Recomendacões de Usabilidade

Fases de projeto: Avaliação

RECOMENDAÇÕES DE USABILIDADE

Os primeiros estudos relacionados ao uso de computadores por seres humanos datam da década de 1980, de lá para cá, muito tem sido feito na área de IHC. A importância da interação com as interfaces computacionais naquela época, não se mostrava tão evidente como nos dias atuais. Hoje os computadores são utilizados em praticamente todas as áreas, fazendo da interface, o principal meio de utilização dos sistemas por usuários comuns. Por esse motivo, pesquisas se voltam cada vez mais para as questões relativas às interfaces digitais.

A área de Usabilidade de Software tem apresentado crescimento considerável nas pesquisas. Atualmente existem muitos estudos de autores renomados internacionalmente que tratam de recomendações de usabilidade. Nielsen, Shneiderman, Bastien e Scapin, são alguns desses autores e estão inclusos nesta pesquisa.

A usabilidade de um sistema é atingida quando recomendações de usabilidade são obedecidas desde o projeto inicial. Quando isso ocorre, o sistema apresenta atributos relacionados à usabilidade como a facilidade de aprendizado, a eficiência de uso, a facilidade de memorização, a baixa taxa de erros e a satisfação subjetiva.

IMPORTÂNCIA DAS RECOMENDAÇÕES DE USABILIDADE

As recomendações de usabilidade são tão importantes como a própria usabilidade, pois é por meio daquelas que esta pode ser alcançada. De acordo com Andrade (2003), o desenvolvimento de interfaces gráficas deve seguir recomendações e critérios ergonômicos e da comunicação visual.

Santos (2006) destaca que interfaces desenvolvidas sem o atendimento aos requisitos de usabilidade levam a uma performance deficiente e a uma redução da qualidade da interação do usuário com um aplicativo.

Os projetos onde as recomendações de usabilidade são obedecidas têm maiores chances de serem bem elaborados e estruturados, colaborando para o menor esforço do usuário, ao diminuir, por exemplo, a realização de tarefas repetitivas e ao evitar que este se perca na navegação do sistema.

As recomendações são um importante guia para desenvolvedores de sistemas ao projetarem, testarem e finalizarem os softwares. Elas podem, e devem ser aplicadas desde o início do projeto, construindo estruturas e layouts que considerem aspectos de bom funcionamento, agradabilidade, facilidade de aprendizado, entre outros fatores. As recomendações também podem ser utilizadas para testar e avaliar os softwares, apontando situações inadequadas e propícias ao erro.

TERMOS EMPREGADOS PARA RECOMENDAÇÕES DE USABILIDADE

Vários são os termos empregados para representar recomendações de usabilidade e esta variedade costuma confundir estudantes e profissionais, como afirmam Preece et al.:

Os vários termos propostos para descrever os diferentes aspectos da usabilidade podem ser confusos. Geralmente são intercambiáveis e apresentam combinações diferentes. Algumas pessoas falam sobre princípios do design da usabilidade; outras sobre conceitos do design. (2005, p. 50).

Muitos são os termos utilizados para apresentar recomendações que guiem o projeto de interfaces

computacionais para melhorar a usabilidade.

Entre eles, podem ser apresentados: Critérios Ergonômicos de Usabilidade (Bastien e Scapin, 1993); Diálogo Homem-Máquina (Dul e Weerdmeester, 1991); Regras de Ouro (Shneiderman, 2005); Princípios Gerais e Heurísticas (Nielsen, 1994); Guidelines ou Guia de Estilos (Dias, 2007 e Winckler, 2001); Princípios de Usabilidade (Jordan, 1998 e Preece et al., 2005);

O Quadro 1, retirado de Preece et al., (2005, p. 50), explana as

diferenças sobre alguns dos termos utilizados para recomendações de usabilidade.

http://www.faac.unesp.br/posgraduacao/design/dissertacoes/pdf/mileni_kaz... [1]

Recomendações de usabilidade de Bastien & Scapin

Embora, sendo um dos materiais mais antigos, ainda sim, bastante completo. Cabe algumas atualizações nos exemplos.

Este documento é uma tradução livre das **Recomendações de usabilidade apresentadas por Bastien & Scapin (1993)**

BASTIEN, C. e SCAPIN, D. **Ergonomic Criteria for the Evaluation of Human Computer Interfaces**. INRIA, 1993.

Apresentação: A abordagem ergonômica em IHC deve estar baseada em oito critérios, os quais são apresentados de modo a identificar e classificar as qualidades e problemas ergonômicos do software interativo, a saber:

1. Condução Definição: A Condução se refere aos meios para advertir, orientar, informar, instruir e guiar o usuário na interação com o computador (mensagens, alarmes, rótulos, etc.). O critério de Condução é subdivido em quatro critérios: Orientação, Agrupamento/Distinção de Itens, Feedback imediato e Legibilidade.

Justificativas:

A boa condução facilita o aprendizado e uso do sistema por permitir aos usuários: saber, a qualquer hora, onde se encontra, numa seqüência de interações ou na execução de uma tarefa; conhecer as ações permitidas, bem como suas conseqüências; obter informações adicionais (eventualmente por demanda). A facilidade de aprendizado e de uso que acompanha a boa condução permite a melhoria do desempenho e redução dos erros.

1.1. Orientação Definição: O termo Orientação tem aqui uma definição maior que aquela que lhe é conferida geralmente. Este critério se refere também a todos os mecanismos ou meios utilizados para ajudar os usuário a saber as alternativas, quando várias ações são possíveis, dependendo do contexto. Orientação também diz respeito ao status do sistema que são informações que permitem aos usuários saber onde estão, informando sobre o status do sistema, bem como informações sobre a ajuda e o seu acesso.

Justificativas:

Uma boa orientação guia o usuário e poupa do aprendizado de uma série de comandos. A boa orientação permite também ao usuário saber exatamente o modo ou o estado que se encontra o sistema, bem como o que fez para se encontrar nessa situação. Uma boa orientação facilita, então, a navegação no aplicativo e ajuda a diminuir a ocorrência de erros.

Exemplos de recomendações:

- Guiar as entradas e saídas e indicar a forma adequada e os valores aceitáveis, por exemplo, incluir num campo um rótulo adicional com formato de data (por exemplo, Data (dia/mês/ano): __/__/__). - Apresentar unidades de medida para entrada de dados;

- Indicar todos as informações de estado (por exemplo, modos, valores, etc.). - Para cada campo, apresentar o rótulo associado. - Fornecer comprimento (quantidade de caracteres permitida) para entrada de dados. - Fornecer um título para cada janela. - Fornecer ajuda online.216

1.2. Agrupamento/Distinção de Itens Definição: O critério Agrupamento/Distinção de Itens diz respeito à organização visual dos itens de informação, relacionados uns com os outros. Este critério leva em conta a topologia (localização) e certas características gráficas (formato) para indicar se pertencem ou não a uma mesma classe de itens, ou também para indicar diferenças entre as classes. Este critério também diz respeito a organização dos itens dentro de uma mesma classe. O critério Agrupamento/Distinção de Itens é subdivido em dois critérios: agrupamento/distinção por localização e agrupamento/distinção por formato.

Justificativas:

A compreensão de uma tela pelo usuário depende, entre outras coisas, da ordem, do posicionamento, e da distinção dos objetos (imagens, textos, comandos, etc.) que são apresentados. Os usuários vão detectar os diferentes itens ou grupos de itens e aprender suas relações mais facilmente, se, por um lado, eles forem apresentados de uma maneira organizada (por exemplo, ordem alfabética, freqüência de uso, etc.), e se, por outro lado, os itens forem apresentados em formatos, ou codificados de maneira a indicar suas similaridades ou diferenças. Dessa forma a aprendizagem e a memorização de itens ou de grupos de itens é melhor. O Agrupamento/Distinção de Itens melhora a condução.

1.2.1 Agrupamento/Distinção de Itens por localização Definição: O critério agrupamento/distinção por localização diz respeito ao posicionamento relativo dos itens, estabelecido para indicar se eles pertencem ou não a uma dada classe, ou ainda, para indicar diferenças entre classes. Este critério também diz respeito ao posicionamento relativo dos itens dentro de uma mesma classe.

Justificativas:

A compreensão de uma tela pelo usuário depende, entre outras coisas, da ordem, do posicionamento, e da distinção dos objetos (imagens, textos, comandos, etc.) que são apresentados. Os usuários vão detectar os diferentes itens se eles forem apresentados de uma maneira organizada (por exemplo, ordem alfabética, freqüência de uso, etc.). Dessa forma a aprendizagem e a memorização de itens será melhorada. O Agrupamento/Distinção de Itens melhora a condução.

Exemplos de recomendações:

- Organize os itens em listas de hierarquia. - Agrupe as opções de menu em função dos objetos na qual eles se aplicam. - Quando muitas opções forem apresentadas, sua organização deve ser lógica (por exemplo, ordem alfabética, funcional, freqüência de uso, etc.).

1.2.2 Agrupamento/Distinção de Itens por formato Definição: O critério agrupamento/distinção por formato diz respeito mais precisamente às características gráficas (formato, cor, etc.) que indicam se itens pertencem ou não a uma determinada classe, ou que indicam distinções entre as classes diferentes, ou ainda distinções entre itens de uma mesma classe.

Justificativas:

Será mais fácil para o usuário saber a relação entre itens ou classes de itens, se diferentes formatos ou diferentes códigos ilustrarem suas similaridades ou diferenças. Tais relacionamentos serão mais fáceis de aprender e de lembrar. Um bom agrupamento/distinção de itens por formato melhora a condução.

Exemplos de recomendações:

- Estabelecer uma distinção visual de áreas que possuem diferentes funções (comandos, mensagens, etc.). - Estabelecer uma distinção visual entre os campos e seus rótulos.

1.3 Feedback imediato Definição: O critério Feedback imediato diz respeito às respostas do sistema com relação às ações do usuário. Estas ações podem ser um simples pressionar de uma tecla, até uma transação complexa como uma lista de comandos. Em todos os casos o computador deve fornecer feedback, e este deve ser rápido, com um tempo de resposta apropriado e consistente para cada tipo de transação. Em todos os casos, uma resposta rápida deve ser fornecida com informação sobre a transação solicitada e seu resultado.

Justificativas:

A qualidade e rapidez do feedback são dois fatores importantes para o estabelecimento de satisfação e confiança do usuário, bem como para o compreender o diálogo. Estes fatores permitem aos usuários ter um melhor entendimento do funcionamento do sistema.

A falta de feedback ou a demora de feedback podem ser desconcertantes para o usuário. Os usuários podem suspeitar de uma falha no sistema, e podem tomar atitudes prejudiciais para os processos em andamento.

Exemplos de recomendações:

- Todas as entradas devem ser apresentadas exceto as entradas de segurança (senhas). Entretanto, neste caso, todas as entradas devem produzir um feedback perceptível, por exemplo, uso de símbolos como asteriscos. - Depois de uma interrupção feita pelo usuário, deve ser apresentada uma mensagem assegurando que o sistema voltará ao estado anterior.

- No caso de processamentos longos, o sistema indicar ao usuário que o processamento está em curso.

1.4 Legibilidade Definição: Legibilidade diz respeito às características lexicais das informações apresentadas na tela que possam dificultar ou facilitar a leitura desta informação (brilho do caractere, contraste letra e fundo, tamanho da fonte, espaçamento entre palavras, espaçamento entrelinhas, espaçamento de parágrafos, comprimento da linha, etc.). Por definição o critério Legibilidade não diz respeito ao feedback ou mensagens de erro.

Justificativas:

A performance melhora quando a apresentação da informação na tela leva em conta as características cognitivas e perceptivas dos usuários. Uma boa legibilidade facilita a leitura da informação apresentada. Por exemplo, letras escuras em um fundo claro são mais fáceis de ler do que o contrário; um texto apresentado com letras maiúsculas e minúsculas é lido mais rapidamente do que texto escrito somente com maiúsculas.

Exemplos de recomendações:

- Títulos devem ser centralizados. - Rótulos devem ser apresentados em letras maiúsculas. - Cursores devem ser facilmente percebidos. - Quando o espaço para apresentação de texto é limitado, é preferível apresentar poucas linhas com texto longo do que muitas linhas com texto curto. - As linhas de textos contínuos devem ter no máximo 50 caracteres.

- A justificação de textos deve ser empregada se puder ser aplicado espaçamento variável, de forma que um espaçamento proporcional constante entre as letras e as palavras seja respeitado. - Em apresentação de textos utilize o mínimo possível de palavras hifenizadas.

2. Carga de trabalho Definição: O critério Carga de trabalho diz respeito a todos elementos da interface que têm um papel importante na redução da carga cognitiva e perceptiva do usuário, e no

aumento da eficiência do diálogo. O critério Carga de trabalho está subdivido em dois critérios: Brevidade (que inclui Concisão e Ações Mínimas) e Densidade Informacional.

Justificativas:

Quanto maior for a carga de trabalho cognitivo, maior será a probabilidade de se cometer erros. Além disso, quanto menos o usuário se distrair com informações desnecessárias, estará mais capacitado a desempenhar suas tarefas com eficiência. Além do mais, quanto menos ações forem solicitadas, mais rápidas são as interações.

2.1 Brevidade Definição: O critério Brevidade diz respeito à carga de trabalho perceptivo e cognitivo do usuário, tanto para entradas e saídas individuais, quanto para conjuntos de entradas (por exemplo, um conjunto de ações necessárias para completar um objetivo ou uma tarefa). Brevidade corresponde ao objetivo de limitar a carga de trabalho de leitura e entradas, e o numero de passos. O critério Brevidade se divide em dois critérios: Concisão e Ações Míni mas.

Justificativas:

A capacidade de memória de curto prazo é limitada. Conseqüentemente, quanto menos entradas, menor a probabilidade de cometer erros. Além disso, quanto mais sucintos forem os itens, menor será o tempo de leitura. Quanto mais numerosos e complexas forem as ações necessárias par se chegar a uma meta, maior será a carga de trabalho e a probabilidade de ocorrência de erros.

2.1.1 Concisão Definição: O critério concisão diz respeito à carga de trabalho no nível perceptivo e cognitivo de saídas e entradas individuais. Por convenção, Concisão não diz respeito a feedback ou mensagens de erro.

Justificativas:

A capacidade de memória de curto prazo é limitada. Conseqüentemente, quanto menos entradas, menor a probabilidade de cometer erros. Além disso, quanto mais sucintos forem os itens, menor será o tempo de leitura.

Exemplos de recomendações:

- Para dados numéricos, não deve ser necessário o uso ‘0’ antes dos números. - Se os códigos forem maiores que 4 ou 5 caracteres deve se usar abreviaturas. - Permitir aos usuários entrada de dados curtos. - Quando uma unidade de medida é associada a um determinado campo, deve-se colocar a medida como rótulo, ao invés de solicitar ao usuário que a coloque.

2.1.2 Ações Mínimas Definição: O critério Ações Mínimas diz respeito à carga de trabalho com relação ao número de ações necessárias para completar um objetivo ou tarefa. Trata-se de limitar ao máximo o número de passos que o usuário precisar para continuar a tarefa.

Justificativas:

Quanto mais numerosas e complexas forem as ações necessárias para se chegar a uma meta, maior será a carga de trabalho e a probabilidade de ocorrência de erros.

Exemplos de recomendações:

- Minimizar o número de passos necessários para selecionar um item de menu. - Não solicitar uma entrada de dado ao usuário quando ela puder ser fornecida pelo computador. - Evitar entradas de comandos que incluem pontuação. - Para salvar dados, apresentar valores padrões em campos apropriados. - Para documentos com muitas páginas, deverá ser possível encontrar uma página sem ter que percorrer todas as páginas uma a uma.

2.2 Densidade informacional Definição: O critério Densidade Informacional diz respeito à carga de trabalho do usuário, do ponto de vista perceptivo e cognitivo, com relação ao conjunto total de

itens de informação apresentados aos usuários, e não a cada elemento ou item individual.

Justificativas:

Na maioria das tarefas, a performance dos usuários é diminuída quando a densidade da informação é muito alta ou muito baixa: nestes casos, a ocorrência de erros é mais provável. Itens que não estão relacionados à tarefa devem ser removidos. A carga de memória do usuário deve ser minimizada. Usuários não devem ter que memorizar listas de dados ou procedimentos complicados (a memória de curto prazo é limitada). Eles não devem precisar executar tarefas cognitivas complexas quando estas não estão relacionadas com a tarefa em questão.

Exemplos de recomendações:

- Limite a densidade informacional na tela, apresentando somente as informações necessárias. - As informações não devem precisar de conversões. - Não solicitar aos usuários que se lembre de dados precisos de uma janela para outra.

- Os dados que podem ser calculados a partir das saídas pelo usuário devem ser feitos automaticamente. Não se deve exigir que o usuário efetue cálculos que podem ser feitos pelo computador.

3. Controle explícito Definição: O critério Controle explícito diz respeito tanto ao processamento de ações explicitas do usuário, como do controle que os usuários têm sobre o tratamento de suas ações. O critério Controle explícito se subdivide em dois critérios: Ações explícitas do usuário e Controle do Usuário.

Justificativas:

Quando os usuários definem explicitamente suas entradas, e quando estas entradas estão sob seu controle, os erros e as ambigüidades são limitados. Além disso, o sistema será mais bem aceito pelos usuários se eles tiverem controle sobre o diálogo.

3.1 Ações explícitas do usuário Definição: O critério Ações explícitas do usuário se refere às relações entre o processamento pelo computador e as ações do usuário. Esta relação deve ser explícita, como por exemplo, o computador deve processar somente aquelas ações solicitadas pelo usuário e somente quando solicitado a fazê-lo.

Justificativas:

Quando o processamento pelo computador resulta de ações explícitas dos usuários, estes aprendem e entendem melhor o funcionamento da aplicação, e menos erros são observados.

Exemplos de recomendações:

- O sistema deve solicitar ao usuário uma ação explícita para iniciar um processamento de entrada de dado; não iniciar o processamento como efeito (como atualizar um dado) de alguma outra ação (como imprimir um arquivo). - Se a seleção de menu é feita por cursor de mouse, elabore uma ação explicita de validação para ambas ações: uma para a seleção do mouse e outra para o clique.

- As entradas de comando devem ser terminadas com uma ação de ENTER, acompanhada de facilidades de edição.

3.2 Controle do usuário Definição: O critério controle do usuário se refere ao fato de que os usuários devem estar sempre no controle do processamento do sistema (como interromper, cancelar, suspender e continuar). Cada ação possível do usuário deve ser antecipada e opções apropriadas devem ser oferecidas.

Justificativas:

O controle sobre as interações favorece a aprendizagem e assim diminui a probabilidade de erros. Como conseqüência, o computador se torna mais previsível.

Exemplos de recomendações:

- Permitir aos usuários o controle do ritmo de suas entradas, ao invés do ritmo ser controlado pelo sistema ou por eventos exteriores. - O cursor não deve se movimentar automaticamente sem controle do usuário (exceto para procedimentos estáveis e bem conhecidos como preenchimento de formulários). - As páginas não devem ser mudadas sem o controle do usuário. - Permitir aos usuários interromper ou cancelar a qualquer momento as ações ou processos em curso. - Fornecer a possibilidade de desistência do cancelamento em curso e fornecer a possibilidade de restaurar a situação anterior.

4. Adaptabilidade Definição: A adaptabilidade de um sistema diz respeito à sua capacidade de se comportar conforme o contexto, e conforme as necessidades e preferências do usuário. O critério adaptabilidade se subdivide em dois critérios: a Flexibilidade e a consideração da experiência do usuário.

Justificativas:

Quanto mais variadas são as maneiras de realizar uma tarefa, maiores são as chances do usuário de escolher e dominar uma delas no curso de seu aprendizado. Deve-se, portanto, fornecer ao usuário procedimentos, opções, comandos diferentes permitindo alcançar um mesmo objetivo.

Além disso, uma interface não pode atender ao mesmo tempo a todos os seus usuários em potencial. Para que não tenha efeitos negativos sobre o usuário, a interface deve, conforme o contexto, se adaptar a ele.

4.1 Flexibilidade Definição: O critério flexibilidade se refere aos meios colocados à disposição do usuário que permite customizar a interface a fim de levar em conta suas estratégias ou seus hábitos de trabalho e as exigências da tarefa. Flexibilidade corresponde também ao número de diferentes maneiras à disposição do usuário para alcançar um dado objetivo, em outras palavras, a capacidade da interface se adaptar as variadas ações dos usuários.

Justificativas:

Quanto mais formas de efetuar uma tarefa existirem, maiores serão as chances de que o usuário possa escolher e dominar uma delas no curso de sua aprendizagem.

Exemplos de recomendações:

- Quando as exigências dos usuários são imprecisas, forneça ao usuário certa liberdade para controlar a configuração das apresentações. - Quando os designers de interface não podem prever quais valores padrões serão úteis, permita aos usuários definir, mudar ou remover esses valores.

- Quando algumas apresentações forem desnecessárias, os usuários devem poder removê-las temporariamente. - A seqüência de entrada de dados deve poder ser modificada para que se adapte às preferências dos usuários. - Quando não se pode especificar o formato de um documento, deve-se permitir aos usuários defini-lo e salvá-los para uma utilização posterior. - Deve-se permitir aos usuários que coloquem nomes para campos de dados que eles tenham criado.

4.2 Experiência do usuário Definição: O critério Experiência do usuário diz respeito aos meios implementados que permitem que o sistema respeite o nível de experiência do usuário.

Justificativas:

Usuários experientes e inexperientes têm diferentes necessidades. Pode-se fornecer aos usuários inexperientes diálogos bem conduzidos, ou mesmo passo a passo. Para usuários experientes, os diálogos de iniciativa somente do computador entediam e diminuem o seu rendimento; atalhos

podem permitir a eles acesso às funções do sistema mais rapidamente. Diferentes níveis de interação devem levar em conta a experiência do usuário.

No entanto, a maioria dos sistemas terá usuários com variações no grau de experiência. Os usuários podem se tornar especialistas, devido à utilização continuada, ou menos especialistas, depois de um longo período de não-utilização. A interface deve também ser projetada para lidar com as variações do nível de experiência.

Exemplos de recomendações:

- Permitir aos usuários desviar de uma série de seleção de menus, fazendo um comando equivalente ou um atalho de teclado direto. - Permitir aos usuários experientes realizar uma série de comandos ao mesmo tempo, e aos usuários inexperientes de modo passo a passo. - Tipos de diálogos devem ser projetados para atender as necessidades dos diferentes usuários.

- Permitir diferentes modos de diálogo correspondente aos diferentes grupos de usuários (por exemplo, ofereça orientação como característica opcional que pode ser selecionada para usuários novatos, mas omitida por usuários experientes). - Técnicas adotadas para guiar usuários inexperientes podem diminuir sua velocidade, para isso, forneça alternativas para permitir que o usuário consiga desviar destes procedimentos. - Em mensagens de erro, permita aos usuários que saibam maiores detalhes do erro com linguagem adaptada ao seu nível de conhecimento.

5. Gestão de erros Definição: O critério Gestão de erros se refere a todos os meios que permitem evitar ou reduzir a ocorrência de erros, e quando eles ocorrem, que favoreçam sua correção. Os erros são aqui considerados como entradas de dados incorretas, entradas com formatos inadequados, entradas de comandos com sintaxes incorretas, etc. O critério Gestão de erros é subdivido em três critérios: Proteção contra os erros, Qualidade das mensagens de erro e a Correção dos erros.

Justificativas:

As interrupções provocadas pelos erros têm conseqüências negativas sobre a atividade do usuário. Em geral, elas prolongam as transações e perturbam o planejamento. Quanto menor é a possibilidade de erros, menos interrupções ocorrem e melhor é o desempenho.

5.1 Proteção dos erros Definição: O critério Proteção dos erros se refere aos meios para detectar e prevenir os erros de entradas de dados ou comandos, ou possíveis ações com conseqüências desastrosas e/ou não recuperáveis.

Justificativas:

É preferível detectar os erros no momento da entrada do que no momento da validação. Isto pode evitar perturbações no planejamento da tarefa.

Exemplos de recomendações:

- Quando um usuário vai realizar o log-off e alguma transação não foi completada, ou se algum dado pode ser perdido, deve-se apresentar uma mensagem de advertência solicitando sua confirmação. - Os rótulos de campos devem ser protegidos.

- Campos projetados para apresentar informações devem ser protegidos: os usuários não devem ter permissão para modificar a informação contida nesses campos. - Assegure que a interface do software estará apropriada de acordo com todas as possibilidades de erro, incluindo entradas acidentais como de teclado.

5.2 Qualidade das mensagens de erro Definição: O critério Qualidade das mensagens refere-se à pertinência, à facilidade de leitura e à exatidão da informação dada ao usuário sobre a natureza do erro cometido (sintaxe, formato, etc.), e sobre as ações a serem executadas para corrigi-lo.

Justificativas:

A qualidade das mensagens favorece o aprendizado do sistema indicando ao usuário a razão ou a natureza do erro cometido, o que ele fez de errado, o que ele deveria ter feito e o que ele deve fazer.

Exemplos de recomendações:

- Se o usuário seleciona uma tecla de função inválida, nenhuma ação deve resultar, exceto uma mensagem indicando as funções apropriadas para aquela etapa da transação. - Forneça mensagens de erro com tarefas orientadas.

- Forneça mensagens de erro mais específicas possível. - Forneça mensagens de erro breves, porém informativas. - Adote um vocabulário neutro para as mensagens de erro, não personalize, não faça reprovações ao usuário e não utilize tom de humor.

5.3 Correção dos erros Definição: O critério Correção dos erros diz respeito aos meios colocados à disposição do usuário com o objetivo de permitir a correção de seus erros.

Justificativas:

Os erros são bem menos perturbadores quando eles são fáceis de corrigir.

Exemplos de recomendações:

- Permita a possibilidade de modificar os comandos no momento da sua saída. - Depois de cometer um erro, forneça ao usuário a possibilidade de corrigir somente a parte incorreta. - Se a transação foi completada e erros foram detectados, permita aos usuários fazer correções diretamente e imediatamente.

6. Coerência Definição: O critério Coerência se refere à forma na qual as escolhas na concepção da interface (códigos, denominações, formatos, procedimentos, etc.) são conservadas idênticas em contextos idênticos, e diferentes para contextos diferentes.

Justificativas:

Os procedimentos, rótulos, comandos, etc., são mais reconhecidos, localizados e utilizados, quando seu formato, localização, ou sintaxe são estáveis de uma tela para outra e de uma seção para outra. Nestas condições o sistema é mais previsível, a aprendizagem mais generalizável e o número de erros é reduzido. A falta de coerência pode aumentar o tempo de procura consideravelmente.

A falta de coerência é uma importante razão de recusa na utilização por parte dos usuários.

Exemplos de recomendações:

- Os títulos de janelas devem estar sempre localizados no mesmo lugar. - Utilize formatos de telas similares. - Utilize procedimentos similares para acessar o menu de opções. - Em ajudas, utilize as mesmas construções de frases.

- Prompts e comandos de entrada devem ser apresentados em localizações padroni zadas. - O formato de campos de entrada de dados deve sempre ser o mesmo.

7. Significado dos códigos e denominações Definição: O critério significado dos códigos e denominações diz respeito à adequação entre o objeto ou a informação apresentada ou solicitada, e sua referência. Códigos e denominações significativos possuem uma forte relação semântica com seu referente.

Justificativas:

Quando a codificação é significativa, a recordação e o reconhecimento são mais fáceis. Além disso, códigos e denominações não significativos para os usuários podem sugerir operações inadequadas para o contexto, conduzindo-os ao erro.

Exemplos de recomendações:

- Os títulos devem ser nítidos e significativos. - Apresente regras de abreviações explícitas. - Códigos devem ser significativos e familiares ao invés de arbitrários (por exemplo, M para masculino e F para feminino ao invés de 1 e 2).

8. Compatibilidade Definição: O critério compatibilidade refere-se ao acordo que possa existir entre as características do usuário (memória, percepção hábitos, competências, idade expectativas, etc.) e das tarefas de um lado, e a organização das saídas, das entradas e do diálogo de uma dada aplicação, de outro lado.

O critério compatibilidade também diz respeito à coerência entre os ambientes e entre as aplicações.

Justificativas:

A transferência de informações de um contexto a outro é mais rápida e eficiente quando o volume de informação que deve ser recodificado é limitado. A eficiência aumenta quando: os procedimentos necessários ao cumprimento da tarefa são compatíveis com as características psicológicas do usuário; os procedimentos e as tarefas são organizados respeitando as expectativas e práticas dos usuários; e quando as traduções, as interpretações, ou referências na documentação são minimizadas.

O desempenho é melhor quando a informação é apresentada de uma forma diretamente utilizável.

Exemplos de recomendações:

- A organização das informações apresentadas deve ser conforme a organização das entradas. - Os procedimentos de diálogo devem ser compatíveis com a ordem que o usuário imagina ou está habituado.

- Os formatos de calendários devem seguir o costume dos usuários (calendário europeu: dia/mês/ano e calendário americano mês/dia/ano). - Os termos empregados devem ser familiares aos usuários e relacionados à tarefa realizada. - As unidades de medida devem ser aquelas normalmente utilizadas. - Apresentações de dados textuais, mensagens ou instruções, devem seguir as convenções de textos impressos.

Recomendações de usabilidade de Dul & Weerdmeester

Recomendações que incluem os objetos de interfaces físicas.

Recomendações de usabilidade apresentadas por Dul & Weerdmeester

DUL, J. WEERDMEESTER, B. **Ergonomia Prática.** São Paulo: Edgar Blücher, 1991.

Apresentação: Diálogo Homem-Máquina Diálogo homem-máquina é definido como sendo uma comunicação de duas vias entre o usuário e o sistema, a fim de atingir um determinado objetivo. Nos últimos anos, os sistemas que permitem esse tipo de diálogo têm evoluído muito, aumentando a efetividade, eficiência e a satisfação do usuário.

1. O diálogo deve ser adequado à tarefa

O diálogo é considerado adequado à tarefa, quando permite que o usuário alcance o objetivo de forma efetiva e eficiente. Algumas características típicas desse princípio são: - o sistema deve apresentar, ao usuário, apenas os conceitos relacionados com as atividades do usuário no contexto da tarefa em execução; - qualquer atividade necessária ao sistema, mas não relacionada com a tarefa do usuário, deve ser executada só pelo sistema; - os formatos de entrada e de saída devem ser especificados de modo que se ajustem à tarefa.

2. Faça o diálogo autodescritivo

Um diálogo é autodescritivo quando o sistema fornece, a cada passo, o retorno *(feedback)* de informações ao usuário ou quando o mesmo pode pedir informações adicionais. Algumas características típicas desse princípio são: - após qualquer ação do usuário, o sistema passa a gerar informação de *feedback*. - as explicações ou *feedback* fornecidos pelo sistema ajudam o usuário a ter uma compreensão melhor do diálogo;

- se houver erros, o usuário deve ser imediatamente informado, se possível, dando alternativas para o prosseguimento.

3. Faça o diálogo controlável

O diálogo é considerado controlável quando o usuário tem possibilidade de direcionar o curso das interações até que o objetivo seja atingido. Algumas características desse tipo de sistema são: - a velocidade da operação não deve ser ditada pelo sistema

- deve haver possibilidade de desfazer a última etapa executada, com interações reversíveis; - a forma de apresentar dados de entrada e de saída deve estar sob controle do usuário, não o obrigando a executar operações desnecessárias, por exemplo, digitar 000123 no lugar de 123.

4. O diálogo deve atender as expectativas do usuário

Pode-se considerar que um diálogo atende às expectativas do usuário quando está de acordo com o seu nível de instrução, conhecimentos, experiências e as convenções normalmente aceitas. Esse tipo de diálogo caracteriza-se por: - os comportamentos exigidos no diálogo devem ser coerentes. Por exemplo, o término deve ocorrer sempre da mesma forma, seja com um *enter* ou *return* ou simplesmente nada;

- os termos usados devem ser familiares ao usuário, com o uso de uma só língua. Por exemplo, não se deve misturar termos em português com inglês, a não ser no caso de termos técnicos consagrados como ‘software’.

5. O diálogo deve ser tolerante a erros

Um diálogo é tolerante a erros quando, apesar dos erros evidentes de entrada, o processo pode ser mantido com apenas algumas ou nenhuma correção, até chegar ao resultado. Para isso, deve ter as seguintes características:

- os erros devem ser apresentados ao usuário, com orientações para que o mesmo possa corrigi-los.

- o sistema deve ter dispositivos para prevenir erros do usuário;

- as mensagens sobre erros devem ser apresentadas de forma objetiva e construtiva. Essas mensagens não devem ter nenhum julgamento de valor do tipo “esta entrada não tem sentido”.

6. O diálogo deve ser adaptável a indivíduos

Um diálogo é considerado adaptável aos indivíduos, quando o sistema admite mudanças para se

adaptar ao nível de conhecimento e às necessidades individuais.

As características principais desse tipo de diálogo são: - a quantidade de explicações necessárias pode ser ajustada para o nível de

conhecimento do usuário; - o usuário tem possibilidade de incluir seu próprio vocabulário para designar

objetos ou ações; - o usuário tem possibilidade de modificar a velocidade do processo, de acordo

com sua própria velocidade.

7. O diálogo deve ser adaptável à aprendizagem

Um diálogo é adaptável à aprendizagem quando fornece meios, orientações e estímulos ao usuário, durante a sua fase de aprendizagem. As principais características necessárias são:

- deve haver informações de help sempre que o usuário necessitar;

- o sistema deve ser organizado de modo a criar familiaridade, por exemplo, tendo padrão pra localização de mensagens e uma disposição constante dos elementos na tela.

Recomendações de usabilidade de Jordan

Tradução livre de Recomendações de usabilidade apresentadas por Jordan (1998)

JORDAN, P. W. An Introduction to Usability. Londres: Taylor & Francis Ltda., 1998.

Apresentação: Princípios para design com usabilidade O objetivo deste capítulo é delinear (traçar em linhas gerais) as características do design associadas à usabilidade. Dez princípios de usabilidade são discutidos abaixo com explicações de porque e como cada um dos princípios afeta a usabilidade.

1. Coerência

Projetar um produto com coerência significa que características similares devem ser realizadas da mesma maneira. Isto significa que, como um usuário ganha experiência com o produto, ele pode generalizar o conhecimento sobre o que aprendeu quando é realizada uma tarefa para ajudar a alcançar a outra tarefa. No contexto de um processador de texto, por exemplo, os passos envolvidos na tarefa para colocar o texto em formato negrito, poderão ser da seguinte forma:

1- Selecione o texto para ser editado 2- Abra o menu ‘Formatar’ 3- Selecione o comando ‘Negrito’ Similarmente, os passos envolvidos para colocar o texto em itálico devem ser da seguinte forma:

1- Selecione o texto para ser editado 2- Abra o menu ‘Formatar’ 3- Selecione o comando ‘Itálico’ Neste caso, o procedimento para formatar o texto em negrito seria consistente para formatar textos em itálico. Isto porque para ser formatado, ambos requerem que o texto seja selecionado e ambos requerem que o usuário selecione o menu ‘Formatar’. Estas são tarefas que os usuários provavelmente vão considerar como similar – talvez o usuário pense nelas como ‘tarefas formatadas’ – então é apropriado que o procedimento seja similar para ambos. Se o comando para colocar o texto em itálico fosse colocado num menu diferente, por exemplo, o menu ‘Fonte’ então estas tarefas estariam inconsistentes umas com as outras.

Incoerências são consideradas como um encaminhamento para os erros. No exemplo acima, se o comando ‘itálico’ não estivesse no menu ‘Formatar’, mas o comando ‘Negrito’ sim, então seria possível esperar que o usuário que tivesse aprendido como colocar textos em negrito, iria ao menu errado quando tentasse procurar o comando

itálico, por exemplo, ele selecionaria o menu ‘Formatar’ para procurar o comando de Itálico e ele não estaria lá. O leiaute de controles em carros é um bom exemplo de benefícios de coerência. Os pedais são sempre colocados com a embreagem à esquerda, o freio ao centro e o acelerador à direita. Este tipo de coerência significa que uma vez que alguém tenha aprendido a dirigir, ele pode transferir sua habilidade de um carro para outro. Se, no entanto, não houvesse coerência – isto é, se os pedais fossem organizados diferentemente de um carro para outro – motoristas teriam que fazer muito esforço em aprender como lidar com cada carro que eles encontrassem.

Coerência – Projetar um produto de maneira que as tarefas similares sejam feitas de maneiras similares.

2. Compatibilidade

Projetar para compatibilidade significa assegurar que a maneira que um produto funciona corresponde às expectativas do usuário, baseada no conhecimento que ele tem do mundo real. Assim como a coerência, a compatibilidade é importante porque as pessoas estão sujeitas a tentar generalizar de uma situação para outra, e desta maneira, um projeto que facilite a generalização possibilita que exista mais usabilidade do que um projeto que não facilite. O conceito de compatibilidade é similar ao de coerência, a diferença é que enquanto a coerência se refere a regularidades no design dentro de uma gama de produtos do mesmo tipo, a compatibilidade se refere às regularidades do design entre um produto e as fontes externas. Estas ‘fontes externas’ podem ser outros tipos de produto ou, certamente, alguma coisa do ‘mundo real’ na qual afeta a maneira que o usuário aproxima o uso de determinado produto. Considere, por exemplo, o comando ‘salvar’ num menu dirigido por planilha eletrônica. Imagine que o usuário de um programa semelhante nunca tenha usado uma planilha eletrônica antes, mas lhe são familiares outros menus como os de processadores de texto e de desenho. Com estas aplicações, o comando ‘Salvar’ está quase sempre colocado num menu de título ‘Arquivo’, ele provavelmente o encontrará imediatamente. Neste caso, então, o projeto do programas estaria compatível com as expectativas do usuário baseadas na experiência com outros tipos de programas. Se, no entanto, o comando fosse colocado em um menu diferente, seria incompatível com o que usuário espera e isso provavelmente causaria problemas. Outra questão que afeta a compatibilidade é o que é conhecido como ‘estereótipo da população’. Estas são preposições e associações na qual tendem a serem feitos por quase todos dentro de uma determinada cultura. Em muitas culturas, por exemplo, a cor vermelha é associada a perigo. Conseqüentemente quando projetando, por exemplo, um painel de segurança, os botões que o operador precisa pressionar em caso de emergência seriam de cor vermelha. Similarmente, o verde é frequentemente associado à permissão para prosseguir (no caso de semáforos de trânsito). Seria, então, sensato colorir os botões de verde se eles estiverem associados com a partida de um processo – por exemplo, um botão para partir uma parte de máquinas de produção. Os exemplos de cores citados acima tendem a ser bastante universal entre as culturas, no entanto, existem alguns estereótipos populacionais os quais tendem a ser mais específicos. Nos Estados Unidos e o continente europeu, por exemplo, um interruptor deve estar pressionado para cima para ligar algo, enquanto que no Reino Unido para ligar algo o interruptor deve ser pressionado para baixo. Onde existir este tipo de divisão, é importante que estas questões estejam envolvidas na criação do produto levando em conta os estereótipos associados com o mercado o qual for vender. De novo, a questão de segurança deve ser de suma importância se os usuários podem reverter seus instintos numa situação de emergência. No caso de interruptores, por exemplo, um usuário americano pode instintivamente tentar encontrar um interruptor para pressionar para baixo para fazer com que a máquina desligue. Compatibilidade – Projetar um produto de maneira que o método para operá-lo

seja compatível com a expectativa do usuário baseado no conhecimento de outros tipos de produtos e do mundo real.

3. Consideração sobre a habilidade do usuário

Ao interagir com um produto um usuário pode usar uma variedade de suas habilidades ou ‘canais’. Por exemplo, quando sintoniza um canal de TV, o usuário irá usar suas mãos para pressionar o botão do controle remoto, seus olhos verificam se a imagem é boa e lêem qualquer informação na tela, e

seus ouvidos verificam se o som está apropriadamente sintonizado.

É importante que quando se usa um produto, nenhuma das habilidades do usuário seja sobrecarregada – se isso acontecer, é provável que seja um problema de usabilidade. Este livro está sendo escrito em um programa de processador de texto. O uso do processador de texto é uma tarefa na qual precisa de uma alta demanda do canal visual, com olhar fixo se movendo para frente e para trás entre a tela e o teclado. Também é instalado no computador um programa de e-mail. De tempo em tempo as mensagens entram na caixa de e-mails e faz com que surja um pequeno ícone no topo da tela para indicar a chegada da mensagem. Enquanto se concentra visualmente no que está sendo digitado, este ícone é provavelmente muito pequeno para ser notado. No entanto, quando uma nova mensagem chega, ela é acompanhada de um bip. Este som indica que alguma coisa aconteceu e um breve olhar faz com que o ícone se torne visível. Este, então, é um simples exemplo de como o design pode empregar o canal de áudio quando o canal visual está altamente ocupado.

Assim, como outro exemplo simples, considere a diferença entre ouvir um rádio enquanto dirige e assistir TV enquanto dirige. Dirigir é uma tarefa que exige uma demanda visual. É óbvio que o motorista deve estar consciente da posição do carro na estrada bem como possíveis obstáculos, como outros automóveis e pedestres. Embora possa ser argumentado que ouvir rádio cause alguma distração, não existem questões de que carregue o canal visual do motorista. Assistir TV, no entanto, com certeza causaria uma ocupação adicional no canal visual e isto então poderia causar uma significante e perigosa distração na tarefa de dirigir.

Um produto tradicional na qual o princípio de consideração das habilidades do usuário tem sido aplicado é o piano. Pelo fato do usuário do piano utilizar as duas mãos para tocar a melodia, pedais são colocados para que um ou outro diminua ou acentue o som. Isto pode ser feito sem qualquer necessidade do pianista remover suas mãos das teclas do piano. Se as alavancas fossem operadas manualmente, então o pianista encontraria sérias dificuldades.

Consideração das habilidades usuário – Projetar um produto de maneira que se leve em conta a demanda das habilidades do usuário requeridas durante a interação.

4. Retorno das ações /feedback

É importante que as interfaces ofereçam reações claras sobre qualquer ação que o usuário tenha realizado. Isto inclui reação para reconhecer a ação que o usuário tenha que cumprir com o produto, e reação como conseqüência de qualquer ação. Um exemplo de problemas que podem ser associados com a falta de feedback vem de um estudo relatado por Jordan e Johnson (1991). Este foi um estudo de adaptabilidade/ adequação de um controle remoto como dispositivo de entrada para a operação de um som automotivo. Motoristas poderiam usar este dispositivo para aumentar o volume do som, escolher um faixa de um cd para tocar, ou alterar o balanço

do som dos alto-falantes. No caso de mudança de faixa de cd, existia um atraso de alguns segundos antes que a faixa selecionada começasse a tocar – isto ocorria simplesmente devido ao tempo necessário para que o laser no aparelho se movesse na posição correta no disco. Este atraso causou problemas para os usuários porque eles não ficavam imediatamente certos de que tinham realmente feito a ação de entrada

necessária para cumprir a tarefa. Isto podia levá-los a pressionar o botão correto novamente ou tentar pressionar outro botão na suposição de que a primeira ação que tinham realizado seria incorreta. Mais seriamente, isto freqüentemente levava os motoristas a tirar a atenção da estrada para checar se o botão que eles tinham pressionado era o correto.

Uma simples solução para este problema seria ter um feedback audível (como um bip) quando o botão fosse pressionado. Desta maneira os usuários saberiam que teriam feito a ação correta e poderiam voltar toda a sua atenção novamente para dirigir enquanto esperariam a música selecionada começar a tocar.

O exemplo acima relata o reconhecimento de feedback que uma ação deve fazer. Também é

importante fornecer feedback mostrando os resultados de uma ação que os usuários fazem. O uso de telefone fornece um simples exemplo. Depois de discar um número, o usuário vai ouvir alguns tipos de tons indicando o número discado – normalmente ou um tom indica que o número discado está chamando ou um tom indica que o telefone discado está em uso (ocupado). É importante que o feedback fornecido seja significativo. No caso dos tons do telefone, o feedback que o usuário recebe não é diretamente um espelho do que está acontecendo, mas são simplesmente sons os quais significam que o usuário precisa aprender por meio da experiência. Isto é provavelmente adequado para um produto simples como um telefone, mas para produtos mais complexos – como programas de computador – um feedback mais representativo pode ser útil. Esta é uma vantagem que pode ser oferecida por algumas interfaces gráficas. Por exemplo, considere novamente o caso da formatação de texto com um processador de texto. Em alguns programas a mudança no texto formatado pode ser representada na tela por uma mudança na cor daquele texto. Então, por exemplo, o texto que o usuário coloca em negrito pode aparecer em vermelho na tela, enquanto que o texto colocado em itálico pode aparecer em azul. Isto é, pelo menos, dado ao usuário feedback como resultado de uma ação tomada, no entanto, o significado do feedback depende que o usuário aprenda e se lembre que o texto em vermelho será impresso em negrito, e textos em azuis, em itálico. Em outros programas, no entanto, o formato que o texto é representado diretamente – então aquele texto em negrito, aparece imediatamente em negrito na tela e texto italizado aparece em itálico. É preferível que o usuário não precise aprender nem lembrar nenhum código de cor, mas possa ver o resultado das ações representadas diretamente na tela. Feedback/ Retorno das ações – Projetar um produto de maneira que as ações tomadas pelo usuário sejam reconhecidas e uma indicação significativa seja dada sobre os resultados dessas ações.

5. Prevenção de erro e recuperação

Parece inevitável que usuários cometam erros de tempo em tempo quando usam um produto. No entanto, os produtos podem ser projetados com a possibilidade de minimizar a ocorrência de erros e o usuário recuperar, de forma rápida e fácil, qualquer erro que tenha feito.

Um exemplo de projeto para recuperação rápida considera uma planilha eletrônica. Normalmente com esses programas o usuário irá digitar em linhas e colunas de números representando o valor de certas variáveis, e então ativará um comando para que se faça algum cálculo desses números. Imagine que enquanto digita os números nas linhas o usuário digite a letra ‘o’ ao invés do numero 0. Quando o usuário for fazer o cálculo este erro causará problemas já que programa não saberá como tratar com a letra que apareceu na coluna de dados. Presumidamente o programa tentará

retornar algum tipo de mensagem de erro, permitindo ao usuário encontrar o dado incorreto (o qual pode ser extremamente difícil, dado que a letra ‘o’ e o número 0 parecem similares), corrigido o erro voltaria ao trabalho de cálculo novamente. Uma solução melhor seria se o programa marcasse o erro tão logo ele ocorresse e alertasse

o usuário do problema. Então, quando o usuário fizesse o erro descrito, uma caixa de diálogo poderia aparecer imediatamente significando que uma entrada de dado não válida teria sido feita. O usuário poderia então corrigir o erro rapidamente antes de continuar a tarefa.

A facilidade do comando ‘desfazer’ disponíveis em muitos programas também é um bom exemplo de como o projeto pode fazer com que o erro possa ser desfeito rápido e facilmente. Estes também são benefícios que encorajam os usuários a ter atitude exploratória quando usam o programa. Além disso, se o usuário tenta usar um comando e algo inesperado ocorre, existe uma ‘segurança’ de saber que a ação pode ser desfeita rapidamente com o comando de ‘desfazer’. Um exemplo de como os erros podem ser prevenidos em primeiro lugar, considere a seqüência de operações que usuários têm que realizar quando utilizam um vídeo- cassete para gravar. O usuário tem que inserir uma série de parâmetros, incluindo a hora que o programa que ele irá gravar começa e termina, o canal de TV que irá passar, a data e o dia da semana que irá ser transmitido. O usuário então ativará o timer e então o VCR irá gravar. Se acontecer de o usuário esquecer de colocar qualquer uma dessas informações ou se esquecer de ativar o timer, uma de duas coisas pode ocorrer – o VCR pode não gravar nada ou pode gravar conforme os parâmetros de padrão do sistema que o usuário tenha adicionado. Desta maneira, se o usuário inseriu parâmetros para o VCR gravar de 17hs às 18hs na quarta-feira, mas esqueceu de inserir o canal, o VCR pode automaticamente gravar em determinado

canal, mas provavelmente este não será o canal que o usuário realmente desejava gravar. Muitos vídeos-cassete são projetados para evitar erros de omissão, solicitando o usuário estágio por estágio durante o processo da programação do VCR. Uma vez que o usuário tenha entrado no modo de programação, ele é solicitado primeiramente a inserir a hora na qual ele deseja que a gravação comece, depois então é solicitada a hora de término, depois o canal e assim por diante até que todas as informações sejam inseridas. É preferível um projeto na qual o usuário costume inserir os parâmetros separadamente, porque este tipo de projeto faz com que o usuário se lembre de todos os parâmetros na qual precisam ser colocados.

Prevenção de erro e recuperação – Projetar um produto de maneira que a probabilidade de erro deve seja minimizada e, então, se os erros realmente ocorrerem, que sejam recuperados de forma rápida e fácil.

6. Controle do usuário

Os produtos devem ser projetados de forma que ofereça o máximo de controle possível aos usuários sobre as interações que eles terão com o produto. Isto significa, por exemplo, oferecer controle sobre os passos e o tempo de interação. Uma crítica feita a interfaces sobre comando de fala é que elas tiram o controle de passos de interação do usuário. Por exemplo, Jordan (1992b), quando considerava o usuário de interfaces sobre comando de fala para sistemas de automóveis, notou que às vezes os motoristas podiam ser surpreendidos com informações quando eles não estavam esperando por elas. Imagine por exemplo que o motorista esteja encarregado de realizar manobras complexas como se locomover em fluxo de tráfico em uma rotatória, quando surpreendentemente a interface lhe fornece alguma informação – como a de que a pressão do óleo no motor está muito baixa. Provavelmente, ou o motorista perderá a informação porque estava concentrado na manobra, ou estará distraído com a manobra na possibilidade de oferecer risco à segurança. Um mostrador visual seria mais apropriado para informações urgentes dede que o motorista possa checá-las quando se sentir seguro para fazê-lo.

Outro tipo de interface na qual pode tirar controle do usuário são aquelas com a facilidade de time-out. Alguns VCRs, por exemplo, tem facilidade de time-out em seus sistemas. Isto significa que se o usuário não insere nenhum tipo de informação durante um período de tempo (talvez cerca de 30 segundos) então o VCR sairá do sistema. Um

estudo experimental que pesquisou a usabilidade de VCRs identificou que este era um problema de usabilidade (Jordan, 1992b). Freqüentemente, quando programando o VCR, os usuários paravam a tarefa para consultar o manual e quando voltavam seu olhar para o display, o VCR voltava para outro modo no sistema. A facilidade de Time- out pode causar determinados problemas para usuários novatos de um produto porque eles precisam de um tempo maior do que usuários experientes para mudar de um estágio da tarefa para o próximo. Talvez uma solução melhor, seria simplesmente incluir algum tipo de botão ‘home’, caso o usuário se sentisse perdido no sistema, ele poderia retornar num modo mais familiar dentro do sistema, de maneira rápida e fácil – com esta solução o usuário poderia ter a preferência e então estaria no controle.

No caso de programas de computador isto poderia ter implicações com algum padrão incluso no sistema. Os padrões podem fornecer benefícios para os usuários em termos de velocidade com o qual eles conseguem no uso do programa. De novo, considerando processadores de texto, imagine o tempo e esforço que seria envolvido em inserir todas as preferências de tamanho de texto para a largura de margens para cada palavra digitada! No entanto, é importante que os usuários estejam informados quais padrões têm sido usados e que esteja claro como podem alterá-los se desejar isto.

Projetos com ajustes é outro bom exemplo de como usuários podem ter controle. Quando se projeta uma cadeira, por exemplo, o designer deve tentar assegurar que as dimensões estejam apropriadas para os usuários destinados, mas ao mesmo tempo pode ser possível que o usuário ajuste facilmente as dimensões de forma que a altura do assento ao chão e o ângulo do encosto se acomodem às suas preferências particulares. Controle do usuário – Projetar um produto de maneira que o usuário tenha o máximo controle possível sobre as ações tomadas no produto.

7. Clareza visual

É importante que a informação seja apresentada de forma que ela possa ser lida rápida e facilmente sem causar qualquer confusão. Isto também inclui tanto os rótulos e informação quanto feedback. Os envolvidos no projeto do produto devem levar em conta questões como caracteres terem tamanho suficiente para serem lidos, qual quantidade de informação pode ser colocada em determinado espaço sem que se torne muito poluído, como cores podem ser efetivamente usadas na interface (enquanto ainda levem em conta que uma significante proporção da população sofra de daltonismo), e onde as informações podem ser colocadas.

Interfaces de tela para canais de TV são bons exemplos de produto onde essas questões são importantes. Devido à tela da TV ser usada, existe uma preponderância a usar muitas cores na interface. Isto pode ser beneficamente usado para distinguir modos – por exemplo, controles para alterar a imagem podem ser em azul. Cores podem também ser úteis onde o usuário precisar selecionar um comando de um menu on-screen. Como os usuários rolam através de listas o comando selecionado pode aparecer em vermelho. Aqueles envolvidos no projeto de interfaces on-screen devem também levar em conta a distância que o usuário estará da tela quando interagir. Normalmente estas interfaces podem ser operadas via controle remoto, então os usuários estarão provavelmente sentados numa cadeira a alguma distância. É claro, então, que é importante que os caracteres sejam grandes suficientes para serem lidos de determinada distância. Muitas TVs, especialmente as mais novas no mercado tem uma grande quantidade de diferentes funções. Os profissionais devem considerar quantas funções diferentes podem ser apresentadas de uma vez sem causar confusão. Quanto mais for apresentado de uma vez, menos será solicitado no menu estrutural – uma vantagem é que é menos provável que o usuário se perca no sistema. Por outro lado, apresentando muita informação de uma vez podem ser uma desvantagem porque os usuários terão que procurar entre muitas informações aquilo que desejam. A questão da posição da informação também é importante. Primeiramente, os profissionais devem decidir se a tela inteira deve ser usada como área de apresentação da informação ou se apenas uma parte dela será utilizada. É preferível usar a tela inteira para evitar poluição visual. Também é bom para usar caracteres maiores, melhorando a legibilidade. Por outro lado, quanto mais tela é preenchida com o display, mais imagem da TV é ocultada quando o usuário ajusta as posições. Uma outra questão focada na posição da informação é considerar se deve colocar ou não em opaco a imagem atrás dos menus. Isto provavelmente tornará o display mais fácil de ler com relação a ocultar mais imagem da TV. A alternativa é usar a imagem da TV como fundo dos menus, mas talvez isso cause poluição na tela dependendo do que se passar na TV naquele momento. Claro que a clareza visual não é importante somente para displays de telas. Também é importante, por exemplo, na questão de rotulação em interfaces baseadas em ‘botões e sintonizadores’. Então, é importante que rótulos para botões e teclas sejam claros. Com interfaces que contenham muitos sintonizadores – por exemplo, alguns painéis de controle – estes devem ser claramente distinguidos uns dos outros e devem ser espaçados de forma que eles estejam numa área visível, mas não tão perto que faça com que o display fique poluído. Clareza visual – Projetar um produto de maneira que a informação apresentada seja lida de forma rápida e fácil sem causar confusão.

8. Priorização da funcionalidade e da informação

Quando um produto tem uma vasta variedade de características, pode ser apropriado priorizar algumas características ao projetar a interface do produto. A priorização pode ter como base a freqüência de uso de determinadas características ou a comparação da importância de diferentes funções. Decidas as funções mais importantes, àquelas consideradas com maior prioridades podem, então, ser dadas maior lugar de destaque no projeto.

O projeto de interfaces gráficas para programas de computador é um bom exemplo de onde tais questões podem surgir. Freqüentemente essas aplicações contêm centenas de características as quais podem ser invocadas por meio de seleção de comandos em menus. Enquanto um grupo de comandos adequados e menus significativos trarão benefícios em termos de usabilidade, ainda existe perigo de que o número total de comandos aumentará o tempo necessário para procurar qualquer um deles. Uma solução comum e efetiva para este problema é incluir barras de ferramentas na interface. Uma barra de ferramentas contém ícones que representam certas características do produto. Usando estes ícones o usuário pode ativar certos comandos sem ter que usar os menus. Colocando as funções mais comumente utilizadas na barra de ferramentas podem

então poupar o usuário de gastar muito tempo e esforço procurando comandos mais freqüentes em meio a uma grande lista junto de outros comandos não tão usados.

Uma outra maneira de direcionar esta questão de menus em programas é o uso de estruturas hierárquicas. Quando o usuário abrir um menu, o comando que é mais freqüentemente usado pode ser imediatamente visível, enquanto que este menu hierárquico pode também incluir comandos de 'passagem' para submenus onde os comandos menos usados podem ser acessados.

O mesmo princípio é aplicado para apresentação de informação – somente algumas funções são mais comuns que outras, e também é verdade que existam algumas informações que querem ser mais vistas mais freqüentemente que outras. O uso de modos de displays padrões pode trazer estes benefícios.

Considere, por exemplo, a tela de display em um VCR (vídeo cassete recorder). Normalmente, existe apenas uma pequena área disponível para isso, no entanto, potencialmente, muita informação pode ser apresentada. Isto inclui, por exemplo, a hora atual, o canal que está sendo assistido, o modo que o vídeo está (se play, Record, stop, etc.), e a informação sobre as seqüências de gravação que foram programadas para o VCR. É claro que, apresentando todas essas informações no display ao mesmo tempo, causaria uma imensa poluição visual e seria extremamente difícil de ler. Para fazer com que esse problema seja evitado, a maior parte dos VCRs são projetados com um modo de display padrão – normalmente a hora atual – com outras informações acessíveis via botões. O modo de display padrão representa, com efeito, a priorização da informação na qual ela é colocada de tal modo que outras informações também possam ser apresentadas.

Priorização da funcionalidade e da informação – Projetar um produto de maneira que a funcionalidade e a informação mais importantes sejam facilmente acessadas pelo usuário.

9. Transferência adequada de tecnologia

Tecnologias que foram desenvolvidas para um propósito sendo aplicadas para outra área podem trazer grandes benefícios para os usuários. No entanto, se feitas sem cuidado suficiente podem também trazer problemas. Considere o histórico do controle remoto da TV. Este aparelho foi originalmente desenvolvido como ajuda a pessoas deficientes que tivessem dificuldades em se locomover até a TV para mudar o canal, alterar o volume, etc. O aparelho foi então adotado pela indústria como algo para ser usados por todos os usuários e hoje vem como acessório padrão em quase todas as TVs. De fato, hoje é comum que mais funções sejam acessadas via controle remoto do que no próprio painel na TV. Isto é um bom exemplo de como transferir a tecnologia desenvolvida de um determinado grupo de usuários para uma grande população lhes trazendo benefícios. Além disso, as pessoas preferem ficar sentadas confortavelmente em suas cadeiras a se levantando para mudar de canal ou alterar algum parâmetro.

Conseqüentemente, o controle remoto tem também sido implementado a outros produtos, desde aparelhos de som, VCRs e sistemas de iluminação – novamente trazendo benefícios em termos de conveniência para o usuário. Outra área onde se tem implementado controles remotos para entradas são os aparelhos de som automotivos (Jordan, 1992b). No entanto, os benefícios da aplicação neste contexto são duvidosos. Um controle remoto é conveniente para usar quando sentado em frente à TV, assim como todos os usuários têm que fazer, pegar o controle, apontá-lo para o aparelho e pressionar o botão apropriado. Estas simples ações, no entanto, podem se tornar muito mais difíceis quando dirigindo um carro ao mesmo tempo. Em primeiro lugar, localizar o controle remoto pode comprovar a dificuldade. Talvez o motorista tenha que colocá-lo no painel do carro ou na parte de baixo da alavanca de câmbio. Qualquer seja o caso, o motorista pode ter que olhar em volta até que o encontre e então, talvez depois de ter procurado para pegar o controle, terá que colocá-lo corretamente na mão para poder apontá-lo em direção ao rádio. É provável que essa perda de tempo seja mais preocupante, pois desviará a atenção do motorista da estrada. De fato, um estudo relatado por Jordan e Johnson (1991), indicou que o uso de controle remoto para operar um aparelho de som automotivo aumentou o nível de exigência do motorista comparado com a técnica convencional de pressionar os botões no próprio aparelho.

Outro exemplo de ambiente em automóveis na qual transfere a tecnologia, e que pode não trazer os

benefícios esperados, é o uso de head-up displays (HUDs). Estes são desenvolvidos originalmente para uso em aeronaves onde tem se comprovado uma interface bem sucedida. Neste caso, a informação é projetada na frente do pára-brisa da aeronave onde a informação pode ser lida sem que o piloto tenha que olhar para baixo no painel de controle. Isto funciona bem porque a cena fora da aeronave, da qual é formado o fundo da informação, é geralmente apresentada por um céu claro.

Se as HUDs fossem usadas em veículos, no entanto, a visão através do pára-brisa seria muito mais poluída – contendo, por exemplo, outros veículos, pedestres, a estrada, árvores, etc. Qualquer informação apresentada ma tela teria que ser lida contra este fundo, o que poderia ser bem difícil. De fato, a informação apresentada na tela poderia ser simplesmente dificultada por qualquer outra informação visual.

Transferência adequada de tecnologia – Projetar um produto de maneira que se faça uso adequado de tecnologias desenvolvidas para outros contextos para aumentar a usabilidade do produto.

10. Explicitação

Produtos devem ser projetados de forma que seja claro a forma como operá-los. Como simples exemplo, considere o projeto de portas em prédios públicos. Quando alguém se dirige a uma porta deve decidir se irá abri-la empurrando ou puxando. Se a porta é bem projetada estará claro qual a forma correta para abri-la. Uma chapa metálica nas portas indica que a porta deve ser empurrada, enquanto que uma barra que pode ser segurada, indica que puxar é a forma adequada. Este exemplo se refere ao que Norman (1988) diz sobre ‘Fornecimento’ (Affordance). Fornecimento são propriedades do projeto os quais fornecem fortes indícios de como o produto funciona – em outras palavras, eles fazem com que o método de operação seja explícito.

Com programas de computadores, a representação de comandos é um exemplo de onde a explicitação pode fazer com que o produto tenha mais usabilidade. Com menu dirigido a sistemas, os comandos que são representados explicitamente são aqueles que o nome do comando significa claramente a sua função. Por exemplo, o comando para enviar o texto num arquivo de texto ou figuras em programa de dados estatísticos para imprimir, normalmente é nomeado de ‘Imprimir’. Para a maioria dos usuários, a função deste comando é provavelmente clara com este nome. Se houver um caso que esta função for apenas ativada pela tecla de nome ‘F1’, então a representação do comando não estaria explícita, pois não existirá razão para que a tecla ‘F1’ represente a impressão.

Onde funções são representadas por ícones, o projeto destes ícones também afetará a explicitação pela qual as funções são representadas. Um estudo feito por Maissel (1990) classificou ícones e o quanto representativos eles eram – quanto mais representativo, mais os usuários associavam o ícone com sua função. Estudos realizados por Moyes e Jordan (Moyes e Jordan, 1993; Jordan e Moyes, 1994), indicaram que a representação tinha efeito acentuado na suposição tomada pelos usuários e algum efeito sobre os estágios de aprendizado de usabilidade. Em outras palavras, durante as suas primeiras interações com o produto, os usuários confiam nas representações dos ícones para identificar as funções que o ícone representa. (A propriedade representativa é menos salientada em termos de efeito no desempenho de usuários experientes, neste estágio, os usuários freqüentemente são capazes de lembrar quais ícones estão associados com quais funções, mesmo se os ícones não são representativos).

Explicitação – Projetar um produto de maneira que sejam dados indícios de como ele funciona e o método para operá-lo.

Recomendações de usabilidade de Nielsen

Tradução livre de Recomendações de usabilidade apresentadas por Nielsen (1994)

NIELSEN, J. Ten Usability Heuristics. Disponível em:

http://www.useit.com/papers/heuristic/heuristic_list.html. Acesso em: 10 set 2006.

Apresentação: Estes são os dez princípios gerais para o projeto de interfaces com usuários. São chamadas heurísticas porque estão mais na natureza da experiência do que guidelines específicas da usabilidade.

1. Visibilidade do status de sistema

O sistema deve sempre manter os usuários informados sobre o que está ocorrendo, com respostas apropriadas e dentro do tempo razoável.

2. Correspondência entre o sistema e o mundo real

O sistema deve falar a língua dos usuários, com as palavras, frases e conceitos familiares ao usuário, ao invés de termos orientados pelo sistema. Seguir convenções do mundo real faz com que a informação apareça em ordem natural e lógica.

3. Controle e liberdade do usuário

Os usuários freqüentemente escolhem funções do sistema pelo erro e precisam de uma saída fácil, ao invés de longas seqüências de ação, quando encontram um estado indesejado. Deve existir suporte de fazer (undo) e refazer (redo).

4. Coerência e padrões

Os usuários não devem ter que saber se palavras, situações, ou ações diferentes significam a mesma coisa. O sistema deve seguir as convenções da plataforma.

5. Prevenção de erro

Por melhor que seja a mensagem de erro, um cuidadoso projeto de interface é que impede a ocorrência dos problemas em primeiro lugar. Eliminar circunstâncias que sejam propícias aos erros, ou verificá-las e apresentar ao usuário uma opção de confirmação antes que incidam no erro.

6. Mais reconhecimento que recordação

Minimizar a carga da memória do usuário permitindo a visualização de objetos, ações, e opções. O usuário não deve ter que lembrar informações de uma parte do diálogo para outra. As instruções para o uso do sistema devem ser visíveis ou facilmente recuperáveis sempre que apropriado.

7. Flexibilidade e eficiência de uso

Aceleradores são despercebidos pelos usuários principiantes, mas freqüentemente aceleram a interação para o usuário mais experiente de tal forma que o sistema possa atender para ambos usuários. Permitir que os usuários customizem ações freqüentes.

8. Projeto estético e minimalista

Os diálogos não devem conter informações que sejam irrelevantes ou que sejam raramente necessárias. Cada unidade extra da informação em um diálogo compete com as unidades relevantes da informação e diminui sua visibilidade relativa.

9. Ajuda ao usuário, diagnóstico e recuperação dos erros

As mensagens de erro devem ser expressas de forma clara (sem códigos), indicar precisamente o problema, e sugerir construtivamente uma solução.

10. Ajuda e documentação

Pode ser necessário que o sistema forneça ajuda e documentação, apesar de ser melhor quando o sistema é usado sem documentação. A informação deve ser fácil de ser encontrada, focada nas tarefas do usuário. Devem ser listados passos concretos a serem seguidos, e não ser muito extenso.

Recomendações de usabilidade de Schneidermann

Tradução livre de Recomendações de usabilidade apresentadas por Schneidermann (2005)

SHNEIDERMAN, B. Designing the User Interface; Strategies for Effective Human- Computer Interaction. 4. ed. Addison Wesley. 2005.

Apresentação: Uso das oito “golden rules” (regras de ouro) da interface do design. Esta seção foca a atenção nos oito princípios chamados “golden rules” (regras de ouro), que são aplicáveis na maior parte dos sistemas interativos. Estes princípios, obtidos por experiência e refinados por mais de duas décadas, precisa de validação e ajustes para projetos específicos. Nenhuma lista como esta pode ser completa, mas ser for bem aceita pode ser um guia útil para estudantes e designers.

1. Esforço por coerência

Esta é a regra mais freqüentemente violada, mas seguí-la pode ser difícil, pois existem muitas formas de coerência. As seqüências de coerência de ação devem ser requeridas em situações similares, terminologias idênticas devem ser usadas em prompts, menus, e janelas de ajuda; e coerência de cores, layout, capitalização, fontes, etc devem ser empregadas por todas as partes. Exceções como confirmações solicitadas do comando de delete ou não repetição de senha devem ser compreensivas e limitadas em número.

2. Atendimento da usabilidade universal

Reconhecer as necessidades de diversos usuários e projetar com flexibilidade, facilitando transformação do conteúdo. Diferenças de principiantes a experientes, faixas etárias, incapacidades e diversidade tecnológica enriquecem a gama de requerimentos que guiam o projeto. Adicionando características para principiantes, como explicações, e características para experientes, como atalhos e faster pacing podem enriquecer a interface e melhorar a qualidade do sistema.

3. Oferecer feedback

Para qualquer ação do usuário, deve existir um sistema de feedback. Para ações freqüentes e menores, a resposta pode ser simples, enquanto que para ações menos freqüentes e maiores, a reposta deve ser mais completa. Apresentações visuais de objetos de interesse proporcionam um ambiente conveniente para mudanças explícitas.

4. Diálogos que indiquem o término da ação

Seqüências de ações devem ser organizadas em grupos com começo meio e fim. Informações de feedback ao término de um grupo de ações dão aos usuários satisfação de realização, sensação de alívio, o sinal para preparar para o próximo grupo de ações. Por exemplo, sites de comercio eletrônico deslocam seus usuários da seleção de produtos para a verificação ao final com a confirmação clara da página que a operação foi completada.

5. Prevenção de erros

Tanto quanto possível projete o sistema da forma com que o usuário não cometa sérios erros, por exemplo, desabilite (em cinza/não visível) itens de um menu que não estejam apropriados e não permita a entrada de dados alfanuméricos em campos numéricos. Se o usuário comete um erro, a interface deve detectar o erro e oferecer, de forma simples, maneiras construtivas e específicas para recuperar a ação. Por exemplo, o usuário não deve digitar novamente todos os dados de um

formulário se caso for inserido algum dado incorretamente, e sim deve ser guiado para corrigir somente o dado incorreto. Ações incorretas devem deixar o sistema inalterado ou então a interface deve oferecer instruções sobre como restaurar o status.

6. Fácil permissão para reverter ações

Tanto quanto possível, as ações devem ser reversíveis. Esta característica alivia o usuário de ansiedade desde que os usuários saibam que os erros podem ser desfeitos, isto encoraja a exploração de opções que não lhe são familiares. As unidades de reversão podem ser uma ação única, uma entrada de dado, ou um grupo completo de ações, tanto como a entrada de nome e endereço.

7. Suporte interno de controle

Usuários muito experientes querem ter a sensação de que estão no controle da interface e que a interface responda as suas ações. Ações inesperadas da interface, seqüências tediosas entrada de dados, falta de habilidade ou dificuldade para obter informações necessárias e a falta de habilidade para alcançar as ações desejadas, todas contribuem para ansiedade e insatisfação do usuário. Gaines (1981) obteve parte deste princípio com a regra “avoid acausality” e com seu empenho em tornar os usuários os elementos que iniciam a ação, mais do que elementos que respondam às ações.

8. Reduzir a carga de curta memória

A limitação do ser humano em processar memórias de curta duração (a regra de “thumb” é que os humanos podem lembrar, em média, sete pedaços de informação) exige que a apresentação seja simples, páginas múltiplas sejam estáveis, a freqüência do movimento de janelas seja reduzida, e o tempo de treinamento suficiente seja designado códigos, ou seja, mnemônicos (associação de idéias) e seqüências de ações. Onde apropriado, acesso online para formulários command-syntax, abreviações, códigos e outras informações devem ser fornecidos.

Observações: Os princípios descritos devem ser interpretados, refinados, e estendidos para cada ambiente. Eles têm suas limitações, mas fornecem um bom ponto de partida para celulares, desktop e webdesigners. Os princípios apresentados visam o aumento de produtividade dos usuários por fornecer procedimentos simples de entradas de dados, displays fáceis de compreender, retorno rápido das respostas, aumento do sentimento de capacidade, domínio e controle sobre o sistema.

Links:
[1] http://www.faac.unesp.br/posgraduacao/design/dissertacoes/pdf/mileni_kazedani.pdf