

Método de Avaliação de Usabilidade na Web Baseado em Modelo e Padrões de Comportamento¹

Natacha Güell
natacha@inf.puc-rio.br

Daniel Schwabe
schwabe@inf.puc-rio.br

Simone Diniz Junqueira Barbosa
sim@inf.puc-rio.br

Departamento de Informática, PUC-Rio. Rua M. de São Vicente, 225, Gávea. Rio de Janeiro, RJ 22453-900

PUC-RioInf.MCC 18/01 Julho, 2001

Abstract

This document proposes a usability evaluation method for Web applications. The proposed method helps designers to diagnose and solve design problems. Our approach is to recognize, during users' interaction with the system, behaviors that are conduct patterns for typical design problems. Initially, behavioral patterns were described, classified and mapped onto classes of design problems. Then, the design problem indicated by each behavioral pattern is diagnosed using the OOHDM methodology. Finally, some design solutions are proposed based on Guidelines and Design Patterns.

Keywords: *Web usability, evaluation method, Web applications design, design patterns*

Resumo

Este trabalho propõe um método de avaliação de usabilidade de aplicações *Web*, que ajuda o projetista no diagnóstico e solução de problemas de projeto. O método propõe reconhecer, durante a interação dos usuários com o sistema, comportamentos que constituem padrões de conduta ante problemas de projeto típicos. Estes padrões de comportamento foram descritos e classificados, bem como mapeados para classes de problemas de projeto. Para cada padrão de comportamento, é diagnosticado, no âmbito da metodologia OOHDM, o problema de projeto que ele manifesta, e são sugeridas soluções alternativas com base em *Guidelines* e *Design Patterns*.

Palavras-chave: usabilidade na *Web*, método de avaliação, projeto de aplicações para a *Web*, *design patterns*

¹ Versão ampliada de artigo submetido ao SBMIDIA 2001.

1. Introdução

A utilização da *Web* como plataforma para disponibilização de aplicações tem levado ao surgimento de diversos métodos e notações para o suporte ao seu desenvolvimento, tais como OOHDM [Schwabe, 1998], HDM 2000 [Garzotto, 2000], OO-H Method [Gomez, 2001], WSDM [De Troyer, 1998], WebML [Ceri, 2000], entre outros. Todos estes métodos cobrem, em maior ou menor grau, as fases de levantamento de requisitos, projeto e implementação.

Em todo método de projeto existem ciclos de *feedback* onde cada fase realimenta as fases anteriores, acrescentando detalhes e muitas vezes levando a revisões. Sob esta ótica, existe ainda uma lacuna em relação à última fase, que é a de disponibilização de uma aplicação no seu ambiente final de uso. Não existem, ainda, métodos integrados para a avaliação do uso destas aplicações de tal forma que os resultados desta avaliação sirvam para fechar o ciclo de *feedback* com as fases anteriores de projeto e implementação.

Para permitir obter um ciclo completo de análise, projeto, implementação e avaliação, iniciou-se uma investigação sobre formas de avaliação do uso de aplicações *Web* que pudessem ser integradas às etapas precedentes. Idealmente, as observações realizadas durante o processo de avaliação de uma aplicação devem fornecer *feedback* sobre decisões de projeto e de implementação que permitem aperfeiçoá-la.

A experiência concreta no projeto de aplicações mostrou que, por mais completo que seja o método utilizado, ainda é necessário que o projetista conheça a sua forma de aplicação, e como abordar uma situação real através do uso do método. De fato, métodos de projeto não são determinísticos; eles fornecem primitivas para construir os modelos e indicam os momentos em que devem ser tomadas as decisões de projeto, mas não dão soluções fechadas para cada problema de projeto. Além disto, sempre existem aspectos que não são completamente determinados pelo método. Por estas razões, vários autores têm proposto o uso de padrões de projeto (*design patterns*) e diretrizes (*guidelines*) [Rossi 1997, 1999], [Garzotto, 1999], [Borchers, 2001] como forma de capturar a experiência de bons projetistas.

Embora aparentemente contribuam para a qualidade da solução, nem sempre é evidente que o uso de um padrão de projeto realmente leva a um melhor resultado. Visando comprovar esta hipótese, foram realizados testes de usabilidade buscando verificar o benefício de utilizar padrões de projeto. Um estudo preliminar verificou, informalmente, que havia padrões de comportamento que pareciam corresponder a problemas conhecidos de projeto de aplicações *Web*.

A partir desta experiência, decidiu-se realizar um estudo mais aprofundado, visando estabelecer um método para avaliação destas aplicações, que apresentamos neste artigo. Antes de detalhar um pouco mais a abordagem proposta, apresentamos uma revisão do nosso entendimento sobre usabilidade.

Usabilidade é um conceito de qualidade de uma aplicação sob uma perspectiva de uso, tradicionalmente relacionado a cinco atributos: facilidade de aprendizado, eficiência, facilidade de reter o conhecimento sobre a aplicação obtido em usos anteriores (memorização), baixo índice de erros, e satisfação dos usuários [Nielsen, 1993]. Segundo [Garzotto, 1998], *usabilidade de aplicações Web* é “a habilidade do usuário em utilizar *sites* e acessar o conteúdo deles do modo mais efetivo. Como consequência, tornou-se obrigatório prover tanto o critério de qualidade que os *sites* têm que satisfazer para ser utilizáveis, como os métodos sistemáticos para avaliar tal critério”.

O processo de projetar e avaliar a usabilidade de uma aplicação se denomina *engenharia de usabilidade* [Nielsen, 1993]. Os métodos de avaliação podem ser classificados como: (1) testes com usuários, em que usuários reais testam a usabilidade das aplicações; (2) inspeção, onde avaliadores especialistas conferem se uma aplicação cumpre um conjunto predefinido de princípios de projeto de interface. Avaliação através de testes com usuários “é o método de usabilidade mais fundamental e é praticamente insubstituível, pois provê informação direta sobre como as pessoas usam os computadores e quais são os problemas exatos dos usuários com a interface concreta que é testada” [Nielsen, 1997]. Os testes de usabilidade visam principalmente verificar, de forma quantitativa, o cumprimento dos princípios de usabilidade. Neles, são estabelecidos fatores críticos e faixas de valores que determinam a usabilidade da aplicação. Em seguida, são realizados testes para verificar se os valores obtidos para cada fator crítico são ou não satisfatórios, e como resultado são fornecidas indicações sobre que aspectos devem ser modificados.

Um método de avaliação complementar ao de usabilidade é o de *comunicabilidade* [Prates, 2000]. Os testes de comunicabilidade visam avaliar o quão bem as decisões do projetista estão sendo comunicadas aos usuários, ou seja, se os signos de interface escolhidos por ele estão transmitindo com êxito o modelo conceitual da aplicação. Estes testes fornecem, além de medidas quantitativas, uma avaliação qualitativa da interface, pois permitem identificar pontos de ruptura da comunicação entre o projetista e os usuários. A identificação é feita a partir de interjeições potenciais do usuário ante um problema de interação. Como resultado, dão indicadores sobre o que precisa ser modificado para que o usuário entenda a aplicação, através de sua interface.

Tanto os métodos de avaliação de usabilidade quanto os de comunicabilidade não utilizam um método de projeto específico como referência, ou seja, podem ser utilizados em qualquer ciclo de desenvolvimento ou metodologia de projeto. Sendo assim, não pressupõem modelos intermediários específicos utilizados na geração das aplicações. A falta do conhecimento destes modelos faz com que os problemas sejam apreciados de modo pontual, associados a algum elemento da interface ou a um modelo conceitual abstrato. As orientações que se derivam destes métodos de avaliação são, portanto, diretrizes genéricas.

De forma análoga ao método de comunicabilidade [Prates, 2000], que classifica as rupturas de comunicação segundo as interjeições que o usuário potencialmente manifestaria, o método proposto classifica problemas de uso de uma aplicação *Web* a partir do comportamento que os usuários manifestam ante os mesmos. Formas típicas de comportamento (seqüências navegacionais e interações inadequadas e recorrentes) que indicam problemas de projeto concretos são tomadas como base na análise da usabilidade de uma aplicação. Padrões de projeto [Garrido, 1997], [Lyardet, 1998] e a metodologia OOHDM [Schwabe, 1998] são utilizados como referência para detectar e tratar os problemas de projeto.

OOHDM (*Object-Oriented Hypermedia Design Method*) [Schwabe, 1998] é uma metodologia de projeto de aplicações hipermídia, que tem sido utilizada amplamente no desenvolvimento de *sites Web* [Schwabe, 1999]. Ao utilizar um método de projeto para *Web* como base, o método de avaliação proposto permite sugerir soluções mais precisas, e portanto aplicáveis de forma mais direta no reprojeto das aplicações. O seu maior benefício é que, por ser calcado no OOHDM, é possível perceber a aplicação como um modelo e saber as razões de projeto para cada uma de suas partes. Isto permite um mapeamento quase direto do padrão de comportamento para um ou vários problemas de projeto e destes para as soluções.

Tipicamente, aplicações são avaliadas por especialistas em avaliação, que tentam comunicar ao projetista o diagnóstico de problemas, bem como alternativas de solução. Entretanto, eles podem não ter conhecimento suficiente sobre o projeto, acarretando importantes problemas de comunicação e de perda de informação. A proposta deste trabalho é que a avaliação seja feita pelo próprio projetista que, a partir de padrões de comportamento dos usuários, consegue detectar problemas no projeto e indicar as correções necessárias. Desta forma, o método fecha o ciclo de um projeto OOHDM, pois provê *feedback* sobre todas as fases de projeto: requisitos, modelagem conceitual, projeto de navegação, e projeto da interface.

Na próxima seção, apresentamos o método proposto; na seção seguinte, é mostrado um estudo de caso ilustrando o uso do método. Em seguida, são apresentadas conclusões e trabalhos futuros.

2. Método

O método proposto consiste nas seguintes etapas: (1) coleta de dados em testes de usuários; (2) mapeamento dos padrões de comportamento que ocorreram nos testes para os problemas de projeto; e (3) mapeamento dos problemas de projeto para as soluções, usando determinados padrões de projetos e diretrizes.

2.1 Coleta de Dados

Os dados são obtidos mediante observação de usuários executando tarefas. É preciso: (1) determinar as tarefas que serão avaliadas; (2) definir o perfil dos participantes e recrutá-los; e (3) realizar e registrar os testes.

Para decidir as tarefas que serão avaliadas deve-se levar em consideração o conjunto de tarefas que a aplicação deva apoiar, segundo o levantamento de requisitos feito no início do projeto.

O perfil dos participantes dependerá do objetivo de cada teste. Serão escolhidas pessoas de cada classe de usuários prevista para a tarefa. No caso de aplicações *Web* é recomendável também que as pessoas tenham diferentes níveis de experiência na utilização da *Web*. Para determinar o número de participantes por classe de usuários podem ser seguidas as recomendações de [Nielsen, 2000], que propõe utilizar poucos usuários nos testes, pois a partir de certo número (5) as observações começam ser repetitivas, resultando em perda de tempo e recursos. Nielsen explica que, quando são várias classes de usuários, este número pode ser ainda menor.

Se o objetivo da avaliação for comparar diversas alternativas de solução para uma tarefa, a ordem das soluções deve variar de um teste para outro: metade dos usuários de igual perfil devem utilizar a mesma ordem. Desta forma, reduzimos algumas distorções causadas pela influência que o conhecimento de uma solução possa exercer na utilização da outra.

Durante os testes, os usuários são observados cumprindo as tarefas predeterminadas. A interação do usuário com a aplicação é gravada, utilizando-se um software para capturar a seqüência de telas e os movimentos do cursor. As expressões e comentários do usuário também são registrados pelo avaliador mediante anotação escrita, ou em fitas de vídeo e/ou áudio.

2.2 Mapeamento dos Padrões de Comportamento

O registro dos testes é analisado para identificar as seqüências de interações que correspondem aos padrões de comportamento que representam problemas de projeto. Os

pontos da aplicação onde se evidenciaram formas de interação problemáticas, assim como o número de incidências, vão sendo registrados. Posteriormente, os registros são interpretados segundo o(s) problema(s) de projeto que o tipo de comportamento pode indicar. O avaliador dispõe de uma tabela de mapeamento dos padrões de comportamentos que indica os possíveis problemas de projeto de cada padrão.

2.3 Mapeamento dos Problemas de Projeto a Possíveis Soluções

Nesta etapa, é fundamental ser capaz de reconhecer as primitivas de projeto que estão por detrás dos elementos de interface. Para isso, é fundamental dominar as primitivas do modelo de projeto, no caso, de OOHD. O avaliador dispõe de uma tabela de mapeamento dos problemas de projeto que indica, para cada problema de projeto, soluções alternativas: padrões de projeto ou diretrizes baseadas em OOHD.

O processo de aplicar padrões de projeto não é complicado. Os padrões descrevem o contexto no qual podem ser aplicados. Explicam o problema que pretendem resolver e as forças que interagem dando vida ao problema. Apresentam vários casos como exemplos do problema, e discutem os diferentes aspectos que devem ser considerados para aplicar a solução. Descrevem a solução e também oferecem exemplos de implementação da mesma. Diretrizes, por sua vez, fornecem orientações mais genéricas, que devem ser observadas na ausência de um padrão específico que se encaixe na situação desejada.

3. Estudo de Caso

Foram realizados testes com usuários, com o objetivo de identificar padrões de comportamento durante a interação, ante problemas de projeto. Teoricamente, aplicações que tenham sido projetadas seguindo algum método, ou em que foram aplicados padrões de projeto, devem ser mais fáceis de usar, e com melhor desempenho dos usuários. Com o objetivo de observar a atitude e o desempenho dos usuários ante diferentes tipos de projetos, foram selecionados *sites* que permitiam realizar uma mesma tarefa, mas que utilizavam diferentes soluções de projeto: pelo menos uma solução similar à solução prevista pelo OOHD, e uma solução *naif*.

3.1 Testes com Usuários

Foram analisados sites *Web* pertencentes a domínios variados: sites de departamentos acadêmicos, centros de pesquisa, museus virtuais e lojas de diferentes tipos de produtos. Parte da caracterização do domínio de uma aplicação inclui as tarefas típicas que ela deve permitir realizar. Para cada domínio analisado, foi levantado um conjunto das tarefas que as aplicações supostamente² deveriam apoiar. Para cada tarefa, foi descrita a solução de projeto que, segundo o OOHD, permitiria

<i>Domínio:</i> Departamento Acadêmico	
<i>Tarefa:</i> Achar, dentre os professores que trabalham em uma determinada área, algum que trabalhe em um determinado tópico	
<i>Passos:</i> 1. Achar os professores da área. 2. Analisar as informações sobre cada professor, navegando caso necessário, até achar o professor que tem alguma informação sobre o tópico desejado	
<i>Sites</i>	<i>Projeto</i>
Dpto. Elétrica SC	<i>Naif:</i> Estrutura de acesso 'Professores', ordenados por nome do professor
Dpto. Elétrica Puc-Rio	Intermediária: Estrutura de acesso 'Professores', agrupados por área e ordenados por nome
Dpto. DI Puc-Rio	Padrão: Página de Área: Apresenta a estrutura de acesso 'Professores por Área' Página de Professor no contexto 'Professores por Área': permite navegação anterior, próximo e link para a estrutura de acesso

Tabela 1 – Registro da busca de *sites* para escolher quais avaliar.

² Como não fomos nós quem construiu a maioria dos *sites* avaliados, não temos acesso aos seus requisitos reais.

a sua realização de forma eficiente e adequada a seus usuários-alvo. A Tabela 1 exemplifica o registro da análise dos *sites*.

Finalmente foram projetados os testes: foram selecionadas as tarefas e os *sites* em que seriam executadas (solução padrão vs. solução *naif*).

Durante a inspeção dos *sites*, foram previstos problemas de usabilidade que os usuários poderiam encontrar. Estes problemas foram analisados e classificados conforme o método OOHDM. Durante os testes, a atitude dos usuários nestes pontos problemáticos seria observada com atenção, tentando vincular padrões de interação em que problemas ocorrem (sintomas) a problemas de projeto (causas).

A seleção dos participantes foi baseada nas categorias de usuários de cada tarefa, na experiência no uso da *Web* e na experiência no uso de aplicações de igual domínio daquele que seria testado. Foi utilizada uma média de 8 usuários por teste.

Classe	Secretária		Estudante				Profissional		
Usuário	U10	U12	U14	U05	U11	U15	U04	U07	U13
Uso Web	Muito	Pouco	Pouco	Muito	Muito	Pouco	Pouco	Muito	Médio
Anos	6	2	3	7	5	1	5	6	7
Hrs./Semana	15	1	6	20	10	1	5	14	7
Info.	Áreas Corpo docente Cursos	Corpo docente	Áreas Cursos Eventos	Disciplinas Publicações	Artigos Equipamentos Softwares Pós-doutoramento Emprego	Livros da biblioteca	Artigos Material didático de cursos	Cursos Endereços	Áreas Pós-doutoramento

Tabela 2 – Perfil dos usuários participantes nos testes da tarefa “Achar, dentre os professores que trabalham em uma determinada área, algum que trabalhe em um determinado tópico”

Por exemplo, nos testes da tarefa “Achar, dentre os professores que trabalham em uma determinada área...”, realizados em três *sites* docentes, participaram 9 usuários. A Tabela 2 resume o perfil destes participantes quanto à: (1) classe a que pertence (secretária, estudante ou profissional da área); (2) experiência no uso da *Web*: há quantos anos utiliza a *Web* e qual a frequência semanal de uso e (3) forma de uso de aplicações do domínio: que informações costumam procurar em *sites* docentes. Neste caso, era desejável que os usuários já tivessem utilizado este tipo de *site*.

No início do teste, cada usuário recebeu um cenário de teste com uma descrição textual detalhada do enunciado da tarefa que deveria realizar, mas evitando induzir alguma estratégia para a execução da mesma. Durante o teste, o usuário realizou a mesma tarefa em cada um dos *sites* sob avaliação. As sessões de testes foram acompanhadas por um avaliador; a interação no computador foi gravada; e os comentários e atitudes do usuário foram registrados de forma escrita. Ao final do teste, o avaliador fazia perguntas ao usuário sobre a realização da tarefa e sobre as dificuldades apresentadas, além de outras perguntas que lhe eram interessantes.

3.2 Classificação de padrões de comportamento

As tabelas a seguir apresentam o resultado da classificação dos padrões de comportamento durante a interação, relacionando-os a suas prováveis causas. Estas tabelas estão divididas de acordo com os elementos do OOHDM onde ocorrem os problemas indicados.

Situações que caracterizam problemas no projeto de uma estrutura de acesso (EA):

<i>Padrão de Comportamento</i>	<i>Diagnóstico</i>
1. Na EA, clicar em uma âncora associada a um objeto, perceber que o destino não era o desejado, voltar à EA e selecionar outra âncora do mesmo objeto.	Projeto da interface: escolha das âncoras seletoras de um objeto
2. Na EA, ficar na dúvida (indicado por uma hesitação, com o cursor se movendo entre dois ou mais elementos, ou comentário) sobre que objeto deve ser selecionado para alcançar um determinado objetivo, selecionar um objeto e ir navegando até perceber que o destino não era o desejado, voltar à EA inicial e tentar novamente selecionando outro objeto.	Projeto conceitual ou navegacional: Classificação inadequada. Projeto da interface: Escolha de termos inadequados.
3. Na EA, ficar na dúvida sobre que objeto deve ser selecionado para alcançar um determinado objetivo, desistir de navegar através da EA e tentar formular uma consulta.	Projeto conceitual ou navegacional: Classificação inadequada. Projeto da interface: Escolha de termos inadequados. Mapeamento navegacional para interface: Mapeamento da categorização.
4. Na EA, ir analisando todos os objetos, mediante observação e, mentalmente, ir comparando suas propriedades.	Projeto navegacional: Não se disponibiliza operação necessária para processar (comparar) o conjunto de objetos.

Tabela 3 – Problemas relacionados a estruturas de acesso.

Situações que caracterizam problemas no projeto da estrutura navegacional:

<i>Padrão de Comportamento</i>	<i>Diagnóstico</i>
5. Na EA, ir acessando, de forma seqüencial, somente aqueles objetos que têm uma determinada característica em comum. Acessar um objeto, analisá-lo, voltar à EA e selecionar o próximo que satisfaz a mesma condição.	Projeto navegacional: Não se disponibiliza um contexto necessário
6. Acessar um objeto e desistir. Acessar uma EA e continuar na forma 5, mas visitando objetos cuja característica comum é estarem vinculados com o objeto que foi visitado inicialmente.	Projeto navegacional: Não se disponibiliza um contexto necessário Projeto navegacional: Não se vinculam objetos
7. Acessar uma EA e desistir. Acessar um objeto e desistir. Voltar na EA e continuar na forma 5, mas visitando objetos cuja característica comum é estarem vinculados com o objeto que foi visitado.	Projeto navegacional: Não se disponibiliza um contexto necessário Projeto navegacional: Não se vinculam objetos
8. Acessar uma EA e formular consulta, com o intuito de filtrar.	Projeto navegacional: Não se disponibiliza um contexto necessário.* *É problema navegacional caso seja uma tarefa elicitada. Elicitação de requisitos: A tarefa não foi considerada relevante, ou não foi identificada. Pode indicar também um requisito novo: na época da elicitación a tarefa não era necessária.
9. Na EA, acessar um objeto e, para selecionar outro objeto do mesmo conjunto, voltar à EA repetindo a mesma seqüência navegacional que tinha sido seguida até a EA, ou utilizar o comando 'back' (usuários experientes em <i>browser</i>).	Projeto navegacional: Não se disponibiliza navegação dentro de um contexto

Tabela 4 – Problemas relacionados a estruturas de navegação.

Situações que caracterizam problemas de interface:

<i>Padrão de Comportamento</i>	<i>Diagnóstico</i>
10. Repetir várias vezes a mesma operação, achando que na tentativa anterior não conseguiu realizar a operação com êxito.	Projeto da interface: <i>Feedback</i> de processo deficiente
11. Verificar cada operação de atualização: depois de aplicar a operação, navegar até o objeto da atualização para conferir.	Projeto da interface: <i>Feedback</i> de processo deficiente

Tabela 5 – Problemas relacionados à interface.

3.3 Sugestões para Reprojeto

Nesta seção são apresentados alguns dos casos que exemplificam os padrões de comportamento descritos nas tabelas da seção anterior (Tabela 3, Tabela 4 e Tabela 5). Para cada padrão se indica o problema de projeto que ele manifesta e se sugere uma possível solução.

3.3.1 Âncoras em Estruturas de Acesso

1. Na EA, clicar em uma âncora associada a um objeto, perceber que o destino não era o desejado, voltar à EA e selecionar outra âncora do mesmo objeto.

Esta situação aconteceu nos testes realizados no *site* do Departamento Acadêmico de Elétrica da PUC-Rio. A tarefa consistia em achar, dentre os professores que trabalham em uma certa área de pesquisa, aqueles que desenvolvem um determinado tópico. Os usuários precisavam analisar os professores da área até achar um que se dedicasse ao tópico de interesse.

A EA da Figura 1 apresenta os professores agrupados por área de pesquisa. Para cada professor se mostra: nome (que é uma âncora), grau, instituição e ano de obtenção do grau, e um link rotulado 'Página Pessoal'.

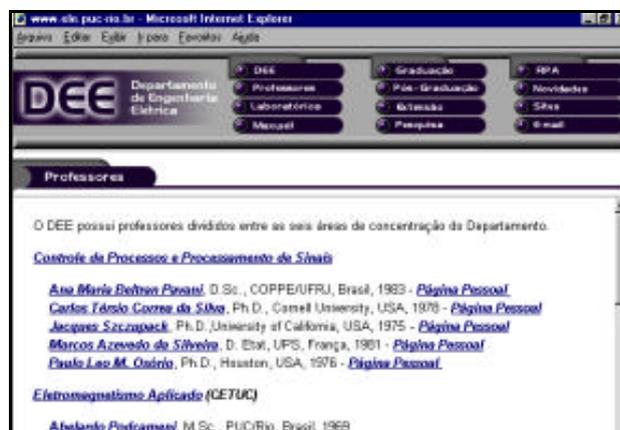


Figura 1 – EA 'Professores': professores ordenados por área.

Durante os testes, os usuários clicam no nome mas ficam desapontados quando o sistema apresenta o editor de correio eletrônico, com o professor como destinatário: “*O que tem no nome é o mail?!*”, “*Achava que levaria à página do professor*”, “*Cliquei errado no nome!*”. Então fecham o editor de correio eletrônico, voltam ao índice, analisam novamente as âncoras e selecionam ‘Página Pessoal’. Muitos usuários clicaram no nome em várias ocasiões que foram selecionar um professor: “*O mail... eu sempre clico errado*”. Dos 9 usuários participantes, somente 2 clicaram no link ‘Página Pessoal’ logo na primeira tentativa, e mesmo assim 1 deles ficou movendo o mouse sobre ambos links, hesitante. Quando questionados sobre por que não selecionaram logo o link ‘Página Pessoal’, eles explicaram que achavam que a página seria acessada através do nome e que ‘Página Pessoal’ levaria a uma página com informações de cunho pessoal, e não profissional.

Problema: As âncoras não sugerem o seu verdadeiro destino.

Solução: É necessário escolher uma âncora que seja semanticamente significativa, para ajudar ao usuário a perceber ou intuir o que ele vai encontrar no destino. Este problema abarca dois passos no projeto OOHDM de uma EA. Durante o projeto da navegação se define qual será a visão do objeto na EA (atributos e funções que o objeto terá: podem ser funções navegacionais ou operacionais). Durante a fase de projeto abstrato da interface se decide como será a aparência das âncoras que ativaram as funções: podem ser atributos do objeto, ícones ou rótulos.

O problema de escolha da âncora é um problema de comunicação entre o projetista e o usuário (enfoque da Engenharia Semiótica [de Souza, 1993]). O projetista está tentando dizer “se você entrar por aqui, você vai acessar tais informações”. O grande desafio é conseguir que os usuários entendam essa mensagem. A âncora deve refletir a natureza da relação que está codificada nela.

Geralmente, quando o destino de uma âncora é um objeto, é padrão utilizar o próprio atributo identificador do objeto. Por exemplo: o *título* para acessar o *livro*, o *nome* e/ou a *foto* para acessar a *pessoa*. Em uma EA, quando o identificador de cada objeto é âncora, geralmente representa uma relação funcional: detalhamento do objeto. E quando a âncora é um atributo funcionalmente dependente de um outro objeto, a semântica é a relação que

Um outro *site* onde houve dúvidas sobre qual seria a categoria certa foi o *site* Walmart.com, que comercializa, entre outras coisas, produtos de beleza. A tarefa consistia em comprar um condicionador de cabelos. Logo na *homepage*, os usuários hesitaram entre as categorias ‘beleza e spa’ ou ‘cuidado pessoal’, ambas no submenu ‘saúde e beleza’. Afortunadamente, selecionar uma ou outra categoria não fazia diferença, pois no nível seguinte de ambas apareceria a categoria ‘cuidados para o cabelo’, supercategoria de ‘condicionadores’. Neste caso, os usuários não precisaram voltar atrás para refazer a seleção, mas a dúvida inicial foi igualmente incômoda. Ante a insegurança provocada pelas categorias alguns participantes (2 dos 9, em cada *site*) desistiram de navegar pela EA e preferiram formular consulta.

Problema: Não é evidente qual o critério de categorização que foi utilizado:

1. O produto poderia pertencer a várias das categorias apresentadas, mas o usuário não entende qual critério foi utilizado para categorizar, não entende por que existem essas categorias, por que uma única não seria suficiente. Então, ele teme errar na hora de selecionar. (ie.: *site* Walmart.com, categoria ‘condicionador de cabelos’)
2. O produto poderia pertencer a várias das categorias apresentadas, porém, não pertencia a todas (ie.: *site* BabyStyle.com, um suéter para uma bebê poderia estar em ‘menina’ ou ‘guarda roupa do bebê’, somente estava em ‘guarda roupa do bebê’).

Em lojas virtuais e catálogos eletrônicos, o usuário seleciona produtos ou serviços. Dois possíveis casos de uso nestas aplicações são: (1) quando o usuário sabe exatamente o produto que está procurando (nome, marca); (2) quando o usuário está interessado em certo tipo de produto mas deseja se informar primeiro sobre as ofertas.

No primeiro caso, muito provavelmente o usuário vai formular consulta; no outro, o usuário geralmente tenta acessar a informação através da navegação.

Com o intuito de apoiar a navegação, as aplicações geralmente fornecem EAs hierárquicas que permitem acessar os produtos através de categorias que formam uma estrutura taxonômica. A qualidade da navegação através das categorias vai depender tanto de como um determinado produto se ajusta na taxonomia, como da forma como os usuários interpretam a classificação utilizada.

Solução: Prover uma EA hierárquica que permita navegar através de uma taxonomia cujos critérios de categorização sejam claros e não gerem dúvidas. Uma boa taxonomia é uma estrutura que forneça caminhos diferenciados até o produto e que não crie incertezas no usuário. Isto envolve três aspectos: a categorização, o elemento de interface para representar cada categoria, e o mapeamento da categorização para a interface da EA.

Categorização: Uma categorização deve ser abrangente no sentido de fornecer acesso a todo objeto da base de conhecimento. Um exemplo de categorização ruim é o caso do usuário que, baseado na observação das categorias, considera que o produto que procura não existe na loja, e o produto existe. A categoria não indica que o produto é parte dela.

Uma categorização deve ser precisa. Cada categoria deve induzir quais objetos podem ser acessados através dela, deve evidenciar de forma clara o que os caracteriza. Desta forma, caso o usuário ache que um mesmo objeto pode ser acessado através de várias categorias, ele poderá selecionar a que mais se ajusta a seu caso de busca, sem duvidar. Quando não há precisão, é possível, através de uma categoria, recuperar objetos indesejados ou não recuperar o objeto procurado. Uma categorização imprecisa, com categorias ambíguas, cria uma incerteza que o usuário somente terá possibilidade de esclarecer mais na frente, e pode que nunca consiga esclarecer. Nas entrevistas após os testes, os usuários explicaram

que não gostaram de ter que ficar com uma indefinição pendente, sem saber quais seriam suas possíveis conseqüências, mesmo que fosse possível chegar no produto por um ou outro caminho. Na hora de fazer a escolha, o usuário não sabe que “tanto faz”. Se para ele ficasse claro que escolhendo qualquer caminho ele atingiria seu objetivo, então não existiria problema algum.

Interface para as Categorias: Construir uma taxonomia abrangente e precisa não é suficiente, pois o que importa é o que o usuário percebe e entende: pode ocorrer que exista o caminho mas o usuário não reconheça a categoria, que exista um único caminho e que o usuário ache que servem várias categorias, que sirvam várias categorias e o usuário não entenda a classificação.

Estes problemas podem ser conseqüência do signo atribuído à categoria, ou seja, o termo, o rótulo e/ou a imagem utilizados para representá-la não correspondem com o conceito que o usuário tem. Tal foi o caso dos submenus ‘menino’ e ‘menina’, do site BabyStyle.com, onde a combinação de imagem e rótulo criou uma idéia errada da categoria. Muitas vezes são utilizados termos técnicos que fazem sentido para alguma comunidade que não é o público-alvo do site.

Mapeamento da Taxonomia para a Interface: Outro aspecto problemático é o mapeamento da taxonomia para a interface: uma estrutura muito profunda não deve ser refletida diretamente na interface, ou seja, pode-se apresentar mais de um nível da estrutura em uma mesma página, evitando passos excessivos de navegação. Uma estrutura muito extensa (larga) poderia gerar menos ambigüidade, porém ser mais difícil de acomodar na interface.

3.3.3 Critério de Ordenação

4. Na EA, ir analisando todos os objetos, mediante observação e, mentalmente, ir comparando suas propriedades.

Um exemplo deste caso foi observado nos testes aplicados ao site Beauty.com, que comercializa produtos de beleza. A tarefa consistia em comprar o condicionador mais barato. A EA de condicionadores (Figura 3) era muito grande e tinha sido fragmentada em três páginas. Para cada produto na EA era mostrada uma foto, a marca, o nome e o preço. Os produtos estavam ordenados por marca e nome. Além da listagem, na parte superior de cada página eram apresentados três produtos em destaque.

Os usuários precisaram navegar através das três páginas, observando os preços e lembrando o produto mais barato observado até o momento. Seus comentários indicavam que a tarefa estava sendo trabalhosa e que era difícil lembrar qual produto seria mais barato.

Alguns usuários, antes de iniciar a busca, fizeram os seguintes comentários: “São três páginas!”; “Não tem como ordenar por preço?”; “Não me permite filtrar?... Se não tem como filtrar e são três páginas de lista eu poderia ir olhando todos os produtos e ver qual vai ganhando, se fossem mais páginas não faria isso nunca”. Outros, depois de analisar



Figura 3 – Estrutura de Acesso para ‘Conditioners’: três produtos em destaque; listagem distribuída por três páginas.

até a última página comentaram: “Acho que na primeira tinha um por 12 dólares”; “O mais barato está na frente, como faço para voltar?”.

Problema: A EA não facilita o processamento do conjunto de objetos, apesar de ser muito comum a comparação de propriedades (preço, quantidade, marca...) no processo de compra.

Solução: A função de uma EA é apresentar um conjunto de objetos para que o usuário selecione algum, seja para acessar mais informações sobre ele, ou para processá-lo. Antes de escolher um objeto, o usuário analisa o conjunto, observa os objetos, percebe suas características, e compara mentalmente. EAs provêm uma maneira de restringir o conjunto que se quer comparar.

Para facilitar sua comparação, há três propriedades que os objetos em uma EA devem cumprir: devem ser diferenciados, devem apresentar informações comparáveis e devem estar ordenados. O problema é, quais informações e qual critério de ordenação são relevantes para comparar os objetos? Isto vai depender das estratégias do usuário ao utilizar a EA, que podem variar bastante: um mesmo tipo de usuário, ou inclusive uma mesma pessoa, pode ter interesses diferentes em cada ocasião.

Portanto, é preciso fazer um levantamento das tarefas junto aos usuários, tentar coletar as mais diversas estratégias para a mesma tarefa, e posteriormente decidir quais delas a EA vai apoiar: podem ser as formas de uso que foram descritas com maior frequência, ou aquelas que são de interesse dos promotores do *site*.

Uma vez decididas as informações que de cada objeto serão apresentadas, deve-se decidir quais podem ser utilizadas para ordenar os objetos. Novamente é necessário balancear entre os diferentes interesses estratégicos dos usuários e promotores do *site*. Em uma loja por exemplo, para alguns faz sentido comparar produtos por preço, para outros por marca, ou ainda por alguma propriedade do produto (potência, capacidade, consumo de energia). O dono da loja, por sua vez, pode estar interessado em apresentar primeiro os produtos em promoção.

Então, ante a diversidade estratégica na execução da mesma tarefa, uma solução é fornecer EAs que possam ser customizadas pelo próprio usuário. Teoricamente, tudo poderia ser customizado: que o usuário selecione, dentre os atributos da classe, aqueles que deseja apreciar; que selecione, dentre os objetos de um conjunto, o subconjunto que deseja comparar; e/ou que selecione, dentre os atributos da classe, aqueles que podem ser utilizados para ordenar o conjunto.

A solução mais utilizada é customizar a ordenação dentre um conjunto de critérios predeterminados pelo projetista. Os objetos são apresentados segundo uma determinada ordenação e posteriormente o usuário pode, sob demanda, rearrumar os objetos. Por exemplo, os produtos poderiam ser apresentados segundo o critério de comercialização que o dono deseje e permitir reordenar por preço ou por marca. Este tipo de EA é descrita em OOHDM como EA de ordenação múltipla [Schwabe, 1998].

OOHDM propõe identificar as informações e decidir a ordenação de EAs a partir da descrição das tarefas [Güell, 2000]. Cenários, *use cases* e diagramas de interação do usuário – UIDs – são utilizados para descrever as tarefas do ponto de vista dos usuários, e validados junto aos clientes do projeto. Cenários que descrevem diferentes estratégias para uma mesma tarefa podem ser unificados em um mesmo *use case* (cada *use case* é representado também em um UID).

3.3.4 Navegação Baseada em Conjuntos / Unidades de Informação

5. Na EA, ir acessando, de forma seqüencial, somente aqueles objetos que têm uma determinada característica em comum. Acessar um objeto, analisá-lo, voltar à EA e selecionar o próximo que satisfaz a mesma condição.
6. Acessar um objeto e desistir. Acessar uma EA e continuar na forma 5, mas visitando objetos cuja característica comum é estarem vinculados com o objeto que foi visitado inicialmente.
7. Acessar uma EA e desistir. Acessar um objeto e desistir. Voltar na EA e continuar na forma 5, mas visitando objetos cuja característica comum é estarem vinculados com o objeto que foi visitado.
8. Acessar uma EA e formular consulta, com o intuito de filtrar.

Cada uma destas situações reflete diferentes problemas de projeto que serão explicados a seguir utilizando um mesmo exemplo. Estas situações aconteceram em dois dos *sites* acadêmicos testados (*site A*, do Departamento de Elétrica da UFSC e *site B*, do Departamento de Elétrica da PUC-Rio).

A tarefa “achar, dentre os professores que trabalham em uma determinada área, algum que trabalhe em um determinado tópico” requeria que os usuários analisassem a página dos professores da área até achar algum que se dedicasse ao tópico de interesse.

O padrão (6) se manifestou em ambos *sites* (4 usuários no *site A* e 3 no *site B*): eles escolhem a área, achando que haverá informação sobre os professores associados a ela. A ausência deste tipo de informação faz com que decidam acessar a EA ‘Professores’.

Padrão	Navegação no Teste	Site A	Site B
6	Acessam a área. Desistem. Percorrem a EA ‘Professores’	4	3
7	Acessam a EA ‘Professores’. Desistem. Acessam a área. Desistem. Percorrem a EA ‘Professores’	2	3
8	Tentam por EA ‘Professores’ e a área. Desistem. Tentam formular consulta.	2	1

Tabela 6 – Padrões de comportamento dos 9 participantes no teste da tarefa

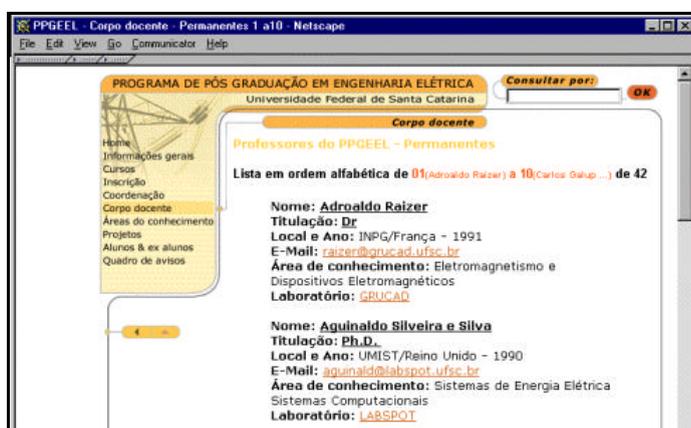


Figura 4 – Site A: estrutura de acesso ‘Professores’: conjunto de 42 professores distribuídos em 4 páginas

O padrão (7) também se manifestou em ambos *sites* (2 usuários no *site A* e 3 no *site B*). No *site A*, por exemplo, a EA ‘Professores’ (Figura 4) apresenta o conjunto dos 42 professores do departamento. De cada professor é mostrado o nome (critério de ordenação), alguns dados relevantes e a área de conhecimento. Os usuários acessam a EA ‘Professores’, desistem, acessam a seção ‘áreas de conhecimento’ e selecionam a área, com a esperança de

encontrar ali os docentes daquela área. Frustrados, voltam à EA ‘Professores’ e começam o trabalhoso processo: ir observando a área de cada professor na EA; quando for a área desejada, acessar a página do professor e analisar o conteúdo, até achar um que trabalhe no tópico procurado.

O padrão (8) também se manifestou em ambos *sites* (2 usuários no *site A* e 1 no *site B*): uma vez na EA ‘Professores’, os usuários tentam formular consulta por área.

Problema: Os padrões (5) e (8) indicam que o usuário somente está interessado em um subconjunto dos objetos. O padrão (5) indica que é necessário processar os objetos do

subconjunto de forma sequencial. Os padrões (6) e (7) indicam que, para o usuário, faz sentido que a página de um objeto apresente informações sobre objetos vinculados.

O padrão 8, em que o usuário decide filtrar os objetos formulando consulta, não indica necessariamente um problema de projeto: Se a necessidade do contexto não foi prevista no levantamento de requisitos, pode estar indicando que a pessoa não pertence às classes de usuários previstas, ou é uma exceção que raramente ocorre. Mas, se estava prevista nos requisitos, denota um problema de negligência no projeto. E, se não estava prevista e acontece com frequência, então pode denotar um problema no levantamento dos requisitos (uma tarefa que não foi identificada, ou uma tarefa que surge a partir da utilização da aplicação, à medida que os usuários evoluem e surgem novas necessidades).

Solução: Todos estes problemas podem ser solucionados aplicando-se padrões de projeto: *Navegação Baseada em Conjuntos* para os padrões (6) e (8) e *Nó como uma Unidade Única de Informação* para os padrões (6) e (7).

O padrão *Navegação Baseada em Conjuntos* [Garrido, 1997] descreve como prover ao usuário sub-espacos de navegação fechados que possam ser navegados facilmente. Explica que existem tarefas em que o usuário realiza pesquisas e comparações, precisando lidar com conjuntos de objetos, que podem ser explorados de formas diferentes segundo a tarefa. Propõe considerar a navegação baseada em conjuntos como uma estratégia primária de navegação, e agrupar os objetos em conjuntos que sejam significativos segundo as tarefas projetadas. OOADM denomina estes conjuntos *Contextos de Navegação*, e constituem uma primitiva de projeto do método [Schwabe, 1998].

No caso do *site* de departamento acadêmico, é interessante agrupar professores segundo diferentes critérios. A solução mais simples é agrupar pelo fato de serem professores, ordenados pelo identificador mais forte (nome ou matrícula). Mas há muitas formas sob as quais os usuários precisam pesquisar professores; uma delas é por área: alunos que procuram professores para serem seus orientadores, empresários que procuram especialistas com o intuito de contratar consultorias ou treinamentos. É necessário fazer um levantamento das tarefas que serão apoiadas e determinar os conjuntos que facilitam a realização de cada uma.

O padrão *Nó como uma Unidade Única de Informação* [Lyardet, 1998] descreve como decidir o que constitui uma unidade de informação em um sistema hipermídia (nó), levando em consideração as tarefas que a aplicação pretende apoiar. Explica que, muitas vezes, devido à quantia reduzida de informação que pode ser exibida ao mesmo tempo, a unidade de informação precisa ser distribuída por nós diferentes. Assim, quando a quantia da informação disponível sobre um determinado tópico é grande, é necessário procurar um equilíbrio entre: (a) a quantia de informação apresentada ao usuário, (b) a relação entre o benefício de se obter a informação necessária e o custo de navegação envolvida no processo, e (c) a sobrecarga cognitiva associada. O padrão explica ainda que a organização da informação nunca deve ser restrita por como a informação vai ser exibida ao usuário. Ele propõe que uma unidade de informação navegacional (nó) seja “auto-suficiente” e faça sentido para um conjunto de usuários que executam certas tarefas em um determinado domínio. Um nó deve ser uma unidade única enfocada em um determinado tópico, de forma tal que toda a informação relevante para o tópico faça parte do mesmo nó. Deve haver uma separação clara entre a navegação através de tópicos diferentes e o modo como eles são exibidos.

Para determinar a relevância de determinados itens de informação com respeito a um tópico, há que se levar em consideração as tarefas que se pretende apoiar. Por exemplo, as

formas de navegação (6) e (7) indicam que, para os usuários, fazia sentido que houvesse na página da área informações sobre os professores vinculados àquela área. Uma análise prévia da tarefa de pesquisar professores por área teria indicado essa necessidade.

É claro que sempre vai existir uma tarefa para a qual é interessante que determinada informação faça parte de determinado tópico. Novamente vemos a importância do levantamento das tarefas e da classificação das mesmas. A aplicação deve apoiar aquelas tarefas mais importantes, as que são realizadas com mais frequência, para assim poder apoiar a maioria dos casos de uso.

O *site* do Departamento de Informática da PUC-Rio foi projetado com OOADM. Para decidir as informações de cada nó, foi aplicado o padrão *Nó como Unidade Única de Informação*.

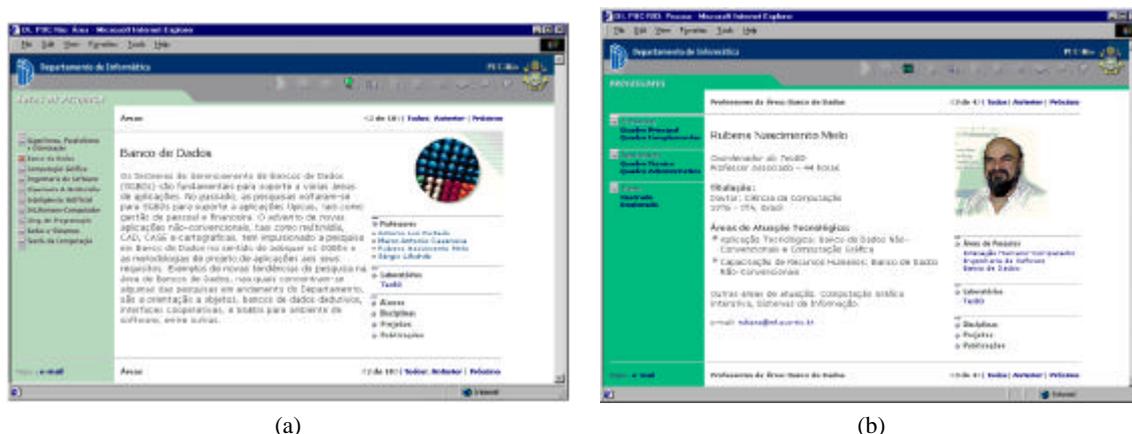


Figura 5 – Site DI PUC-Rio: (a) Página de uma área no contexto ‘Áreas do Departamento’ (b) Página de um professor no contexto ‘Professores da área Banco de Dados’.

Muitos nós no *site* são composição de diferentes objetos. Como mostra a Figura 5, toda área apresenta uma relação dos professores e laboratórios vinculados. Isto é importante, por exemplo, para tarefas onde se quer analisar as condições de trabalho na área. Da mesma maneira, para todo professor se apresenta a relação das áreas e laboratórios nos quais atua. Isto é importante para quando se analisa o trabalho do professor.

Os testes demonstraram que os usuários aproveitam esta forma oportunista de arrumar os itens de informação: dos 9 participantes, 4 entraram primeiro na página da área e logo começaram a processar os professores. Outros 4 entraram primeiro pela EA ‘Professores’ e desistiram: “*Aqui não disse nada de área, então vou ter que ir catando um a um*”; então acessaram a área e começaram analisar os professores. Somente 2 ficaram tentando outras vias (pesquisar todos os professores, pesquisar disciplinas, projetos ou publicações da área, formular consulta) até que finalmente decidiram pesquisar os professores através da área. Durante os testes, pesquisar os professores da área neste *site* foi muito mais rápido e confortável do que nos outros dois *sites* avaliados.

3.3.5 Navegação Intra-Conjunto

9. Na EA, acessar um objeto e, para selecionar outro objeto do mesmo conjunto, voltar à EA repetindo a mesma seqüência navegacional que tinha sido seguida até a EA, ou utilizar o comando ‘back’ (usuários experientes em *browser*).

Um exemplo deste caso foi observado nos testes aplicados ao site *Babybazaar.com*, loja que comercializa roupas de bebês. Parte da tarefa consistia em comprar dois suéteres de determinada cor (característica que podia ser observada na foto do produto na EA, Figura 6). Para acessar um suéter, o usuário precisava navegar a seguinte seqüência, partindo da *homepage*: selecionar seção ‘roupa de bebê’, selecionar a categoria ‘suéteres’ e selecionar

o produto. Para voltar à EA e selecionar outro suéter era necessário repetir a seqüência, pelo menos desde a página da seção ‘roupa de bebê’.

Durante os testes, a maioria dos usuários (7 de 9) repetiram a navegação desde a seção, e os outros 2 utilizaram as facilidades providas pelo *browser*: a opção ‘Back’ e a lista dos últimos endereços (URLs) visitados.

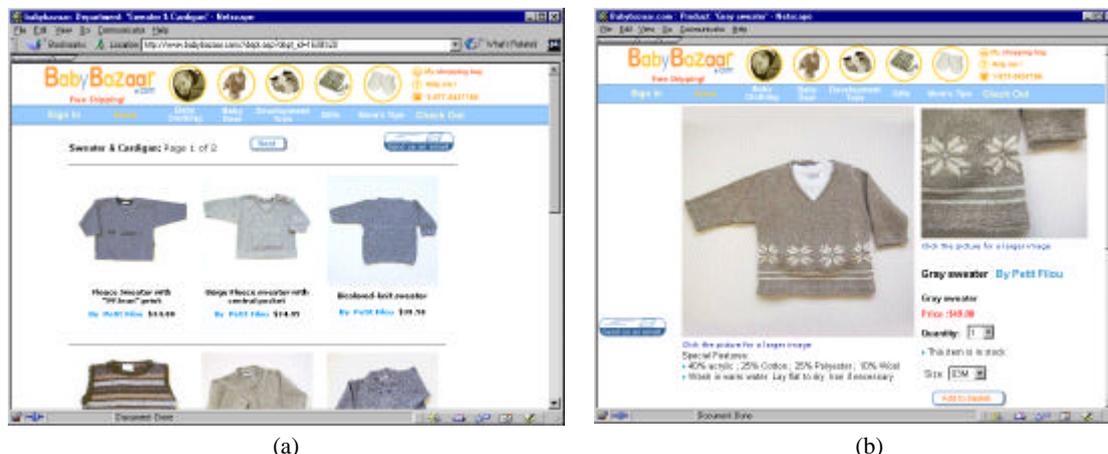


Figura 6 –Site BabyBazaar.com: (a) EA de produtos da categoria ‘suéteres’ (*sweaters*); (b) Página de um produto

Problema: Não oferece navegação intra-contextual.

Solução: Este problema poderia ter sido resolvido aplicando-se também o padrão *Navegação Baseada em Conjuntos* [Garrido, 1997].

O padrão explica que é desejável permitir ao usuário se mover facilmente de nó a nó. Explica que, como os elos entre os objetos de um conjunto normalmente não refletem relações semânticas, eles raramente são implementados, e que usualmente a estratégia seguida pelos projetistas é prover uma EA com os objetos do conjunto (como no exemplo do *site* Babybazaar.com). O padrão também explica que, a partir de um objeto, os usuários precisam voltar para o índice para poder navegar ao próximo membro do conjunto, perdendo, desta forma, a simplicidade habitual da metáfora de navegação do hipertexto. Ao prover ao usuário formas de navegação dentro dos conjuntos, complementa-se os relacionamentos semânticos convencionais. Assim o usuário pode navegar pelo conjunto ou pode deixá-lo para explorar outros objetos (ou eventualmente outros conjuntos).

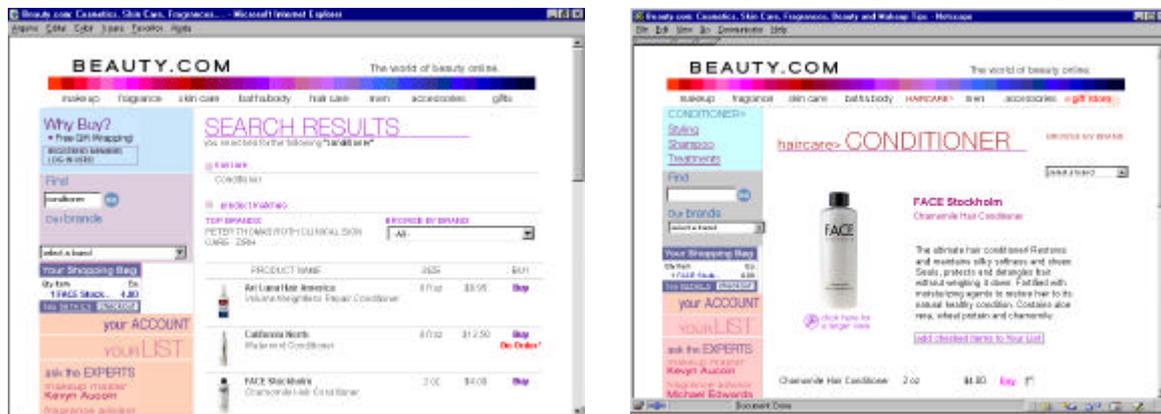
Alguns dos sites testados tinham facilidades para navegar dentro de conjuntos. Os testes demonstraram que os usuários tiram proveito destas facilidades: No *site* BabyStyle.com, onde a EA é mantida enquanto o produto é apresentado (Figura 8), todos os usuários utilizaram a EA para navegar até outro objeto do conjunto. No *site* do Departamento de Informática da PUC-Rio, que disponibiliza navegação seqüencial (âncoras ‘anterior’ e ‘próximo’) e por índice (*link* ‘todos’) para todos os contextos (Figura 5), 7 dos 9 usuários utilizaram as âncoras ‘anterior’ e ‘próximo’ (lembre-se que a tarefa exigia um processamento seqüencial dos professores). Somente 2 usuários navegaram através do índice, utilizando o comando ‘Back’, e quando foram questionados sobre isto responderam “Não utilizei anterior e próximo porque não os vi”, o que reflete um provável problema na interface, mas não na estrutura de navegação.

3.3.6 Feedback

10. Repetir várias vezes a mesma operação, achando que na tentativa anterior não conseguiu realizar a operação com êxito.

Este padrão foi observado nos testes aplicados aos *sites* Beauty.com e BabyStyle.com.

No site Beauty.com, há duas formas de inserir produtos na cesta de compras: (a) na EA resultante de uma consulta e (b) na página de qualquer produto. Para inserir, é necessário clicar no link ‘buy’ associado ao produto, e então o sistema volta a apresentar a página atual, atualizando uma visão simplificada da cesta que aparece na área esquerda da página ‘Sua Cesta de Compras’ (*Your Shopping Bag*). Esta área pode ser apreciada na Figura 7.



(a)

(b)

Figura 7 Site Beauty.com: (a) EA dos resultados da consulta (b) Página típica de produto

Durante os testes, os usuários inseriram na cesta um mesmo produto repetidamente, até perceber o *feedback* : “*Cliquei e não aconteceu nada. Voltei clicar e agora vejo que tem 3 aqui na lateral*”.

No site BabyStyle.com, pode-se inserir um produto na cesta a partir da página do produto. Quando isto é realizado, o sistema responde voltando a apresentar a página do produto com uma mensagem em fonte pequena, na parte superior da página, longe do botão ‘adicionar’ (*add*), que ativou o comando, como ilustrado na Figura 8. Caso tenha sido possível inserir o produto, a mensagem seria de confirmação ‘1 item foi adicionado à sua cesta’ (*1 item has been added to your basket*). Caso contrário, a mensagem seria de advertência ‘Por favor selecione um item. O item foi adicionado à sua cesta.’ (*Please select an item. 0 items have been added to your basket.*).



(a)

(b)

Figura 8 – Site BabyStyle.com: mensagens de retorno (a) advertência (b) confirmação

Durante os testes, muitos usuários demoram em perceber a mensagem de *feedback*: “*Não sei se já inseriu...*”; “*Eu cliquei no botão e não sei se colocou na cesta.*”; “*Apertei ‘buy’ achando que ia comprar mas ele volta a apresentar a tela do produto e não notei nenhuma mudança*”;

um fez rolar a página e disse “*Engraçado, para mim não está claro... ah, tá, adicionou*”. Na dúvida, inserem o mesmo produto novamente.

Problema: Os usuários não percebem o *feedback* do processo, duvidam se conseguiram realizar uma operação e a repetem. Consideramos que o *feedback* não é percebido por causa do projeto de interface (no caso, a fonte utilizada e a posição na tela).

11. Verificar cada operação de atualização: depois de aplicar a operação, navegar até o objeto da atualização para conferir.
--

Este padrão foi observado nos testes aplicados ao *site* BabyStyle.com. A tarefa consistia em comprar vários produtos. Durante os testes, os usuários iam inserindo os produtos. Uns poucos usuários estavam confiantes de que estavam conseguindo inserir os produtos na cesta. Eles perceberam logo as mensagens de retorno, e no final do processo, depois de ter inserido todos os produtos, foram verificar na cesta de compras.

Alguns usuários (4 de 9 participantes), porém, não perceberam as mensagens de retorno ou não entendiam a mensagem de confirmação: “*Disse que inseriu 1 produto, se eu já tinha inserido outro?*”, “*Por que se agora são 2 produtos na cesta a mensagem continua sendo ‘1 item ...’?*”, ou consideravam que a mensagem era insuficiente “*Disse que acrescenta 1, mas não disse quanto está dando a compra, gostaria de saber o subtotal*”. De repente, estes usuários paravam de escolher produtos e decidiam navegar até a cesta para verificar o que estava acontecendo.

Na cesta, alguns usuários percebiam que, de fato, os produtos não correspondiam ao esperado. Havia menos produtos pois, às vezes, não tinham conseguido inserir e, como eles não percebiam a mensagem de advertência, continuavam comprando. A partir desta constatação, a maioria destes usuários atuaram de forma similar: repetiram o processo de compras, verificando a cesta em cada inserção.

Problema: Quando usuários não percebem ou não entendem o *feedback* do processo, sentem-se inseguros. Alguns repetem a operação e acessam o objeto alvo do processo para verificar o que houve.

Solução: Estes problemas poderiam ter sido contornados aplicando-se o padrão *Feedback de Processos* [Garrido, 1997], [Lyardet, 1998]. Este padrão descreve como manter ao usuário informado sobre o estado de uma interação de tal modo que ele saiba o que esperar. Quando um usuário interage com uma aplicação, é preciso que a aplicação comunique ao usuário se a operação foi concluída com êxito ou não. Caso contrário, “O usuário pode sentir que ele não selecionou a operação certa, ou que ele errou, ou inclusive que o sistema não está funcionando bem. Pode perder a paciência e selecionar a mesma ou outra opção...” Os testes evidenciaram que efetivamente os usuários ficam com temores e tomam atitudes: voltam a inserir o produto, acessam a cesta para verificar. O padrão recomenda prover *feedback* constante e perceptível para operações ‘não atômicas’ (informar sobre o começo, o progresso e o fim da operação) e considera que operações ‘atômicas’ não necessitam ser rastreadas.

Consideramos que toda interação onde o usuário demande um processamento do sistema precisa de *feedback*, e que este *feedback*, além de ser perceptível, deve informar de forma clara o resultado da operação e, caso tenha acontecido algum problema, deve explicar as causas e providências a serem tomadas antes de tentar aplicar a operação novamente. Por exemplo, para a operação de inserir produtos na cesta de compras, seria importante informar quantos produtos há na cesta e qual o valor total da fatura até o momento.

É interessante comentar que, quando projetamos os testes, consideramos que a operação de inserir produtos na cesta seria mais cômoda justamente nos *sites* Beauty.com e

BabyStyle.com. Nos outros *sites*, Walmart.com e Babybazaar.com, a cada inserção o sistema navega até a cesta. Como previmos, isto foi ruim do ponto de vista navegacional: alguns usuários ficaram desorientados sem saber como continuar comprando (passam o mouse sobre cada botão da cesta, “*Não sei o que fazer, não quero atualizar cesta, nem remover, quero continuar comprando. Se escolho outra opção a cesta continuará cheia?*”; “*Não posso voltar? Não? Se eu dou ‘back’ des-compro? Falta um botão ‘continuar comprando’*”; “*Agora eu queria é voltar*”). Porém, os usuários se sentiram mais seguros com relação à operação inserir (“*Aqui é melhor porque quando compras ele te mostra o subtotal*”, “*Aqui é mais legal, mostra que você já tem 1 item na cesta*”).

4. Conclusões

O método de avaliação proposto resultou da constatação de que diferentes usuários atuam de forma similar ante dificuldades de igual natureza, independentemente da tarefa ou do domínio do *site*. Eles seguem seqüências navegacionais e padrões de interação semelhantes quando há um mesmo problema de projeto. Situações de erro típicas foram classificadas e tabuladas, constituindo uma ferramenta para diagnóstico e solução de problemas de projeto.

Os testes realizados permitiram conferir as hipóteses sobre o benefício de se ter um método de desenvolvimento de aplicações *Web* baseado em modelos: houve um melhor desempenho em *sites* que utilizavam soluções-padrão; e houve problemas de usabilidade em *sites* cuja solução *naif* não respeitava os princípios de projeto.

O maior benefício deste método de avaliação resulta do fato de se basear em modelos: um projetista proficiente em OOHD, mas com pouca ou nenhuma experiência em avaliação de usabilidade, consegue detectar problemas e apontar soluções bastante precisas. Estas soluções se baseiam não apenas no produto, mas também no processo que resultou naquele produto. É possível apontar onde, naquele processo, houve uma falha. Além disto, é possível prever as conseqüências das modificações sugeridas, o que geralmente não ocorre em outros métodos de avaliação, podendo acarretar efeitos colaterais imprevisíveis e indesejados.

Com a incorporação do nosso método, o OOHD se apresenta agora como uma metodologia de análise, projeto, implementação e avaliação de aplicações hiper-mídia. Demos o primeiro passo para descrever os ciclos de realimentação que ocorrem durante o processo de desenvolvimento segundo o OOHD, e já encontramos algumas oportunidades para trabalhos futuros.

O estudo de caso apresentado aponta para a necessidade de mais testes com usuários, visando não apenas refinar o método de avaliação, mas também abranger mais classes de problemas.

Grande parte das sugestões de reprojeto podem ser justificadas com base nos padrões de projeto e diretrizes utilizadas. Entretanto, algumas decisões necessitam de conhecimento sobre o *design rationale* da aplicação. Um próximo passo será verificar a influência do *design rationale* no processo de desenvolvimento, e em particular seu papel nas etapas de avaliação.

5. Referências

- [Borchers, 2001] Borchers, J.: *A Pattern Approach to Interaction*. John Wiley & Sons, 2001.
- [Ceri, 2000] S. Ceri, P. Fraternali, A. Bongio. *Web Modeling Language (WebML): a Modeling Language for Designing Web Sites*. Proc WWW9 Conference, Amsterdam, NL, May 2000 (also in Computer Networks, 33 (2000), pp. 137-157).
- [CreativeGood, 2000] CreativeGood: *The Dotcom Survival Guide*, [June, 2000.](http://www.creativegood.com/holiday2000/) (<http://www.creativegood.com/holiday2000/>)
- [de Souza, 1993] de Souza, C.S. The Semiotic Engineering of User Interface Languages. *International Journal of Man-Machine Studies*. No. 39. pp. 753-773. 1993.
- [De Troyer, 1998] De Troyer, O.; Leune, C.; *WSDM: A User-Centered Design Method for Web Sites*, in Computer Networks and ISDN systems, Proc. 7th International WWW Conference, Elsevier, pp. 85-94.
- [Garrido, 1997] A. Garrido; G. Rossi; D. Schwabe: *Pattern Systems for Hypermedia*, Proceedings of Plof'97, Pattern Language of Programming, 1997.
- [Garzotto, 2000] Garzotto, F.; Baresi, L.; Paolini, P.; *From Web Sites to Web Applications: New Issues for Conceptual Modeling*. In *Conceptual Modeling for E-Business and the Web*. Proceedings of the 2nd International Workshop on The World Wide Web and Conceptual Modeling. S.W. Little, H. C. Mayer, and B. Thalheim (eds.) Lecture Notes in Computer Science 1921, Springer, 2000.
- [Garzotto, 1999] F. Garzotto, P. Paolini, D. Bolchini, S. Valenti: *Modeling-by-Patterns of Web Applications*; Proc. International Workshop on the World Wide Web and Conceptual Modeling, WWW CM'99, Paris, Novembro, 1999.
- [Garzotto, 1998] Garzotto, F.; Matera, M.; Paolini, P.: *To Use or Not to Use? Evaluating Usability of Museum Web Sites*, Museums and the Web 1998: Conference Proceedings, CD-ROM documenting the Second International Conference held in Toronto, Ontario, Canada, April 21-25, 1998. Pittsburgh, PA: Archives & Museum Informatics.
- [Gomez, 2001] J. Gómez, C. Cachero, O. Pastor. [*Conceptual Modeling of Device-Independent Web Applications*](#). IEEE Multimedia Special Issue on Web Engineering (Part II). Volume 8 Number 2. April-June 2001. Pags 26-39. IEEE Computer Society.
- [Güell, 2000] Güell, N.; Vilain P.; Schwabe, D.: *Cenários, Use Cases e Diagramas de interação do usuário no Projeto da Navegação de Aplicativos Hipermídia*, SBMidia'2000, junho 2000.
- [Lyardet, 1998] F. Lyardet; G. Rossi; D. Schwabe.: *Using Design Patterns in Educational Multimedia Applications*, AACE EDMEDIA Conference, Freiburg Germany, 1998.
- [Nielsen, 1993] Nielsen J., *Usability Engineering*, Academic Press, New York, 1993.
- [Nielsen, 1997] Nielsen, J.: *Usability Testing*. Salvendy, G. (ed.): Handbook of Human Factors and Ergonomics. 2nd edition. New York, NY: John Wiley & Sons. 1997.
- [Nielsen, 2000] Nielsen, J.: *Test With 5 Users*. Alertbox, March 19, 2000.
- [Prates, 2000] Prates, R.O., de Souza, C.S., Barbosa, S.D.J. *A Method for Evaluating the Communicability of User Interfaces*. *ACM Interactions*, 31-38. Jan-Feb 2000.
- [Rossi 97] Rossi, G; Schwabe, D.; Garrido, A. *Design Reuse in Hypermedia Application Development*. Anais do *Hypertext'97*. Southampton, Grã-Bretanha. 1997. ACM. NY, EUA. p.57-66. 1997.
- [Rossi 99] Rossi, G.; Schwabe. D.; Lyardet, F.; *Integrating Patterns into the Hypermedia Development Process*, New Review of Hypermedia, Vol. 5, 1999, Taylor Graham Publishers pp. 59-80.
- [Schwabe, 1998] Schwabe, D. and Rossi, G., An Object Oriented Approach to Web-Based Application Design, Theory and Practice of Object Systems 4(4), 1998. Wiley and Sons, New York.
- [Schwabe, 1999] Schwabe, D.; Pontes, R. A.; Moura, I.; *OOHDM-Web: An Environment for Implementation of Hypermedia Applications in the WWW*, SigWEB Newsletter, Vol. 8, #2, Junho de 1999.