste sentido, em termos praxiológicos, associar o *design* de interface às atividades de definição e formalização dos casos de uso parece uma necessidade urgente para facilitar o diálogo entre técnicos e usuários: "ver é um fato natural do organismo humano [...] [, mas] o fato de ver não garante a ninguém a capacidade de tornar compreensíveis [...] manifestações visuais" (DONDIS, 2003, p. 137). Sob uma perspectiva teórico reflexiva, parece relevante destacar que estudos associados ao efeito de sentido, comuns no campo da pesquisa comunicacional podem passar a ser considerados por teóricos de software durante as definições de formas de diálogo entre técnicos e usuários. Propor notações com base em suposta intuitividade eventualmente cria novos problemas práticos, ao invés de resolver problemas anteriores. Como argumenta Jacques Fontanille, "se o problema for colocado na perspectiva do reconhecimento da figura, é necessário se perguntar sob quais condições ela é reconhecida" (2005, p. 100). Assim, uma avaliação de possibilidades de trabalho interdisciplinar entre as áreas de software e comunicação aponta para novos e instigantes campos de pesquisa.

Figuras

FIGURA 1 – EXEMPLOS DE REPRESENTAÇÃO DE ATOR



FONTE: SCHNEIDER; WINTERS, 2001, p. 14

FIGURA 2 – EXEMPLO DE TRECHO DE DIAGRAMA DE CASO DE USO

FONTE[13]: [adaptação a partir de] SCHNEIDER; WINTERS, 2001, p. 38

FIGURA 3 – PROPOSTA DE ESTRUTURA ELEMENTAR DO ESQUELETO HUMANO

FONTE: ECO, 2005, p. 36

Anexo 1 – Exemplo de especificação de um fluxo de eventos

Schneider e Winters apresentam como possibilidade de documentação do fluxo de eventos de um caso de uso "Realizar pedido" (tradução do autor):

"**Fluxo de Eventos:**

Caminho Básico

1. O caso de uso inicia quando o cliente escolhe [a opção] Realizar Pedido.
2. O cliente informa seu nome e endereço.
3. Se o cliente informa apenas o código postal, o sistema preenche a cidade e o estado.
4. O cliente informa os códigos para os produtos a pedir.
5. Para cada código de produto informado
    a. o sistema fornece uma descrição e preço.
    b. o sistema adiciona o preço do item ao total.

   fim da repetição
6. O cliente preenche informações de pagamento com cartão de crédito.

[13] O diagrama apresentado é a tradução elaborada a partir de um recorte do diagrama original.

7. O cliente pressiona Submeter.

8. O sistema verifica a informação, salva o pedido como pendente e envia informações de pagamento ao sistema de contabilidade.

9. Quando o pagamento é confirmado, o pedido é salvo marcado como confirmado, um identificador do pedido é enviado ao cliente e o caso de uso termina.

Caminhos Alternativos

Alternativa 1: Dado incorreto

1. Este caminho alternativo inicia no passo 8 do caminho básico quando o sistema detecta informações incorretas.
2. O sistema solicita ao cliente que corrija a informação.
3. O caminho básico continua no passo 8.

Alternativa 2: Cancelamento

1. Em qualquer passo do caso de uso, o cliente pode selecionar cancelar.
2. O sistema solicita ao cliente que confirme o cancelamento.
3. O cliente seleciona OK e o caso de uso termina". (SCHNEIDER; WINTERS, 2001;, p. 46).

Referências

CAETANO, Kati. **Das Linguagens Secretas aos Segredos das Lingugens** [sic]. São Paulo: Arte e Cultura da América Latina – Revista da Sociedade Científica de Estudos da Arte (CESA) v.VIII, n. 1, p. 59-75, 1º Semestre/2002.

CHRISSIS, Mary Beth; KONRAD, Mike; SHRUM, Sandy. **CMMI Guidelines for process integration and produc improvement**. Boston: Addison-Wesley, 2003.

DONDIS, Donis A. **Sintaxe da linguagem visual**. São Paulo: Martins Fontes, 2003.

ECO, Umberto. *A estrutura ausente*. São Paulo: Editora Perspectiva, 2005.

ERIKSSON, Hans-Erik *et al*. **UML 2 Toolkit**. Indianapolis: Wiley Publishing Inc, 2004.

FONTANILLE, Jacques. **Significação e visualidade***: exercícios práticos.* Porto Alegre: Sulina, 2005.

FOWLER, Martin; SCOTT, Kendall. **UML Distilled: Applying the standard object modeling language**. Boston: Addison-Wesley, 1999.

GOMBRICH, E. H. **Arte e Ilusão: Um estudo da psicologia da representação pictórica**. São Paulo: Martins Fontes Editora, 1995.

ISO. **International Standard ISO 9241 Part 1: Ergonomic requirements for office work with visual display terminals General Introduction**. s/l (US), 1993.

JACOBSON, Ivar; BOOCH, Grady; RUMBAUGH, James. **The Unified Software Development Process**. BOSTON: Addison-Wesley, 1999.

JOLY, Martine. **Introdução à análise da imagem**. Campinas: Papirus Editora, 1996.

JOHNSON, Steven. **Cultura da Interface: Como o Computador Transforma Nossa Maneira de Criar e Comunicar**. Rio de Janeiro: Jorge Zahar Editor, 2001.

KOTONYA, Gerald; SOMMERVILLE, Ian. **Requirements Engineering: Processes and Techniques**. New York: John Wiley & Sons Inc, 1998.

PIGNATARI, Décio. **Informação, linguagem, comunicação**. Cotia: Ateliê Editorial, 2003.

PRESSMAN, Roger. **Software Engineering: A Practitioner's Approach: European Adaptation**. London: McGraw Hill International Limited, 2000

ROSENBERG, Doug; SCOTT, Kendall. **Applying use case driven object modeling with UML: An annotated e-commerce example**. Boston: Addison-Wesley, 2001.

SCHNEIDER, Geri; WINTERS, Jason P. **Applying Use Cases: A practical guide**. Boston: Addison-Wesley, 2001.