# Notas da Aula 01: Introdução às Interfaces com o Usuário

Prof. Daniel Caetano

## 1. Evolução das UIs

Os primeiros computadores desenvolvidos, há cerca de um século, eram equipamentos muito simples, mecânicos, que sequer eram programáveis. A interface de comunicação destes equipamentos, que tinham apenas algumas poucas funções, era feita através alavancas mecânicas e engrenagens.

Os equipamentos foram evoluindo lentamente e já em meados do século XX existiam computadores programáveis, mas sua interface de comunicação ainda era bastante precária, através de conexões elétricas físicas e cartões perfurados.

Durante a segunda metade do século XX os avanços foram incríveis. Surgiram teclados, impressoras e posteriormente monitores, e foram estes equipamentos que trouxeram as interfaces com o usuário para a era visual.

Inicialmente foram desenvolvidos os primeiros sistemas para impressora, que operavam basicamente com estrutura de terminal. O usuário digitava e o conteúdo aparecia na impressora. Com os monitores de vídeo, a interatividade aumentou, mas os primeiros software eram apenas adaptações do que já se utilizava nas impressoras para o vídeo. Estas interfaces estão disponíveis até hoje, em diversos sistemas, e são chamadas Interfaces de Linha de Comando (CLIs, Command Line Interfaces).

Durante algum tempo, os sistemas operavam na forma de texto puro, mas à medida em que os usuários se diversificavam (engenheiros, matemáticos, etc... e não apenas especialistas em computadores), começaram a surgir as primeiras interfaces de menus, embora ainda em modo texto. Seu uso ficou bastante popular no início da década de 80 e a grande maioria dos software desenvolvidos até 1985 usava este tipo de interface.

Entretanto, já no fim da década de 1970, uma idéia brilhante já havia surgido em um dos laboratórios da Xerox, o PARC (Palo Alto Research Center), onde havia sido criado o computador Star, que era operado através de um *mouse* (o primeiro da história) e a comunicação do usuário era feita através de imagens gráficas. Este equipamento nunca tornou-se viável devido ao seu alto custo.

Posteriormente foi desenvolvido o padrão X-Windows para computadores que executassem o sistema operacional compatível com UNIX. Este padrão tornou-se muito importante e tem muitas características especiais, mas só veio a ter participação no mercado de usuários não especialistas a partir do meio da década de 1990.

Entretanto, já no início da década de 1980, a Apple Computers, naquela época uma pequena empresa, começava a desenvolver um computador pessoal que fosse capaz de executar uma interface daquele tipo, a um preço acessível. O resultado disso foi um computador chamado Lisa, que pode ser confundido com uma "versão beta" do MacIntosh. O Lisa não fez tanto sucesso, mas a Apple tinha feito o mais importante: tinha criado o know-how, ou seja, ela sabia como fazer.

Pouco tempo depois surgia a primeira versão do MacIntosh, a um preço acessível (apesar de alto), que tornou-se o primeiro computador pessoal comercial a oferecer uma interface gráfica com o usuário (GUI). A interface era bastante pobre, mas já era de uso bem mais simples que as tradicionais interfaces de "linha de comando".

A IBM, a grande empresa ocidental dos computadores pessoais da época (no oriente a história é bem mais complexa e pouco conhecida) não quis ficar atrás e contratou com a Microsoft que fizesse uma nova versão do MS/IBM-DOS, com uma interface gráfica e usando os recursos dos mais recentes processadores da Intel (na época, o i80286). Neste mesmo instante a IBM iniciou pesquisas na área de interfaces com usuário, para determinar qual seria a interface perfeita.

Deste convênio surgiram dois software distintos: o Microsoft Windows e o MS/IBM-OS/2. O primeiro deles era apenas uma interface nova para o velho MS/IBM-DOS, já o segundo era um sistema operacional completamente novo, o qual já levava a interface gráfica em consideração em seu projeto. Na aparência, entretanto, os primeiros Windows e os primeiros OS/2 eram absolutamente similares, mas o OS/2 exigia bem mais recursos para executar, uma vez que fazia bem mais coisas que o Windows de então.

Por diversas razões mercadológicas, o Windows acabou se instituindo como padrão e a Microsoft resolveu romper seus contratos com a IBM, no início da década de 1990. A IBM não aceitou essa quebra de contrato pacificamente e, com os primeiros resultados de sua pesquisa sobre interfaces gráficas nas mãos (padrão CUA-1991, Common User Access), a IBM resolveu dar um passo ousado e contribuir significativamente para o avanço das interfaces gráficas.

O padrão CUA-1991 especificava diversas características de usabilidade para as interfaces, de forma a minimizar ao máximo a curva de aprendizagem dos usuários. Além disso, o padrão considerava uma implementação totalmente orientada a objetos, permitindo que os software desenvolvidos para a interface se integrassem a ela de forma tão sutil que o usuário nem precisaria ter consciência de quando utilizava uma ferramenta de sistema ou um software de terceiros.

Em 1992 a IBM lançava sua versão 2.0 do OS/2, contendo a WorkPlace Shell, implementando grande parte das características do CUA-1991. A Apple correu e implementou muitas das características também em seu MacOS 7, que na época sequer tinha o nome de MacOS.

Muitos dos recursos do CUA foram mimetizados pela Microsoft no Windows na versão 4.0 do mesmo (conhecido pelo nome de Windows 95). A partir de então os avanços foram muito disputados entre Apple, Microsoft, IBM e um novo jogador: Linus Torvalds que, com o seu Linux, trazia o mundo Unix para a realidade do usuário comum e, com ele, o X-Windows na versão gratuita chamada XFree.

Todas estas interfaces evoluíram e mudaram em muitos aspectos, culminando hoje com as disponíveis no MacOS X, Windows XP (em breve, Vista) e, no Linux, as diversas como KDE, Gnome, dentre outras. A WorkPlace Shell da IBM, apesar de ter sido a única a realmente implementar completamente orientação a objetos na interface, foi descontinuada em 2005, devido à mudanças na estratégia da empresa desde 1996, que fizeram com que o número de usuários se tornasse bastante reduzido.

Paralelamente ao desenvolvimento do que hoje é considerado o "estado da arte" das GUIs, começava a ser desenvolvido, no início de década de 1990, um novo paradigma, o das WUIs (Web User Interfaces - Interfaces Web com o Usuário). Mas esta história começou muito antes, e remonta o início do século XX.

Baseado no paradigma "hipertexto", pensado pela primeira vez no início do século XX por Paul Otlet e ligeiramente desenvolvido por H. G. Wells na década de 1930. Mas foi apenas na década de 1940 que o paradigma foi melhor formalizado (por Ted Nelson e Douglas Engelbart), sendo o termo "Hipertexto" criado apenas em 1965 e o primeiro editor hipertexte tendo sido criado em 1968, por Andries van Dam.

Os sistemas hipertexto foram bastante usados em diversas aplicações ao logo dos anos, mas seu uso tornou-se absolutamente mais expressivo quando Berners-Lee, um cientista do CERN (Organisation Européenne pour la Recherche Nucléaire - Organização Européia para a Pesquisa Nuclear), criou a World Wide Web.

Um dos primeiros software para "navegação" pelo hipertexto da World Wide Web foi o ViolaWWW, criado no início da década de 1990. Pouco depois, entre 1992 e 1993, foi desenvolvido o Mosaic, da NCSA (National Center for Supercomputing Applications - Centro Nacional para Aplicações de Supercomputadores), que rapidamente tomou conta do mercado.

O domínio do Mosaic durou até o lançamento do Netscape Navigator, em 1994, que tornou-se o padrão de navegador web em praticamente todas as plataformas existentes. A então empresa Netscape começou a imaginar novas aplicações para seu "suite", independentes de plataforma, que permitiriam o usuário editar e manusear documentos em qualquer equipamento conectado, independente de seu sistema operacional... algo que nunca chegou a acontecer.

Observando o plano pretensioso da Netscape, a Microsoft temeu por seu já consagrado domínio do mercado de sistemas operacionais caseiros. Rapidamente tratou de desenvolver seu próprio navegador Web, o qual foi incluído pela primeira vez no Windows 95 OSR2, ainda no fim de 1995.

De 1995 a 1998 a "guerra dos navegadores" foi ferrenha. O uso de táticas questionáveis de marketing por parte da Microsoft tiveram como resultado a extinção da Netscape Communications e o completo domínio do Internet Explorer a partir de 1999. Entretanto, a necessidade para um novo navegador para outros sistemas operacionais (O Internet Explorer só existia para Windows e MacOS), aliada à falta de segurança e problemas inerentes ao navegador da Microsoft, fizeram com que a comunidade OpenSource se movimentasse.

Aproveitando-se da liberação pública do código do antigo Netscape Communicator, formou-se a Mozilla Foundation, em 1998, para criar um novo navegador web gratuito, de código aberto e multiplataforma. Várias versões "Mozilla Internet Suite" foram lançadas de 1999 a 2003, até que em 2004 foi liberada a primeira versão oficial do novo navegador: Mozilla Firefox.

Desde o lançamento do Mozilla Firefox até os dias atuais, tem havido uma nova onda de migração do Internet Explorer 6 para Firefox 1.x. Estima-se que ainda em 2006 devam ser lançados os navegadores Internet Explorer 7 e Firefox 2, iniciando uma nova rodada na disputa pela liderança do mercado.

## 2. O que é uma Interface Gráfica com o Usuário (GUI)?

Uma interface gráfica com o usuário (GUI) é um meio pelo qual pode ocorrer comunicação entre um ser humano e um computador, utilizando-se para isso de uma metáfora de manipulação direta através de imagens e texto. Em outras palavras, as GUIs são instrumentos para que o usuário possa operar um computador através de metáforas gráficas.

Existem diversos tipos de GUIs, sendo o mais comum aquele que usa janelas, ícones, menus e um dispositivo apontador (WIMP: Windows, Icons, Menus e Pointing Device, tendo sido este um dos modelos pioneiros de interface, desenvolvido pela Xerox, e utilizado até hoje nos computadores.

Um dos grandes benefícios que as GUIs trouxeram para o ramo da computação foi uma grande redução na dificuldade de uso dos computadores. Isto é conseguido através do uso de metáforas de fácil reconhecimento pelo usuário para a execução de tarefas computacionais (por exemplo: apagar um documento pode ser feito arrastando o ícone do documento em um ícone de lixeira), o que permite que um usuário opere o computador com uma necessidade mínima de aprendizagem.

Além disso, as GUIs permitem uma grande flexibilidade na forma com que as tarefas são executadas, possibilitando um projeto de comunicação com o usuário que vise a um aumento de produtividade do mesmo, reduzindo seu cansaço ao máximo.

## 2.1. A Orientação a Eventos de uma GUI

Do ponto de vista do usuário, a principal característica das GUIs é, obviamente, que sua operação é gráfica. Entretanto, esta é apenas a diferença mais superficial, no que se chama de "*look*" (aparência). Existe uma diferença também referente ao "*feel*" (sentimento) que é fundamental e tem uma importância fundamental inclusive na etapa de programação.

Esta diferença se manifesta em "quem tem o controle" da operação. Nas antigas interfaces modo texto, quem definia "o que acontecerá a seguir" era sempre o computador; de uma certa forma, o usuário era refém do software, algumas vezes sem saber como sair de uma dada tela ou mesmo como fechar um programa.

As interfaces modo texto via de regra se baseiam na interação do tipo diálogo: o computador pergunta e o usuário responde. Assim que o usuário responder todas as perguntas, o computador faz algum processamento e posteriormente permite que o usuário escolha alguma outra opção.

As interfaces gráficas, por sua vez, operam no modo de "manipulação direta", onde o usuário pode selecionar qualquer elemento visível da tela e comandar uma ação sobre ele, **sem uma ordem pré-estabelecida**. Isto significa que, nas GUIs, em geral é o usuário quem determina "o que vai acontecer em seguida".

Esta característica é fundamental do ponto de vista da programação, dado que ela implica numa mudança completa no conceito base para o projeto da interface. Enquanto nas interfaces texto o código da interface é meramente seqüencial, com uma ordem bastante rígida, nas GUIs este código não tem uma ordem de execução prevista: dependendo da ação do usuário, um trecho totalmente diferente do código será executado.

Chamamos de "**evento**" cada ação do usuário e, se são estes eventos que controlam a execução de um software, este software é dito *orientado a eventos* e, portanto, requer uma programação orientada a eventos.

## 2.2. Como é a Programação Orientação a Eventos?

A programação orientada a eventos, no fundo, não difere muito da programação usual. A diferença reside na estrutura de controle do que será executado em cada instante.

Na programação seqüencial usual, o software costuma ter um *loop* principal que é executado até que o usuário selecione uma opção para sair. Todo o código do software fica dentro deste *loop* e é executado ininterruptamente.

Na programação orientada a eventos, o *loop* contém apenas uma chamada ao sistema, verificando se alguma *mensagem de evento* chegou para o programa. Caso nenhuma mensagem tenha chegado, o software fica bloqueado (sem executar) e o sistema (ou outros programas) continua sua operação. Caso alguma mensagem tenha chegado, o sistema retorna

a mensagem que, através de uma estrutura do tipo *switch* escolhe qual trecho de código vai ser executado naquele instante.

Uma má prática de programação é colocar *código útil*, ou seja, o código que realiza a tarefa para a qual o software foi projetado, dentro deste *switch* ou mesmo em outros lugares. Isto não deve ser feito pois, fazendo isso, o programa exigirá a interface gráfica para poder realizar suas tarefas. O ideal é que dentro destas regiões sejam apenas chamadas operações de um outro componente do sistema, este sim executando as atividades do programa. Neste tipo de arquitetura, a MVC, a interface gráfica ocupa a camada *visão* (*view*) e as atividades do sistema ocupam a camada *controle* (*controller*).

## 3. O que são as Web User Interfaces (WUIs)?

Atualmente é bastante comum um tipo diferente de interface de comunicação com o usuário: as WUIs (Web User Interface), que são similares às GUIs, mas são baseadas em pressupostos distintos.

As WUIs são originadas de um paradigma diferente de comunicação, chamado *hipertexto*, que é a estrutura de *links* que relacionam um documento a outro. O objetivo do hipertexto é permitir que um texto seja fluente e, ao mesmo tempo, possibilite que o leitor se aprofunde em tópicos específicos que forem de seu interesse, através dos *links*.

Apesar de ser possível utilizar uma WUI para exercer o papel de uma GUI (e vice-versa), é necessário cuidado com este tipo de inversão. Tal atenção especial deve existir devido ao fato que, enquanto os elementos das GUIs são orientados às atividades a serem realizadas, os elementos das WUIs são orientados às informações a serem apresentadas.

Sendo assim, o projeto de cada um destes tipos de interface deve seguir conceitos bastante distintos, sendo inadequado projetar uma WUI seguindo conceitos de GUI e vice-versa.

## 4.Bibliografia

DOMINGUES, D. G. *O uso de metáforas na computação*. Dissertação - Escola de Comunicações e Artes da Universidade de São Paulo. São Paulo, 2001.

NIELSEN, J. *Projetando Websites*. Editora Campus, 2000.

DIX, A; FINLAY, J; ABOWD, G; BEALE, R. *Human-Computer Interaction***.** Edinburg, England: Prentice-Hall, 1997.

SHNEIDERMAN, B. *Designing the User Interface.* 3rd. Ed. Addison-Wesley. Reading, Ma. 1998.

DEITEL, H.M; DEITEL, P.J. *Java: como programar* - Sexta edição. São Paulo: Pearson-Prentice Hall, 2005.

WIKIPEDIA. http://www.wikipedia.org/

Notas da Aula 02: Características Gerais de Interfaces com o Usuário
Prof. Daniel Caetano

## 1. O que é uma Interface com o Usuário?

De uma forma intuitiva, uma interface com o usuário é o mecanismo pelo qual ocorre a comunicação do usuário com algum equipamento. Certamente os computadores não fogem a essa regra e também possuem uma interface com o usuário.

Mas **o que é uma interface**? Segundo Brenda Laurel, "Uma interface é uma superfície de contato que reflete as propriedades físicas das partes que interagem, as funções a serem executadas e o balanço entre poder e controle". No caso dos computadores, a interface com o usuário é uma camada responsável por intermediar a "conversa" entre o computador e o operador do mesmo, de forma que o computador possa entender as instruções do operador e o operador possa compreender as informações fornecidas pelo equipamento.

Por esta razão, pode-se dizer que a **interface deve refletir**:

- **As características de ambos**, computador e usuário;
- **As funcionalidades que o computador oferece** ao usuário;
- **"Quem controla" a operação** (usuário ou computador)

Por outro lado, a **interface deve esconder** do usuário **detalhes internos que pouco interessam** ao usuário (se um arquivo está na máquina local ou em outro lugar da rede etc.). Adicionalmente, deve **permitir que o usuário identifique** rapidamente qual **a aplicação sendo utilizada**, ao mesmo tempo em que deve **padronizar o mecanismo de interação** com diferentes aplicativos.

Assim, a interface com o usuário visa **facilitar e padronizar a interação** entre humano e computador. A interação pode ser definida como a troca de informações entre humano e computador, sendo que a interface pode ser decomposta em **duas** interfaces distintas:

- **Interface de Ação**: usada pelo ser humano para comandar o computador.
- **Interface de Percepção**: usada pelo computador para informar o usuário.

## 2. Quem é o Usuário?

Quando se fala em projeto e desenvolvimento de interfaces, a primeira pergunta que deve ser feita é: "Quem é o usuário?". A resposta de tal pergunta tem influência direta tanto no projeto da interface de ação quanto na interface de percepção.

É importante ressaltar que, em geral, **não há um único usuário**. Deve-se considerar sempre os usuários de diversas categorias, como por exemplo:

**- Iniciantes**
**- Intermediários**
**- Experientes**

Além disso, há de se considerar se o **usuário é especialista** em uma determinada área envolvida pelo software ou não. Um software médico, por exemplo, pode ter características de comunicação distintas, se for usado tanto por médicos quanto por enfermeiras e leigos.

Da mesma forma, mesmo **usuários de um mesmo nível** de conhecimento do software e **de uma mesma área** podem ter **preferências distintas**, sendo interessante, na medida do possível, tornar a **interface** com o usuário **configurável** de acordo com o gosto do usuário. Obviamente a configuração padrão deve ser boa para a maioria dos usuários, uma vez que é inadequado exigir que a maioria dos usuários do sistema perca tempo alterando características de apresentação, por exemplo.

Assim, é necessário levar em conta os tipos de usuários distintos no projeto de cada um dos tipos de interface (ação e percepção).

## 3. Conhecimentos Envolvidos no Projeto das Interfaces

O projeto de interfaces com o usuários é, em geral, composto de equipes mistas, com profissionais de diversas áreas como psicologia, design, programação etc. Isto se dá devido ao vasto conhecimento necessário para que um projeto atenda a todas as necessidades de forma adequada.

Algumas das áreas envolvidas são a **Psicologia Cognitiva** (que estuda os processos de aprendizado, resolução de problemas, memória e atenção), **Psicologia Perceptiva** (que estuda a forma como o mundo é percebido pelo ser humano), **Psicologia Social** (que estudas as interações entre indivíduo, grupo e atividades), **Psicologia Organizacional** (que estuda a influencia das práticas de trabalho no indivíduo e o funcionamento das organizações), **Ergonomia** (que estuda as formas mais práticas, cômodas e satisfatórias de se realizar uma tarefa) além de diversas outras áreas como **Lingüística**, **Inteligência Artificial**, **Filosofia**, **Sociologia**, **Antropologia**, **Engenharia** etc.

## 4. Características Desejáveis da Interação

A interação humano-computador deve ser projetada tendo em vista principalmente as seguintes características:

- **Ser natural**, intuitiva;
- **Vocabulário e regras simples**, com um mínimo de regras;

- **Conceitos fáceis de assimilar** ou já conhecidos pelo usuário;
- **Personalizável** pelo usuário;
- **Capacidade de recuperação de erros** e enganos (undo!);
- **Ajuda ao usuário**.

Entretanto, **não existem regras claras** que garantam tais características a um projeto, o que torna a criação de uma adequada interface com o usuário um trabalho árduo e propenso a equívocos. **Para minimizar** tais **equívocos**, deve-se **seguir os padrões** de uso da interface (guidelines fornecidos pelo fabricante, como por exemplo a ordem dos menus *Arquivo, Editar, ..., Ajuda*). Deve-se sempre tentar fazer com que o comportamento da interface seja o mais padronizado possível, com **tempos de resposta** tão **curtos** quanto possível.

Existem **ferramentas** (Borland Delphi, Microsoft Visual Basic etc.) que **auxiliam no projeto** de interfaces, **facilitando a manutenção da consistência**, **aumentam a produtividade**, reduzindo o tempo de desenvolvimento em até 50%, **possibilitam participação de não programadores**, **facilitam a prototipação**, além de **lidar com outras dificuldades inerentes** a este tipo de projeto. Entretanto, o uso te tais ferramentas **não garante boas interfaces**. O sucesso repousa na mão da equipe de projeto e desenvolvimento.

## 5. Usabilidade e Desenvolvimento Centrado no Usuário

As boas características de uso de uma interface são denominadas **Atributos de Usabilidade** e, em geral, o "grau de usabilidade" pode ser medido pela facilidade de uso encontrada pelo usuário, ou seja, quão bem o usuário interage com o software através da interface. Este é um **conceito importante** em todas as interfaces, **em especial nas WUIs**, pois o **usuário Web não é um usuário cativo**.

Em geral, a usabilidade é medida através dos seguintes fatores:

- **Facilidade de aprendizado**
- **Eficiência**
- **Facilidade de relembrar**
- **Tratamento dos erros**
- **Satisfação do usuário**

Como é possível observar, são fatores intimamente ligados às características desejáveis apresentadas na seção anterior. Isso, obviamente, não é uma coincidência. Este fato vem da diretriz de projeto de interfaces que é o "**desenvolvimento centrado no usuário**", focando em suas habilidades e deficiências.

É importante frisar que é necessário se preocupar também com usuários deficientes, como daltônicos, cegos etc., na medida do possível. Não faz sentido tornar um software inútil a alguma destas categorias (que no total representam quase 10% da população) por mera displicência.

Estes critérios, que proporcionam o conforto do usuário, visam **aumentar a aceitabilidade** de um produto no mercado. Um **bom produto com uma péssima interface** (desconfortável, feia, difícil de usar, etc.) tem enormes chances de ser um **fracasso**, ainda que seja "o que há de melhor" na execução de sua tarefa (dado que o usuário conseguiu superar as dificuldades de interação).

A **usabilidade vem tendo uma importância cada vez maior** devido ao fato que, cada vez mais, temos mais e mais capacidade de processamento disponível para a interface, com o barateamento dos recursos computacionais. Se nos computadores antigos os usuários toleravam uma interface lenta e feia (devido às diversas limitações dos equipamentos), hoje esta tolerância não existe mais.

Além disso, em algumas **atividades críticas** (controle de aviação, usinas nucleares etc.) um problema de usabilidade de interface pode fazer com que grandes catástrofes ocorram. Assim, o desenvolvimento de tais sistemas requerem extrema atenção por parte dos projetistas de interface.

Resumidamente, os princípios do Desenvolvimento de Interfaces Centrado no Usuário são:

- Enfoca preferências, capacidades e limitações do usuário;
- Considera a existência de vários tipos de usuário;
- O usuário é quem controla o programa;
- O resultado das ações do usuário é previsível;
- Economia de ações do usuário.

## 6.Bibliografia

NETTO, A.A.O. Modelagem e Gerência de Interfaces com o Usuário. Ed. Visual Books, 2004.

FOLEY, J. D. et al. *Computer Graphics Principles and Practices*. Addison-Wesley, Reading, 1990.

Bibliografia Complementar:

DIX, A. et al. *Human-Computer Interaction***.** Edinburg, England: Prentice-Hall, 1997.

MARCUS, A; VAN DAM, A. *User-Interface Developments for the nineties.* Computer, Sept., 1991. p.49-57

SHNEIDERMAN, B. *Designing the User Interface.* 3rd. Ed. Addison-Wesley. Reading, Ma. 1998.

Notas da Aula 03: Estilos de Interfaces Gráficas
Prof. Daniel Caetano

## 1. Introdução

Uma vez que o **papel** fundamental **da interface** com o usuário é **promover a interação entre ser humano e computador**, os **estilos** de interface são **definidos** exatamente **pela** maneira com que essa **interação** ocorre.

Como vimos anteriormente, a interação é composta pela execução de comandos por parte do usuário (**interface de ação**) e a apresentação de resultados pelo computador (**interface de percepção**). Dependendo do formato como cada uma destas interfaces é implementada, a interface é classificada dentro de uma dada categoria.

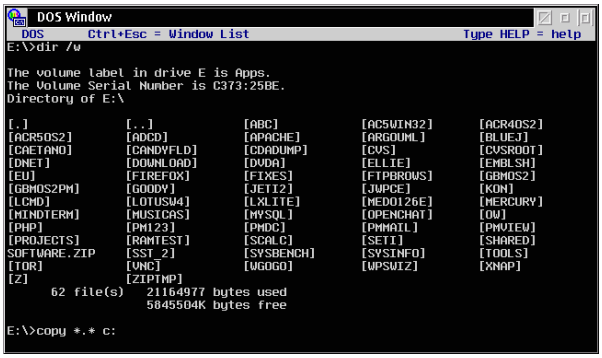
Algumas **classificações** mais comuns são **linha de comando/linguagem natural**, baseadas em **menus**, **diálogos**, **manipulação direta/icônicas** e **WYSIWYG** (What You See Is What You Get, ou "Você Vê Aquilo Que Será o Resultado"). Cada uma delas será descrita com maiores detalhes a seguir, onde será possível perceber que a maioria das interfaces gráficas atuais (como a fornecida pelo Windows XP e MacOS X, bem como Gnome e KDE, dentre muitas outras) são **compostas** por elementos de cada uma destas formas de interação.

## 2. Interfaces de Linhas de Comando/Linguagem Natural

As linguagens de linha de comando foram as **primeiras que surgiram como interface de interação complexas entre seres humanos e computadores**. As interfaces de linha de comando são caracterizadas por **solicitar** alguma **ação** do usuário (descrita na forma de **comando ou linguagem natural**), **fornecer a resposta** e logo em seguida **solicitar outra ação** e assim por diante.

Este tipo de interface originou-se com os equipamentos de teclado/impressora conjugados e, posteriormente, foi adaptado para equipamentos de teclado/monitor de vídeo conjugados. A grande **vantagem** deste tipo de interface são sua **grande expansibilidade e poder**, mas o **tempo de aprendizado é alto** e é considerado uma **grande desvantagem**.

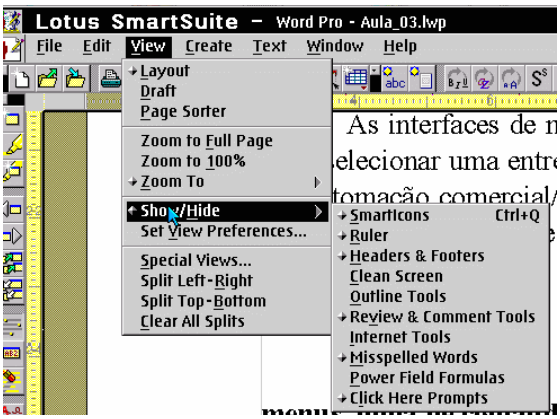
Este tipo de interface é **usada até hoje** na maioria dos sistemas operacionais (Prompt de Comando do Windows XP e Terminal do Linux, por exemplo) e até mesmo em alguns programas profissionais, como o AutoDesk AutoCAD. Uma interface de linha de comando é como a a apresentada a seguir.




## 3. Interfaces de Menus

As interfaces de menus **são aquelas que apresentam** uma série de **opções prontas** e o usuário deve selecionar uma delas. Este tipo de interface começou a ser usado em **automação comercial/empresarial**, associada à interface de diálogos, ainda no **período de domínio das interfaces de texto** (início da década de 1980).

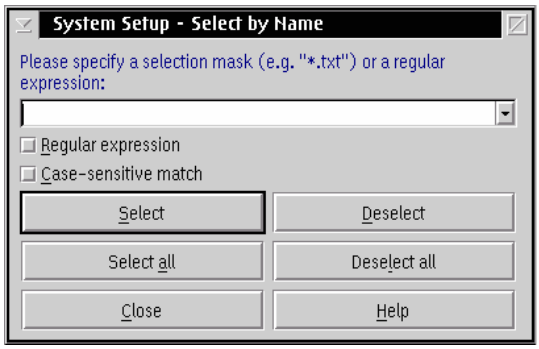
As interfaces de menus são **hoje** muito utilizadas como **complemento às interfaces gráficas** como usuário de forma geral. Sua **facilidade de uso e expansibilidade** são suas grandes **vantagens**. Os menus do tipo **pull-down** possuem ainda a vantagem de possibilitar o **agrupamento de funções em categorias**, facilitando o aprendizado do usuário sem abarrotar a tela com figuras e texto. A seguir, um exemplo de menu pull-down.




## 4. Interfaces por Diálogos/Form Fill

As interfaces por **diálogos são bastante antigas**. Em verdade, a maioria dos **primeiros software interativos** eram baseados nesta técnica. São interfaces em que **uma seqüência de perguntas é apresentada** ao usuário e ele precisa respondê-las para completar alguma ação.

Tais interfaces evoluíram bastante, nas interfaces hoje temos **diálogos normais** e **modais**, com auxílio de contexto, etc. Um exemplo de diálogo pode ser visto abaixo:




As grandes **vantagens** deste tipo de interface é que o **aprendizado é quase instantâneo**, quando ela é bem planejada. A grande **desvantagem** é sua **rigidez e dificuldade de expandi-la** dinamicamente.

## 5. Interfaces de Manipulação Direta/Icônicas

As interfaces de manipulação direta **são aquelas em que todos os elementos sobre os quais se pode realizar operações são apresentados na tela, e as ações sobre estas representações causam ações efetivas no sistema**, como por exemplo "Arrastar a folha de documento para a lixeira", ação esta que apaga um arquivo.

Apesar de tentativas anteriores de implementação, a primeira interface comercialmente aceita a implementar manipulação direta foi a **interface gráfica do System 1.0, do primeiro MacIntosh**. Posteriormente ela foi adotada com sucesso em sistemas como o **Windows, KDE, Gnome, OS/2, BeOS** e outros.

Normalmente as interfaces de manipulação direta usam a **representação icônica** dos elementos passíveis de sofrer uma ação. Em geral esta representação icônica contêm **um texto e uma figura**. Ambos **devem ser aprendidos** o usuário e, tanto quanto possível, facilitar a identificação/reconhecimento do que ele representa, por parte do usuário.


Mouse


Firefox

Este tipo de interface é bastante interessante pois, dado que os **programas seguem algum padrão** (em geral proposto pelo fabricante do sistema operacional, como a Microsoft ou a Apple), o **aprendizado é rápido** e seu **uso é simplificado por parte de usuários leigos**.

Entretanto, **usuários experientes** costumam achar essas **interfaces lentas e desajeitadas** (usuários que, em geral, usam as **"hotkeys"**), além de **alguns tipos de operações serem pouco intuitivas** (como diferenciar entre **mover e copiar** um arquivo de um lugar para outro?), e **dificultar** a seleção e execução de **ações que envolvam grande quantidade de arquivos** (selecionar um por um é mais complexo que digitar "**copy *.com c:**").

## 6. Interfaces What You See is What You Get (WYSIWYG)

As interfaces do tipo WYSIWYG não são exatamente recentes: as primeiras foram desenvolvidas por **época** da implementação **das primeiras interfaces gráficas** (um exemplo notório foi o **Aldus Page Maker** para o Windows e o System da Apple).

A característica destas interfaces é que elas **tentam reproduzir na tela exatamente aquilo que o usuário obterá ao imprimir o resultado** e, por esta razão, são **normalmente voltadas a edição de documentos que visam ser impressos**.

Programas comuns do dia-a-dia como Microsoft Word (e, em algum nível, Microsoft PowerPoint e Macromedia DreamWeaver) usam esta tecnologia, perceptível até mesmo em pequenos detalhes como a impressão de acentos e estilos de letra no próprio texto, assim como a apresentação de textos. Abaixo está a aparência do mesmo texto digitado no Microsoft Word e no LaTeX:

Word: Um texto com **negrito**, *itálico* e <u>sublinhado</u>!
LaTeX: Um texto com \textbf{negrito}, \textit{itálico} e \underline{sublinhado}!

Ambos são impressos extamente da mesma forma:

Um texto com **negrito**, *itálico* e <u>sublinhado</u>!

As interfaces do tipo **WYSIWYG** têm uma grande **vantagem** que é permitir que o **usuário veja exatamente como está ficando o resultado de suas ações**. Entretanto, este tipo de interface **não é viável em qualquer tipo de aplicação** (em especial coisas que não serão impressas), além de ser um tanto restrito e, em geral, os formatos de documentos de software que implementam este tipo de interface são proprietários, limitando a portabilidade.

Programas que usam interface do tipo **não-WYSIWYG** (geralmente "edição de texto puro", como o NotePad do Windows) em geral têm conteúdo **mais portável**. Alguns formatos como TeX (do LaTeX) e Rich Text Format (RTF) permitem a representação de caracteres especiais (como a indicada anteriormente), fórmulas, figuras e outros de forma totalmente portável (e editável sem o editor original).

Existem editores WYSIWYG para documentos LaTeX e a maioria dos editores WYSIWYG grava documentos no formato RTF, com algumas restrições (o mesmo valendo

para HTML). Ainda assim, os arquivos gerados por estes editores são plenamente editáveis manualmente, usando um editor de texto puro normal (como o NotePad).

## 7. Interfaces Mistas (WIMP)

**Idealizada pela Xerox na década de 1970**, no PARC, a interface do tipo WIMP **(Windows, Icons, Menus and Pointers)** mistura todos os elementos usados até então (**diálogos e menus, agora colocados em janelas**) com novos elementos **icônicos e ponteiros** (mouse). Este tipo de interface foi a base para a criação dos sistemas do tipo **manipulação direta**, e posteriormente passaram a incorporar também o tipo de interface **WYSIWYG**.

Estas interfaces costumam usar **metáforas com elementos do mundo real**, sendo que o usuário trabalha no "**desktop**" (mesa), manipula **documentos**, **pastas**, **impressoras**, **latas de lixo**, **trituradores**, etc.

Este tipo de interface ainda não parou de evoluir, embora venha ganhando recursos em um ritmo bem inferior atualmente. Sua grande **vantagem** é combinar os **aspectos positivos de todas os tipos de interface em um único**. A **desvantagem** é que a **programação** de software para plataformas WIMP é bastante **mais complexa**, principalmente quando se opta por seguir os padrões propostos pelo fabricante (Microsoft, Apple, IBM...).

## 8. Tabela Comparativa

A tabela abaixo (Foley, 1990) mostra as vantagens e desvantagens de cada tipo de interface de forma sumarizada. L significa "Low" (baixo), M significa "Medium" (médio) e H significa "High" (alto).

| Estilo | Tempo de aprendizado | Velocidade de uso | Propagação de erros | Extensividade | Habilidade no teclado |
|---|---|---|---|---|---|
| WYSIWYG | L | | L | L | |
| Manipulação Direta | L | M | L | L | |
| Menus | M | M | L | M | |
| Form Fill-in | L | H | L | M | H |
| Comandos | H | H | H | H | H |
| Linguagem Natural | L | M | H | H | H* |
| Pergunta-e-resposta | L | L | L | H | H |

* - teclado; nada se usar voz

## 9.Bibliografia

FOLEY, J. D. et al. *Computer Graphics Principles and Practices*. Addison-Wesley, Reading, 1990.

DOMINGUES, D. G. *O uso de metáforas na computação*. Dissertação - Escola de Comunicações e Artes da Universidade de São Paulo. São Paulo, 2001.

Bibliografia Complementar:

DIX, A. et al. *Human-Computer Interaction***.** Edinburg, England: Prentice-Hall, 1997.

PREECE, J, et al. *Human-Computer Interaction.* Addison-Wesley, 1994.

Notas da Aula 04: Critérios de Projeto de IHC
Prof. Daniel Caetano

## 1. Introdução

Como foi visto em aulas anteriores, o projeto da Interação Humano-Computador é feita com base no modelo "voltado ao usuário" o que, em outras palavras, significa que o foco do desenvolvimento são as necessidades do usuário.

Os passos para o desenvolvimento voltado ao usuário são:

a) Compreender e assimilar os conceitos da aplicação;
b) definir a linguagem de interface;
c) considerar fatores ergonômicos;
d) escolher as tarefas interativas;
e) selecionar dispositivos de interação.

Cada um destes passos será examinado detalhadamente a seguir.

## 2. Compreender e Assimilar os Conceitos da Aplicação

Antes de mais nada, o que significa **compreender** os conceitos da aplicação? Neste caso, **significa gerar uma especificação de requisitos**. Mas em quê deve ser baseada esta especificação?

* O primeiro aspecto é determinar **quem é o usuário**, pois os requisitos variam de acordo com o tipo de usuário considerado. É importante estudar e observar o usuário, interagir com o mesmo, entendendo como ele pensa e porque age de uma determinada forma.

* O segundo aspecto é definir o **vocabulário** a ser usado. Em geral o vocabulário deve ser o **da aplicação**, **evitando** usar **termos** técnicos **da área computacional**.

* Deve ser definido também o que deve ser feito (e como fazer) no caso da ocorrência de **erros**.

* Finalmente, deve ser feita a **especificação da interface**, levando em conta os requisitos desta interface como as **capacidades do software** a serem representadas, **como atender cada categoria de usuários** etc.

## 3. Definir a Linguagem de Interface

Depois da definição dos requisitos, deve ser feita a definição da linguagem da interface. Em outras palavras, deve-se escolher **como será a comunicação** do usuário com o computador (**interface de ação**) e como será a comunicação do computador com o usuário (**interface de percepção**). Para definir a interface, há dois modelos mais comuns: o **modelo lingüístico** e o **modelo orientado à implementação**.

O **modelo lingüístico** é baseados em descrições lógicas das interações, **usando regras matemáticas** para expressar tais interações. Por exemplo:

**< Comando do Usuário > ::= < Nome do Comando > < Objeto >**
**< Nome do Comando > ::= MOVER | COPIAR | DELETAR**
**< Objeto > ::= < Nome de Arquivo >**

O **modelo orientado à implementação** é aquele em que a **interface é desenhada diretamente**, sem uma preocupação em explicitar a linguagem, como é feito no desenvolvimento de aplicações **Visual Basic, Delphi** etc.

## 4. Considerar Fatores Ergonômicos

A principal **medida de qualidade** de uma interface pode ser feita pela avaliação do **esforço do usuário**. A interface desenvolvida deve buscar atender alguns objetivos básicos como:

* **Máxima realização**, ou seja, a interface deve ser planejada de forma que o usuário seja mais produtivo. Os elementos e funções devem ser organizados de acordo com as atividades a serem realizadas.

* **Mínima distração**, ou seja, a interface não deve tirar o foco do trabalho a ser realizado. Nada de animações, textos piscantes e cores fortes desnecessárias.

* **Mínima atenção**, ou seja, a interface deve ser feita de forma que o usuário não cometa erros irreversíveis por engano. As ações devem ser, sempre que possível, reversíveis. Caso contrário, confirmações devem ser pedidas, mas não com alta freqüência a ponto do usuário respondê-las sem prestar atenção ao que responde.

* **Mínimo bloqueio psicológico**, ou seja, evitar **pânico**, **aborrecimento**, **desconforto**, **confusão**, **frustração**...

De certa forma, não há como negar que o projeto de interfaces ainda **tem um caráter artístico**, uma vez que os critérios técnicos ainda não são suficientes para garantir uma boa usabilidade.

Por esta razão, a avaliação da qualidade tem um importante papel no projeto de interfaces. Os **critérios de qualidade primários** para esta medida são o **tempo de realização**, a **acuidade** (índice de acertos) e o **prazer do usuário**. Tais critérios são **de difícil medida**. Em algumas aplicações tais critérios podem ser distintos. Por exemplo, em um ambiente de **produção pura**, o **critério** do **prazer** do usuário costuma **não** ser **contabilizado**, já num ambiente de criação este critério **é fundamental**.

Mesmo com medida difícil, a chave para o bom desempenho nos critérios primários é atender aos anseios do chamado "**usuário consciente**", o qual **não tolera uma alta carga de memória** (seqüências de comando muito longas, por exemplo), **não aceita sistemas projetados só para iniciantes** (sem hotkeys, por exemplo) e **exige ferramentas precisas e outras facilidades**.

Em especial, a carga de memória pode ser dividida em **carga da memória de curto prazo** (exigir muitas contas de cabeça, por exemplo), **longo prazo** (exigir que o usuário decore grandes seqüências de comandos) e **curto prazo motora** (uso de grande quantidade de dispositivos simultaneamente). A **carga de curto prazo** pode levar a um **baixo desempenho** e **frustração do usuário**. A **carga de longo prazo** afeta de forma contundente o **tempo de aprendizado** e **reaprendizado** (critérios secundários).

Como os critérios primários são de avaliação complexa, em geral são usados os **critérios secundários**, mais fáceis de medir, como o **tempo de aprendizado**, **fadiga**, **conveniência**, dentre outros.

* O **tempo de aprendizado** é o tempo gasto para que o usuário que nunca utilizou a interface consiga um nível desejado de manipulação técnica.

* O **tempo de reaprendizado** é o tempo que um usuário antigo do sistema demora para relembrar a utilização da interface e obter o nível desejado de manipulação técnica, depois de um tempo sem usar o recurso.

* O **tempo de aprendizado motor** é o tempo gasto para executar uma ação já conhecida.

* **Fadiga** pode ser causada pelo uso repetitivo de elementos nas telas (telas parecidas), estímulos desagradáveis (visuais ou auditivos), cargas de memória exageradas, excesso de movimentação necessário (projeto ruim da interface), incerteza sobre operações (seqüências de operações pouco claras). As conseqüências diretas são o **aumento da taxa de erros** e **tempo de execução das tarefas**, além da **redução do prazer do usuário**.

* **Conveniência** é uma avaliação da semelhança com o mundo real (tornando o sistema natural para o usuário, com o uso de sliders e indicadores de ponteiros em aplicações industriais, por exemplo) e **espaço disponível** para o usuário, afetando diretamente o prazer do usuário no uso do sistema.

Assim, deve-se tentar **acomodar todas as necessidades dos diferentes tipos de usuários**, fornecendo **comandos simples e de fácil acesso para os iniciantes**, disponibilizando **acesso completo por menus e teclas de atalho para usuários mais avançados**. Além disso, deve-se buscar uma **taxa de erros mínima ou**, em alguns casos, **nula**.

## 5. Escolher as Tarefas Interativas

Um sistema interativo é aquele em que existe uma **ação** (**realizada pelo usuário**) e uma **resposta** (**realizada pelo computador**), sendo que ambas formam uma **seqüência interativa**.

Assim, devem ser selecionadas **quais ações dos usuários** (uma seleção no menu, por exemplo) **devem acionar quais efeitos** e **como o computador responderá ao usuário** (movimentação de um ícone sobre a tela, por exemplo).

## 6. Selecionar Dispositivos de Interação

Uma **tarefa interativa** qualquer quase sempre **pode ser realizada através de** diversas **técnicas interativas diferentes**, por diversos dispositivos. Por exemplo, para **selecionar um ícone na área de trabalho** é possível **apontar com o mouse**, **usar as setas do teclado** ou mesmo **dizer o nome do ícone**, em sistemas com reconhecimento de voz.

Há diversos dispositivos de interação, sendo os mais comuns os **teclados**, **mouses**, **trackballs**, **joysticks**, **microfones** e **tablets**.

## 7.Bibliografia

FOLEY, J. D; WALLACE, V.K; CHAN, P. *The Human Factors of Computer Graphics Interaction Techiniques*. IEEE CG & Applicaitions, Los Anlamitos, nov.1984, p. 13-48.

FOLEY, J. D. et al. *Computer Graphics Principles and Practices*. Addison-Wesley, Reading, 1990.

Bibliografia Complementar:

DIX, A. et al. *Human-Computer Interaction***.** Edinburg, England: Prentice-Hall, 1997.

PREECE, J, et al. *Human-Computer Interaction.* Addison-Wesley, 1994.

Notas da Aula 05: Aspectos Ergonômicos
Prof. Daniel Caetano

**Objetivo**: Apresentar as principais características humano-computador e sua influência no projeto de IHC.

## 1. Introdução

Uma das **maiores preocupações do desenvolvimento** de um sistema de informações, além da própria informação, é a **interface do sistema com o usuário**, que irá receber comandos do usuário e transmitir-lhe informações.

A interface de um sistema de informações **deve ser adaptada à toda sorte de usuários** que possa vir fazer uso da mesma, **observando as características humanas**, tanto em seus pontos fortes quanto seus pontos fracos. Dentro do possível, também devem ser incluídas nas especificações de projeto as **pessoas com necessidades especiais**.

Esta questão é de importância ainda maior para o desenvolvimento de sistemas para público geral como caixas de banco, terminais de informações turísticas e transporte, etc. Nestes sistemas não há qualquer tipo de controle sobre os usuários que utilizam o sistema e suas limitações, sejam elas físicas ou não.

## 2. Fatores de Influência no Desenvolvimento de Interfaces

Antes de qualquer colocação, é sempre importante lembrar que **nunca é adequado que o desenvolvedor se baseie apenas em sua percepção** para avaliar as necessidades e características de uma interface. **Desenvolvedores costumam tolerar cargas cognitivas extremamente altas**, por exemplo, e portanto não devem ser considerados como exemplo de usuário médio.

Assim, um estudo mais detalhado dos fatores que determinam a interface se faz necessário, sendo eles os aspectos ergonômicos e cognitivos, além da aparência. Na aula anterior já foram apresentados alguns aspectos da cognição e ergonomia. Agora serão apresentados alguns detalhes a mais.

### 2.1. O Ser Humano

Principal foco do desenvolvimento, tem limitações de percepção e aprendizado. Este estudo pode ser categorizado em "canais de entrada e saída", "memória" e "pensamento".

**Canais de Entrada e Saída**: São basicamente os **órgãos sensoriais e motores**. Os mais importantes são considerados, em termos de interação com computador: a visão, a audição, o tato, os dedos, a voz, posição da cabeça e olhos.

***A visão***: principal órgão sensor humano, tem **maior capacidade de distinção de padrões no centro** da imagem focada, sendo esta a região com **melhor coloração**. Nas **regiões periféricas** quase **não há detecção de cores, a percepção de padrões é pobre,** mas a **detecção de movimentos é alta**. O uso de **fundos claros com imagens escuras aumenta a percepção de informações,** mas também **de *flickering*** e, portanto, **o cansaço visual**.

É importante ressaltar que a capacidade de **percepção de cores, acuidade, detecção de movimentos** etc., **podem variar bastante** de um indivíduo para outro e, portanto, tais variações devem ser consideradas.

***A audição***: é o segundo canal mais importante para obtenção de informações e, nos **deficientes visuais**, sua **importância** é ainda **maior**. Permite estabelecimento da posição de eventos, identificação da origem dos sons (com base na memória) e construção de modelos de imagens mentais, além do ***feedback* de** algumas **ações**, como a indicação de tecla pressionada. Existem **diferenças de percepção de freqüência, intensidade e timbre** (envoltória de intensidade). A **execução contínua** de som pode **causar irritação**.

**Sons repetitivos (ou de ambiente)** podem passar a ser **ignorados** quando o usuário foca alguma atividade. **Sons diferentes tiram a atenção do usuário** e podem **causar irritação** e, portanto, devem ser usados com cuidado.

***O Tato***: apesar de ser considerado menos importante que a visão e audição sua importância é grande na eficiência de execução de atividades. O tato **proporciona *feedback***, permitindo um maior conforto (tecla pressionada). A detecção da **posição dos membros** do corpo também é um fator importante para uma **boa eficiência e conforto**.

**Memória**: Permite que informações sejam armazenadas e recuperadas para uso posterior. O preenchimento da memória com informações chama-se ***aprendizado***. Além das memórias de **curto prazo** (informações parciais de processamentos em execução) e **longo prazo** (armazenamento final) que foram comentadas na aula anterior, há também a **memória sensorial** (de curto prazo), que pode ser dividida em memória **icônica**, memória **ecóica** e memória **háptica** (relacionada ao tato, e envolve a memória de curto prazo motora).

A maioria das informações **entra pela memória sensorial, passa para a memória de curto prazo e depois é transferida para a de longo prazo**, quando é interligada com outras informações existentes. Este processo é **facilitado pela repetição** ao longo do tempo. Após armazenada, a recuperação por ser feita por **recuperação simples** (seguindo estruturas de conhecimento) ou **reconhecimento** (percepção de que a memória já existia quando ela é reapresentada). A recuperação (processo mais comum) é **facilitada pelo agrupamento da informação em categorias**, o mesmo ocorrendo com a apresentação de "dicas" ou partes da informação.

Por estas razões, tarefas similares devem ser programadas com seqüências similares, para facilitar o aprendizado, e seqüências muito longas devem ser evitadas, pois ocorre sobrecarga da memória de curto prazo antes que as informações tenham passado para a

memória de longo prazo. Além disso, o uso de ícones junto com os nomes das funções pode facilitar o processo de recuperação.

Como os **indivíduos podem ser muito díspares no quesito memória** e capacidade de memorização/aprendizado, é fundamental utilizar a **menor carga de memória possível**. Por esta razão, sempre é importante inserir dispositivos de ajuda detalhada, caso todas as precauções anteriores falharem.

**Pensamento**: atividade de **manipular informações armazenadas**, de acordo com a capacidade de recuperá-las ou não. É possível dividir o pensamento em duas categorias: "raciocínio" e "solução de problemas".

O ***raciocínio*** é a capacidade de **chegar a informações novas a partir de conhecimentos anteriores**, havendo o raciocínio **dedutivo** (concluir o efeito a partir da causa), **indutivo** (generalização do efeito da causa) e **abdutivo** (concluir as causas de um efeito). Uma interface que se comporta de acordo com o esperado pelo usuário, nas três formas de raciocínio, é chamada de uma **interface consistente**.

A ***solução de problemas*** é a capacidade de **entrontrar solução para uma tarefa desconhecida**, usando conhecimento existente, detectando semelhanças e diferenças da nova situação com relação às anteriormente conhecidas. Existem duas teorias comuns para explicar estes mecanismos: a **Gestalt** (reprodutiva/tentativa-e-erro ou produtiva/ter-idéias) e a do **Espaço de Problemas** (Estados/Transições/Proximidade-de-Solução). O **uso de analogias** facilita a identificação de padrões de solução já conhecidos para problemas novos... mas devem ser usadas com **cuidado**, pois são bastante ligadas ao conhecimento prévio do indivíduo.

## 2.2. O Computador

O computador também tem forte influência no desenvolvimento da interface. Assim como os humanos, os computadores também possuem limitações em cada um de seus subsistemas, "canais de entrada e saída", "armazenamento" e "processamento".

**Canais de Entrada e Saída**: Mais simples que os humanos, mas em maior variedade.

**Teclado**: entrada de dados por toque. Exige precisão motora, mas pode ser adaptado. Indicadores para cegos nas teclas 'F', 'J' e '5'. Existem teclados braile.

**Mouse e Trackball**: dispositivos de posicionamento e seleção. Podem ser usados para desenho. Desconfortáveis para texto e exigem capacidade motora. Depende da visão.

**Joystick**: dispositivo de posicionamento e apontamento através de uma alavanca direcional. Não exigem muita destreza, pouco sensível.

**Touch-screen e Lightpen**: dispositivos apontadores por indicação direta. Em outras palavras, aponta-se na tela a posição que se deseja selecionar. Facilidade de uso relacionada ao tamanho do objeto a selecionar.

**Touch-pad e Tablet**: dispositivos de posicionamento, voltados à digitalização de figuras, onde é usada uma caneta especial para traçar a figura sobre o dispositivo. Requer boa coordenação.

**Scanner e Câmera**: dispositivos de captura de imagem. Podem ser usados desde dispositivos de identificação (como leitores de digitais) até de auxílio a outros dispositivos (como de reconhecimento de gestos ou *eyegaze*). A dificuldade de uso pelo usuário depende diretamente da forma com que o equipamento deve ser usado.

**Sistema de Reconhecimento de Voz**: permitem a entrada de comandos ou de texto. Requer pouco conhecimento prévio do usuário. Possuem alta taxa de reconhecimento, mas são bastante suscetíveis a ruídos de fundo.

**Sistema de Reconhecimento de Movimentos**: traduzem movimentos do usuário em comandos para o computador. Ex.: luvas, *helmets* ("capacetes" que detectam movimentos da cabeça), dispositivos de acompanhamento de vista do tipo *eye-tracking* ou *eyegaze*. A habilidade necessária depende do dispositivo.

**Monitor de Vídeo**: dispositivo para a apresentação de imagens. Uso depende de capacidade de leitura ou capacidade de interpretação de figuras.

**Monitor Braille**: dispositivo de imagem voltado primariamente a deficientes visuais.

**Impressora**: capaz de fornecer informações na forma textual ou gráfica através de mídia física (papel, plástico, tecido, etc.), podendo ser essa representação em cores ou tons de cinza ou até mesmo em braille. A compreensão da informação fornecida, em geral, depende de capacidade de leitura ou capacidade de interpretação de figuras.

**Dispositivos de Áudio**: usados para produzir os mais diversos tipos de sons. Desnecessidade de conhecimento prévio por parte do usuário.

**Armazenamento**: similar à memória dos seres humanos. Existe a memória dos dispositivos (análoga à sensorial) representada pelos ***buffers*** dos diferentes dispositivos. Existe também a memória de curto prazo, a memória conhecida como **RAM** (*Random Access Memory*), que é usada como memória de trabalho. Finalmente existe também a memória de longo prazo, para armazenamento permanente, como os **discos, *hard-disks*, *CD-ROM*s**, etc. Cada uma delas tem **vantagens e desvantagens** com relação à **velocidade** e também com relação ao **uso**: algumas memórias podem ser lidas e escritas, outras podem ser apenas lidas.

**Processamento**: um dos **maiores limitantes** dos computadores e afetam grandemente o projeto de interface. Baixa capacidade de processamento limita a complexidade da interface, sob pena de torná-la lerda e desajeitada.

## 3. Detalhes de Ergonomia

Ergonomia é o estudo das características físicas do ambiente em que a interação humano-computador ocorre. Estas características são ligadas não apenas à questões de saúde, mas também à compreensão da lógica de funcionamento do sistema e também conforto do usuário.

A fim de evitar confusão e facilitar o aprendizado, os **controles devem ser agrupados segundo alguma regra**, como funcionalidade, seqüência de uso ou freqüência de uso. Além disso, os **controles devem dispostos em uma posição confortável** ao uso de todos os usuários, independente de seu tamanho.

Também é importante observar o **posicionamento dos equipamentos físicos**, que devem ser localizados em ambientes adequados ao uso por deficientes, como os especificados na NBR 9050.

Sempre que possível as **informações devem ser apresentadas** ao usuário **em mais de um formato**: um que facilite a identificação de situações críticas através de rápida inspeção visual e outra que mostre detalhadamente os valores de interesse. É preciso tomar cuidado, porém, para **não "poluir" o campo de visão** do usuário, distraindo o usuário das informações mais importantes. Ainda com relação à apresentação, as **cores devem ser usadas com cuidado** para não criar interfaces pouco intuitivas.

## 4. Deficiências e Soluções

Segundo a **OMS (MPT, 2006), 10% da população** média dos países **sofrem de algum tipo de deficiência**, sendo especificamente:

| | |
|---|---|
| **- Mental / Dificuldade de Aprendizagem.** | **5,0%** |
| **- Física / Motora** | **2,0%** |
| **Auditiva** | **1,5%** |
| **- Múltiplas** | **1,0%** |
| **- Visual** | **0,5%** |

**No Brasil**, este número cresce para **14,5% (IBGE, 2000)**. Somado a este valor os **8,5% da população que são idosos** (AME, 2004), que também possuem dificuldades de acesso, temo-se que quase um quarto da população brasileira pode possuir algum tipo de problema relativo à acessibilidade.

Este número elevado torna maior a preocupação com pessoas com necessidades especiais no projeto de interfaces cujo objetivo é o público geral. Sendo a configuração inclusiva de ambientes sociais uma tendência atual, muitas propostas têm sido feitas nos últimos anos, para promover o que hoje é chamado de *inclusão social e digital* aos portadores destes tipos de deficiência.

Segundo o Centro para o Design Universal na North Caroline State University (EUA), são **requisitos fundamentais para um Projeto Universal**:

- **Uso eqüitativo**: o produto do projeto deve ser útil e adquirido por pessoas com as mais diversas habilidades.
- **Flexibilidade no uso**: o resultado do projeto deve prever e permitir a acomodação de vasta gama de preferências e habilidades individuais.
- **Simples e intuitivo**: o uso do produto deve ser fácil de compreender, independente de experiência, conhecimentos anteriores, habilidades lingüísticas ou nível de concentração.
- **Informação perceptível**: o produto deve apresentar as informações necessárias ao usuário de forma efetiva, independentemente das condições do ambiente ou das capacidades sensoriais do indivíduo.
- **Tolerância ao erro**: o produto deve minimizar o risco e as conseqüências de ações acidentais ou não intencionais.
- **Baixo esforço físico**: o produto deve ser confortável no uso, causando o mínimo de fadiga.
- **Tamanho e espaço para aproximação e uso**: o produto deve ter tamanho e espaços adequados devem ser fornecidos para aproximação do usuário, além de espaço para operação independente do tamanho do corpo, postura ou mobilidade.

**São basicamente critérios ergonômicos** que garante o uso de um equipamento ou serviço por qualquer indivíduo, tenha ele limitações ou não.

## 7.Bibliografia

CAETANO, D. J. *Fatores Humanos nas Interações Humano-Computador e Soluções para Usuários com Necessidades Especiais.* Trabalho de Disciplina de Doutorado. Escola Politécnica da Universidade de São Paulo, 2006. Disponível em: < http://www.caetano.eng.br/aulas/fb/ihc/Fatores_Humanos_nas_Interacoes_Humano_Computador_e_Solucoes_para_Usuarios_com_Necessidades_Especiais.PDF >.

Notas da Aula 06: Desenvolvimento de Interfaces
Prof. Daniel Caetano

**Objetivo**: Detalhar algumas das regras de desenvolvimento da interface de ação.

## 1. Introdução

Como foi visto, os passos para o desenvolvimento voltado ao usuário são:

a) Compreender e assimilar os conceitos da aplicação;
b) definir a linguagem de interface;
c) considerar fatores ergonômicos;
d) escolher as tarefas interativas;
e) selecionar dispositivos de interação.

Destes, o item (a) é coberto pela análise de sistema e os itens (c) e (e) estão relacionados às capacidades de percepção e equipamentos disponíveis, analisados com mais cuidado na aula anterior.

Nesta aula serão abordados maiores detalhes sobre a definição da linguagem da interface (b) e escolha das tarefas interativas (d), objetivando **definir mais características das interfaces de ação**, tendo como **objetivo** tornar o **sistema agradável**, **confiável** e **produtivo/prático de usar**. Também neste caso é necessário realizar algum tipo de **avaliação de desempenho**, onde os aspectos primários a serem considerados são o **tempo de resposta** do sistema (tempo em que uma ação é executada), a **aceitabilidade** (se o usuário gosta do sistema) e **eficiência do usuário** (se os usuários conseguem desempenhar suas funções melhor e mais rapidamente).

É claro que também aqui são adotados conceitos de psicologia perceptiva, psicologia cognitiva e também as questões relacionadas a ergonimia/fatores humanos. Entretanto estes serão sempre apresentados de maneira aplicada, não na forma de teorias.

## 2. A Interface de Ação

Como já foi visto, a interface de ação é a forma com a qual o usuário informa o que ele deseja para o computador. Para isso, é preciso uma **linguagem de comunicação** e ela deve sempre ser **natural**, com **poucas regras**, **vocabulários simples**. Além disso, os **conceitos de fácil aprendizagem** (ou já conhecidos pelo usuário) e, sempre que possível, deve ser **expansível**, com facilidade de **personalização**.

Há alguns **elementos da interface de percepção** que são **relacionados** à interface de ação e que merecem destaque inicial, que são o ***feedback*** (informação imediata de que o equipamento está compreendendo o que o usuário está fazendo), permitindo que o usuário corrija enganos mais rapidamente caso não esteja se expressando corretamente. Neste aspecto, a existência do "***undo***" e de sistemas de **ajuda *online*** são bastante importantes.

## 2.1. Modelos de Comunicação

Como já foi visto, a interface de ação pode ser projetada com base no **modelo lingüístico** ou no **modelo orientado à implementação**. Os modelos orientados à implementação, em geral, seguem regras de um modelo lingüístico já prontos, que devem ser os mesmos do sistema operacional.

Mas o que é o **modelo lingüístico**? Primeiramente, existem os **elementos** de comunicação que, no caso da **interface de ação** são as **ações do ser humano**. No caso da **interface de percepção**, são os **gráficos**, **sons**, **textos** e **sensações táteis**.

O modelo lingüístico é composto por quatro **níveis**:

**1. Nível Conceitual**
**2. Nível Semântico**
**3. Nível Sintático**
**4. Nível Léxico**

Os níveis **1 e 2** são responsáveis pelo **significado** da mensagem, enquanto os níveis **3 e 4** são responsáveis pela **forma** da mensagem.

**Projeto Conceitual**: Identifica os conceitos chave da aplicação, como **objetivos e requisitos**. É neste projeto que são definidos os **objetos** da aplicação e suas **relações e operações**. Tais objetos devem ser imaginados de forma que o usuário os compreenda facilmente, se possível usando **metáforas** e de forma **consistente**. Existirão objetos da aplicação e de controle (grade, cursor, etc).
**Exemplo: Um editor de textos.**
**Objetos: Linhas e Arquivos.**
**Relações**: Arquivo = Conjunto ordenado de linhas.
**Operações**: (de linha): inserir, apagar. (de arquivo): apagar, copiar.

**Projeto Semântico**: **Detalhamento de cada função**, definindo as **informações necessárias**, **erros** possíveis e seu **tratamento** e os **resultados desejáveis**.

**Projeto Sintático**: Especifica **seqüências de entrada e saída**. Na **entrada** há uma seqüência de **tokens**, como "**MOVER 10M MESA**", onde temos um primeiro elemento de ação, o segundo é um parâmetro para a ação e o terceiro é o objeto em que a ação é aplicada. Na **saída**, a seqüência inclui uma **noção espacial/temporal**, envolvendo **organização da tela**, **animações**, etc. A sintaxe pode ser **pré-fixada** (comando antes do objeto) ou **pós-fixada** (comando depois do objeto)

**Projeto Léxico**: Especifica **como "converter" entradas dos dispositivos** (clique no mouse, teclas) **nos tokens** de entrada (comandos) e como converter as **primitivas do sistema** (linhas, cores) **em saídas úteis** (figuras informativas).

## 2.2. Feedback

O feedback (uma forma de o usuário perceber que o sistema está recebendo suas ações) deve, em geral, ser implementado nos níveis **léxico (imediato), sintático (1 segundo) e semântico (tempos maiores)**.

Exemplos de feedback **léxico** são os **movimentos** do cursor do **mouse** ou a impressão de caracteres digitados na tela. Exemplos de feedback **sintático** seriam a **mudança de cor** de um **ícone** clicado ou a mudança de cor de uma opção de menu selecionada. Exemplos de feedback **semântico** seria uma **barra** com a **indicação de trabalho** sendo executado ou textos informativos sobre o que está acontecendo.

É comum a apresentação de erros e mensagens em **posição fixa**, ou ainda o uso de caixas de mensagem na **posição do cursor**.

## 2.3. Ajuda ao Usuário

A primeira característica importante é que a ajuda deve ser **planejada para usuários iniciantes**, passando pelos **intermediários** e também os **avançados**. Para suprir estas necessidades, existem basicamente dois tipos de **ferramentas**: as de **prompting e help online**, podendo ser **contextualizados ou não**, **obstrutivos ou não**.

**Prompting não obstrutivo**: **cursor piscando**, indicando entrada de texto. Uso de uma **escala** para indicar a necessidade de fornecimento de um valor.

**Prompting obstrutivo**: abertura de **janela com opções** (menus).

**Ajuda Online não contextualizada**: **menu "Ajuda > Tópicos da Ajuda"**.

**Ajuda Online contextualizada**: **pressionar F1** com algum item selecionado.

**Documentação (Online ou não)**: **Manual completo**, totalmente detalhado.

## 2.4. Tratamento de Erros

Primeiramente, deve-se procurar **evitar que o usuário erre**, não permitindo seleção de comandos ilegais, não permitir apagar algo que já não mais existe, não permitir atributos a objetos que não os possuem, etc. Em outras palavras, deve-se usar **comandos sensíveis a contexto**: só estão disponíveis quando são válidos.

Entretanto, algumas vezes o usuário comanda **ações válidas indesejadas**, por engano. Nestes casos, é importante permitir o **"undo"** (desfazer última ação) e, muito importante,

permitir que seja possível **abortar** comandos sendo executados, antes que eles sejam finalizados. Sempre que possível, deve ser realizado um **backup** automático antes da execução dos comandos.

O **tratamento de erros** tem um **custo semântico alto**, dado que o computador precisa analisar e compreender a ação do usuário minimamente, de forma a detectar ações inválidas ou indesejáveis, exigindo confirmações, por exemplo. Entretanto, o **não tratamento** dos erros podem causar uma grande **redução de produtividade**, além da **frustração do usuário**.

## 3. Requisitos para boa avaliação da Interface de Ação

### 3.1. Tempos de Resposta

O tempo de resposta do sistema está intimamente relacionado à **satisfação e confiança** do usuário. Os tempos aceitáveis dependem da aplicação e dos usuários, entretanto.

Em geral, para o **feedback léxico e sintático** o tempo de resposta deve ser da ordem de **décimos de segundo**, enquanto para o **semântico** este tempo pode ser até da ordem de **2 segundos**. É importante lembrar que quanto **menor este tempo, melhor a reação** do usuário.

A **não observação** destas regras causa **desconforto e dúvida**, além da **perda de concentração** do usuário, prejudicando a produtividade e a aceitação do sistema pelo usuário. Para aplicações web, por exemplo, não deve ser admitido um tempo superior a 10s para carregar uma página, pois o usuário sentirá um extremo desconforto com relação à velocidade de navegação.

É importante **padronizar os tempos de resposta** em termos da **execução da operação** como um todo. Se o tempo de uma resposta for muito **além do tempo médio**, o usuário pode pensar que o **sistema travou**. Se o tempo de uma resposta for **muito curto**, o usuário pode ter **dúvidas se a ação** de fato **já foi realizada**.

### 3.2. Consistência

Uma **interface** é considerada **consistente** quando seu **modelo conceitual**, sua **semântica, sintaxe e entradas/saídas** são **uniformes**, com **regras simples** e **não possuem exceções ou omissões**. Esta características **permite** que o usuário **generalize seu conhecimento** e **não se frustre**.

A **chave** para obter consistência reside em usar sempre os **mesmos códigos**. Isso quer dizer que cores devem sempre ter o mesmo significado, as mensagens sempre devem surgir nas mesmas posições e itens de menu devem sempre aparecer nas mesmas posições.

A consistência tem um **papel fundamental na redução da carga de memória**. **Juntamente com o uso de um modelo conceitual similar ao universo do usuário**, permite que este usuário use seus conhecimentos do dia-a-dia para o perar o sistema e, a partir do aprendizado de algumas operações, possa realizar praticamente todas as outras.

## 4.Bibliografia

FOLEY, J. D; WALLACE, V.K; CHAN, P. *The Human Factors of Computer Graphics Interaction Techiniques*. IEEE CG & Applicaitions, Los Anlamitos, nov.1984, p. 13-48.

NETTO,A.A.O.Modelagem e GerÛncia de Interfaces com o Usußrio.Ed.Visual Books, 2004.

FOLEY, J. D. et al. *Computer Graphics Principles and Practices*. Addison-Wesley, Reading, 1990.

Bibliografia Complementar:

DIX, A. et al. *Human-Computer Interaction***.** Edinburg, England: Prentice-Hall, 1997.

PREECE, J, et al. *Human-Computer Interaction.* Addison-Wesley, 1994.

Notas da Aula 07: Inteligência Artificial em Interfaces com o Usuário
Prof. Daniel Caetano

**Objetivo**: Apresentar a idéia por trás do uso de Inteligência Artificial em interfaces.

## 1. Introdução

O **uso da Inteligência Artificial em interfaces** certamente **não é novo**. Entretanto, as **inovações recentes** nesta área são **crescentes** e, com uma grande probabilidade, **devem promover** em breve uma **grande mudança** na forma como os equipamentos atuais são operados.

A **idéia** por trás do uso de Inteligência Artificial em interfaces é **tornar o uso** de um aplicativo ou página **mais simples**. Alcançar este objetivo passa por **identificar a necessidade** do usuário **em tempo real** e **promover auxílio contínuo**, propiciando um uso mais agradável do software.

A Inteligência Artificial possui **uso** tanto **na Interface de Ação** - ajudando o usuário a transmitir sua informação ao equipamento - quanto **na Interface de Percepção** - auxliando ao usuário compreender as informações oferecidas pelo sistema.

## 2. Inteligência Artificial na Interface de Ação

Talvez o maior campo para uso de Inteligência Artificial em interfaces seja com relação à Interface de Ação. Isso se deve ao fato que a **Interface de Ação**, apesar de todos os esforços dos projetistas, geralmente **está longe de permitir uma comunicação natural** do usuário com a máquina. Em outras palavras, a Interface de Ação envolve, quase sempre, algum nível de aprendizado por parte do usuário.

Aplicar Inteligência Artificial na Interface de Ação significa **fazer com que** o **computador "intua" o que o usuário deseja fazer** e, com base nisso, **o auxilie a executar** esta tarefa. Este auxílio pode ser de diversas formas, como apresentado a seguir:

**Eliminando Alternativas**: Ao perceber as tarefas que o usuário quer (ou quais ele pode querer) realizar, o sistema simplesmente pode **"sumir" com as opções menos prováveis**. Um exemplo deste tipo de aplicação já existe, ainda que de funcionamento questionável, nas aplicações do Microsoft Windows. É o caso dos **menus dinâmicos**, em que as **opções pouco usadas são escondidas**, tornando os menus menores e mais fáceis de navegar.

**Sugerindo o Próximo Passo**: Ao perceber as tarefas que o usuário quer realizar, o sistema pode **oferecer dicas do próximo passo**, com mensagens explicativas, ícones animados dentre outros. Um exemplo de implementação deste tipo é **Google Suggest** (

http://labs.google.com/suggest/ ), uma versão do sistema de busca Google que faz sugestões de busca em tempo real, à medida em que o usuário digita os termos de sua própria busca. Outra forma de implementação permite ao usuário encontrar coisas já visualizadas por ele usando o sistema, como no sistema "**Stuff I've Seen**" proposto pela Microsoft para o **Windows Desktop Search** do Windows Vista.

**Forçando o Próximo Passo**: Uma aplicação de sistema especialista, é uma alternativa em que o **usuário seleciona** explicitamente **que deseja ser tutoriado** em algum processo, e o **sistema o ajuda a tomar as decisões**. Um exemplo deste tipo de aplicação são, por exemplo, os **Wizards** de instalação do Windows, os Wizards de criação de arquivos comprimidos do WinZip ou mesmo os Wizards de conexão à rede.

**Compreensão de Linguagem Natural**: Uma das grandes dificuldades do ser humano ao lidar com as máquinas é a dificuldade em transmitir comandos a este equipamento. Isto ocorre por que esta comunicação normalmente é feita em uma linguagem imposta pela máquina e não na própria linguagem do ser humano. Ocorre que a linguagem do ser humano é fundamentalmente ambígüa e, se usada para transmitir comandos ao computador, exigirá que o **equipamento** seja **capaz de eliminar a ambiguidade**, seja por conta própria, seja por perguntas ao usuário. Acredita-se que este será o maior avanço futuro com relação a interfaces, produzido nas chamadas **CUI** (**Conversational User Interfaces**).

## 3. Inteligência Artificial na Interface de Percepção

Também na área da Interface de Percepção existe um bom campo de aplicação de Interligência Artificial, usualmente **auxiliando** o usuário **na compreensão dos resultados** ou do prórpio software. Algumas possíveis aplicações são apresentadas a seguir:

**Adequação de Informações**: Em muitos casos, a maior dificuldade que os usuários encontram é na compreensão dos textos e figuras apresentados pelo sistema, devido à linguagem e aos tipos de gráficos utilizados. Uma aplicação interessante de Inteligência Artificial seria a identificação do tipo de usuário e, com base nisso, **modificar o vocabulários** e **tipos de informações apresentadas**.

**Apresentação Seletiva de Informações**: Alguns dos programas mais úteis, após concretizar suas ações, podem apresentar muitas informações ao usuário, em relatórios extremamente confusos. Uma das formas interessantes de aplicar inteligência artificial nestes casos é **identificar, de acordo com o contexto das operações** realizadas, **quais informações de saída são efetivamente importantes** e interessantes, ressaltando-as nas primeiras páginas de relatório, transferindo para as páginas seguintes as informações que normalmente não são importantes naquele contexto. Um exemplo deste tipo de aplicação são os **filtros de SPAM** de e-mail, como aqueles que usam **sistemas de "aprendizado" bayesianos**.

**Ajuda Online de Contexto Dinâmico**: Em geral, os sistemas de help online contextualizados simplesmente verificam qual é a última opção que o usuário marcou e então

apresentam uma ajuda com relação àquela opção. Entretanto, algumas vezes o usuário não tem dúvidas sobre aquela opção em sim, mas sim sobre qual é a próxima a ser realizada. Esta situação **exige que o sistema compreenda a ação que o usuário está tentando desempenhar** e, quando o usuário solicitar ajuda, tal **ajuda será fornecida explicando** não só o último comando selecionado, mas também **os possíveis próximos passos e suas conseqüências**. Este tipo de solução é similar à "Sugestão de Próximo Passo", mas enquanto esta é referente à Interface de Ação (que se modifica sozinha), a Ajuda Online de Contexto Dinâmico exige solicitação do usuário (apertando F1, por exemplo). Uma outra forma de implementação deste tipo, ainda que questionável, é o **Sistema de Assistentes**, como o usado nos aplicativos **Office da Microsoft**.

## 4.Bibliografia

Accelerating Change 2005 - < http://www.accelerating.org/ac2005/themes.html >, acessado em 20 de dezembro de 2005.

Adaptive Systems and Interaction - < http://research.microsoft.com/adapt/ >, acessado em 12 de agosto de 2006.

Conversational Architetures- < http://research.microsoft.com/adapt/conversation/ >, acessado em 27 de setembro de 2006.

Stuff I've Seen - < http://research.microsoft.com/adapt/sis/index.htm >, acessado em 27 de setembro de 2006.

The Lumiere Project - < http://research.microsoft.com/~horvitz/lumiere.htm >, acessado em 13 de agosto de 2006.

The TaskGallery - < http://research.microsoft.com/adapt/taskgallery/ >, acessado em 27 de setembro de 2006.

Notas da Aula 08: Desenvolvimento de Interfaces II
Prof. Daniel Caetano

**Objetivo**: Apresentar o processo de desenvolvimento da interface como um todo.

## 1. Introdução

Uma vez que a interface de ação esteja definida, bem como suas características de inteligência, chega a hora de **transformar** estas **documentações** em um **projeto de software** e **implementá-lo**, ao mesmo tempo em que se desenvolve o **aspecto visual**, ou seja, da interface de percepção, do software.

O **processo** de projeto e implementação de uma interface de software é **complexo** e **criar meios para facilitar** este desenvolvimento **pode melhorar o desempenho da equipe de projeto**, **facilitar alterações futuras** e **facilitar a manutenção da consistência** (dado que muitas ferramentas já possuem algum tipo de suporte com relação a isso).

É importante lembrar que **existe** uma certa **distância entre** a **implementação** e a **utilização** de uma interface, ou seja, entre a **visão interna** e **externa** de uma interface. Mas a **forma como** ela **é implementada** tem **influência direta** no seu **comportamento** perante o usuário, podendo fazer com que ele se sinta **"à vontade"** ou **"amarrado"**.

A **visão interna** de um software leva em consideração o **funcionamento interno**, **como será implementado** aquele determinado tipo de **inteligência artificial**, como será feito para o **botão mudar de cor** e **uma animação** acontecer. Além disso, **antes** mesmo de iniciar o projeto do software e a implementação é preciso **fazer a escolha** entre os já citados **modelo orientado à implementação** ou **modelo lingüístico**.

De qualquer forma, o sistema será desenvolvido através de um **processo de software**, que é um **conjunto de atividades e resultados associados**, visando a **geração de** um produto de **software**. Estes processos podem ser descritos através de **paradigmas de desenvolvimento** de software.

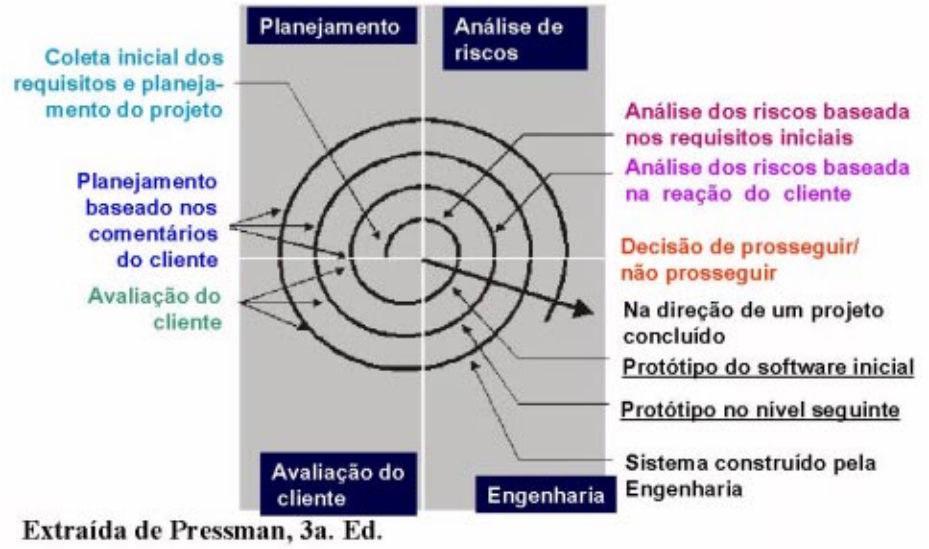
## 2. Paradigmas de Desenvolvimento

### Modelo em Cascata

Um dos paradigmas mais clássicos é o **modelo em cascata**. Embora ele não seja aplicável na prática, ele transmite uma idéia da principal relação entre as várias etapas do desenvolvimento: **análise, especificação, projeto, implementação, testes e operação final**.

Modelo do Ciclo Espiral

Um paradima também clássico que se aproxima mais da situação real é o **modelo do ciclo espiral**, ou espiral de projeto, em que são realizadas de forma sucessivas as atividades de: **coleta de requisitos**, **analise de riscos** (baseada nos requisitos), **protótipo de software** inicial, **avaliação do cliente**, **planejamento** (baseado no relatório do cliente), **análise de riscos** (baseada nos comentários do cliente) e assim por diante, como apresentado na figura a seguir.



Extraída de Pressman, 3a. Ed.

Na última etapa da espiral aparece o "**Sistema Construído pela Engenharia**", que é uma etapa **composta por** um **projeto detalhado, codificação em si, testes de unidade, testes de integração, testes de aceitação e definição da operação** do sistema.

Modelo com Prototipação

Um outro modelo, derivado do modelo em cascata, é o **modelo com prototipação**. Ele seque as **mesmas etapas do modelo em cascata**, mas ele é **repassado algumas vezes** para o **desenvolvimento do protótipo** e apenas quando o protótipo está adequado é que se parte para a **implementação final**: **análise, especificação do protótipo, projeto do protótipo, implementação do protótipo, testes do protótipo.** Passando por esta etapa, **volta-se à etapa de especificação** e recomeça-se a cascata **para concluir o projeto do produto final**.

## 3. Prototipação

Muitos estão habituados com o termo "protótipo", mas uma definição formal é sempre interessante. Segundo Sommerville (2003), **"Um protótipo é uma versão inicial de um sistema de software que deve ser utilizada para mostrar conceitos, experimentar opções de projeto e, em geral, para conhecer mais sobre os problemas e suas possíveis soluções"**.

O protótipo de software pode **apoiar diversas atividades** da Engenharia de Requisitos, como o **levantamento de requisitos** (identificar pontos positivos e negativos do sistema), **validação dos requisitos** (evidenciar erros e omissões) e **treinamento de usuários**.

## 4. Ferramentas de Implementação de Interface

Para a implementação da interface, são necessárias várias ferramentas. Considerando um ambiente "janelado", elas são, basicamente:

- **Ambiente de gerenciamento de janelas** (Windows, Gnome, etc)
- **Linguagem de programação** (C, C++, Java etc.)
- **Framework de desenvolvimento**, englobando um toolkit/bibliotecas e um ambiente de desenvolvimento de interfaces (Swing, OWL etc.).

Com relação ao Framework, um de seus componentes mais importantes são as bibliotecas. No caso do desenvolvimento de interfaces, essa bibliotecas fornecem os **widgets**, que nada mais são que objetos de tela como **botões**, **menus**, **barras de scroll**, **forms etc**.

Estes **objetos são importantes** porque a **programação de interfaces** é, em geral, **orientada a eventos**, que são gerados por tais objetos. Por exemplo: uma determinada **função é associada ao evento "clique no botão"**. Quem vai **disparar** essa **função** é exatamente o **objeto botão**, **quando** for **clicado** pelo usuário.

Assim como estes widgets, **outra fonte de eventos são os dispositivos de entrada**, como o **teclado** (KeyPress, KeyRelease, por exemplo), **mouse** (ButtonPress, Motion, por exemplo), dentre **outros** (ResizeRequest, Timer etc.).

O **gerenciador de janelas** é quem **cuida de captar estes eventos e enviá-los para nosso programa** (que pode estar sendo executado conjuntamente com diversos outros programas, na mesma tela). **Se uma tecla foi pressionada**, **apenas a janela ativa recebe** este evento. Se o usuário tentou **redimensionar uma janela**, o **gerenciador de janelas decide**, com base em vários critérios, **quais são as janelas que precisam ser redesenhadas e envia este evento** para cada uma delas.

## 4.Bibliografia

FERREIRA, M.A.G.V. *Notas de Aula - Tópicos de Comunicação Homem-Máquina*, EPUSP, 2006.
FOLEY, J. D. et al. *Computer Graphics Principles and Practices*. Addison-Wesley, 1990.
PRESSMAN, R. *Engenharia de Software*. Makron, 1995.
SOMMERVILLE, I. *Engenharia de Software*. Pearson, 2003.

Bibliografia Complementar:

NETTO,A.A.O.Modelagem e Gerência de Interfaces com o Usuário. Visual Books, 2004.

Notas da Aula 09: Projeto Visual
Prof. Daniel Caetano

**Objetivo**: Apresentar princípios que norteiam o projeto visual de interfaces.

## 1. Introdução

O projeto visual de interfaces é aquele que determina grande parte da interface de percepção. O **projeto visual afatea** basicamente a **impressão inicial** do usuário. Entretanto, a longo prazo pode influenciar muito a **usabilidade de um software**.

Os **objetivos** do projeto visual são a **clareza e consistência visual**, usando também corretamente os princípios de **codificação visual**, que serão vistos adiante, incluindo todos os elementos gráficos como formulários, menus, imagens, bem como o **layout geral da janela** da aplicação.

Entretanto, antes de mais nada, é preciso entender exatamente o que são os objetivos, ou seja, o que são clareza, codificação e consistência visual.

## 2. Clareza Visual

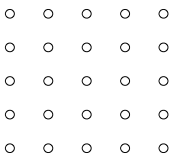
Diz-se que existe clareza visual sempre que o **significado de uma imagem for claro**, explícito para o usuário, sendo que os **elementos visuais devem ser organizados de forma a reforçar a organização da operação** do software.

Segundo Wertheimer (1930), baseado na teoria da **Gestalt**, deve-se dar **uma ênfase no todo** (organização da janela como um todo) **e não nas partes** (elementos isolados, como botões ou caixas de texto). Os conceitos que devem reger a organização da janela são os de similaridade, proximidade, fechamento e continuidade. Cada um será visto em maiores detalhes abaixo.

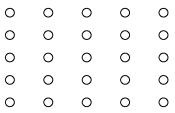
**Similaridade**: o ser humano tende a **ver da mesma forma os estímulos que possuam algumas propriedades comuns**. Ou seja: figuras similares devem representar coisas semelhantes. Por exemplo:

Proximidade: O ser humano tende a **agrupar coisas que estão próximas**. O **mesmo** ocorre **quando se usa cores**. Essas características devem ser utilizadas de forma a auxiliar o usuário a compreender quando alguns elementos fazem parte de uma mesma atividade ou não. Por exemplo:

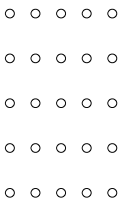
a) Visto como um quadrado



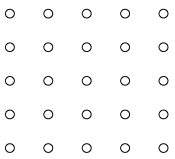
b) Visto como linhas verticais



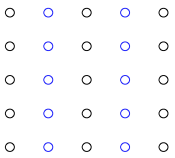
c) Visto como linhas horizontais



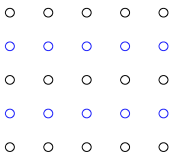
d) Visto como um quadrado



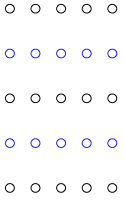
e) Visto como linhas verticais



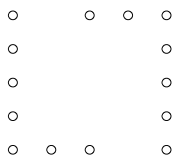
f) Visto como linhas horizontais



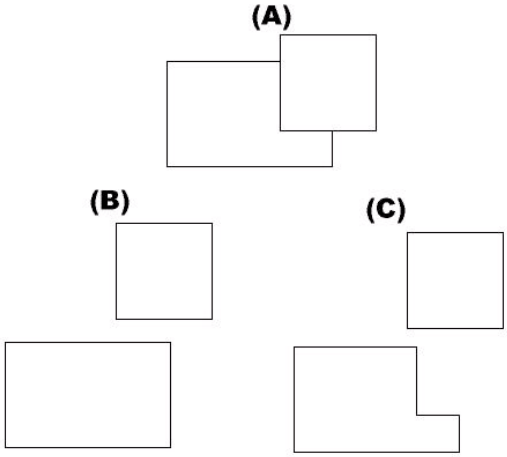
g) Visto como linhas horizontais (reforçado)



**Fechamento**: Se alguns **elementos praticamente determinam uma área** (ou podem assim ser interpretados), o **observador de fato tomará aquela representação como uma área**. Por exemplo, o desenho abaixo pode ser facilmente visto como um quadrado:

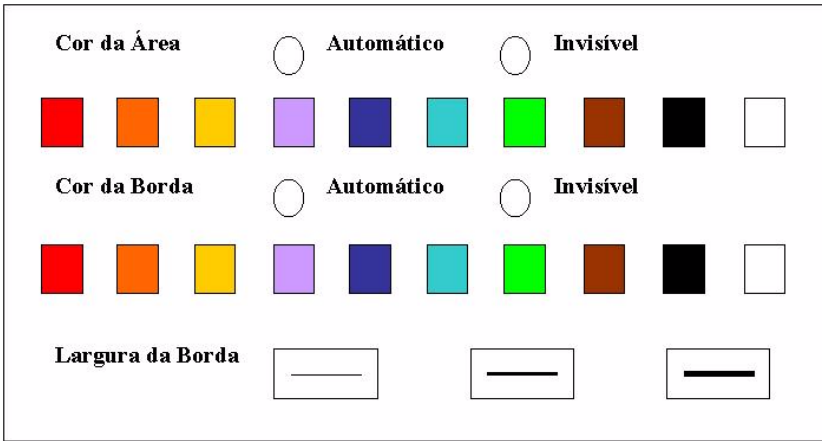


**Continuidade**: O **ser humano tende a ver figuras regulares quando observa junções complexas de linhas**. Em outras palavras, se algumas linhas parecem formar figuras regulares, o observador as perceberão como figuras regulares, mesmo que não sejam. Por exemplo, a figura (A) pode ser facilmente interpretada como a figura (B) (janelas sobrepostas), mas na realidade ela pode ser a composição dos elementos da figura (C):

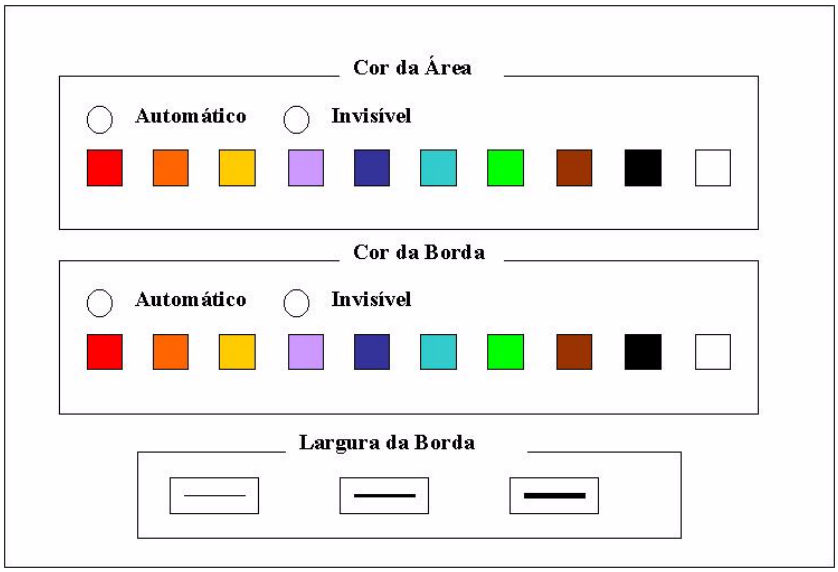



## 2.1. Aplicação de Clareza Visual a Projetos Visuais

Um exemplo de implementação de uma janela de seleção de cores e espessura para um editor de desenhos, feito sem nenhuma preocupação visual, pode ser visto abaixo (FERREIRA, 2003):

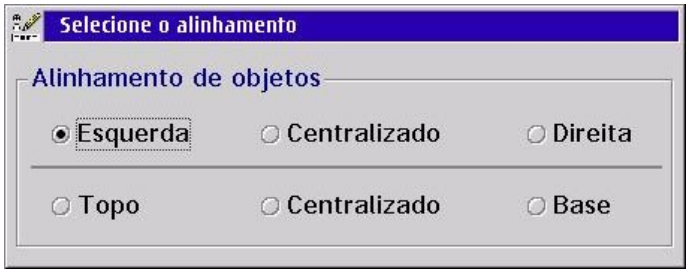


É possível observar que a representação acima é extremamente confusa. Entretanto, aplicando alguns conceitos de proximidade e fechamento, temos uma janela bem melhor, embora ainda não perfeita:




Um outro exemplo mal realizado é uma janela indicando alinhamento de objetos:


É possível observar vários problemas. Primeiro o alinhamento sugerido pela proximidade dos elementos é como se "Esquerda" e "Topo" fossem duas opções relacionadas, o mesmo com "Direita" e "Base" e as duas opções "Centralizado". Além disso, a organização dos elementos não sugere uma seqüência lógica. Na figura abaixo alguns destes elementos são corrigidos:




Agora uma linha separa as duas partes, facilitando a identificação dos relacionamentos (por fechamento). Além disso, a ordem das opções foi trocada, inserindo a opção centralizado

no centro, como é mais lógico. Entretanto, a organização horizontal das opções "Topo", "Centralizado" e "Base" ainda não contribuem para a lógica de seu funcionamento.

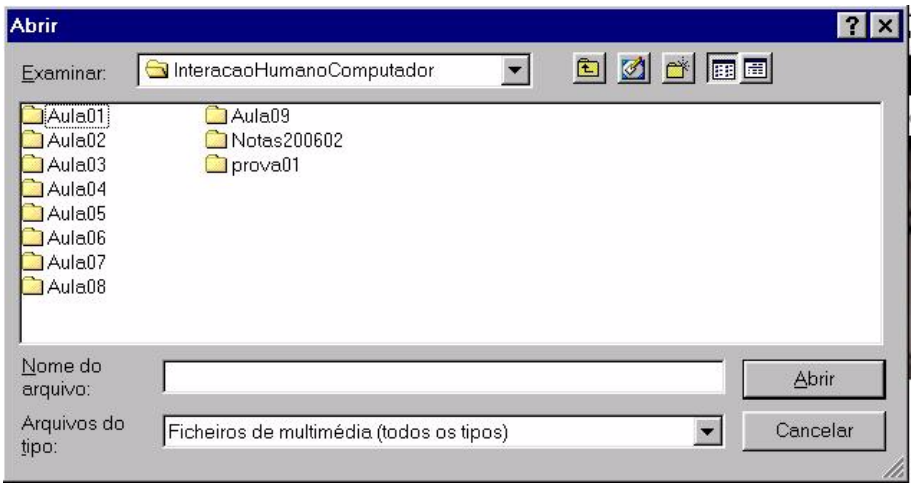
Abaixo é apresentada uma jenela que, embora não seja tão "limpa" quanto a anterior, apresenta uma organização que rapidamente possibilita a compreensão por parte do usuário:


Observe que o uso incorreto destas regras pode trazer conseqüências ruins. Se for digitado DIR /W num diretório, por exemplo, temos a seguinte apresentação de informações:

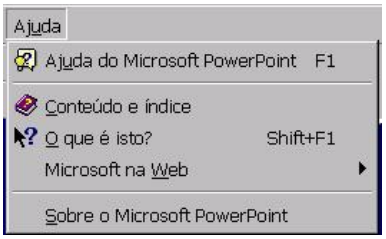
Volume in drive Z has no label.
Directory of Z:\windows\system

```
[.]             [..]            AVICAP.DLL      AVIFILE.DLL     COMMDLG.DLL
dlsnet.dll      Dlstb.dll       KEYBOARD.DRV    LZEXPAND.DLL    MCIAVI.DRV
MCISEQ.DRV      MCIWAVE.DRV     mmsystem.dll    MMTASK.TSK      MOUSE.DRV
MSVIDEO.DLL     myocr.dll       ocrutil.dll     OLECLI.DLL      OLESVR.DLL
segment.dll     segsdk.dll      setup.inf       SHELL.DLL       SOUND.DRV
stdole.tlb      SYSTEM.DRV      TAPI.DLL        TIMER.DRV       VER.DLL
VGA.DRV         WFWNET.DRV      windls.dll      winspool.drv
       34 file(s)    1710151 bytes used
                    5608808 K bytes free
```

Esta organização sugere que a seqüência da organização é nas colunas, quando na realidade é nas linhas. Por esta razão, procurar o nome de um arquivo no resultado de um comando DIR pode ser algo extremamente aborrecedor. Observe que em aplicativos mais recentes (como a caixa de abrir um arquivo do Windows) a ordenação dos nomes de arquivos é nas colunas, como pode ser observado abaixo:



Estes mesmos princípios podem ser aplicados a menus. Observe como em vários aplicativos são usadas pequenas barras nos menus para separar categorias de funções. Isso é um uso de fechamento e organização lógica de informações.




## 3. Codificação Visual

A codificação visual é o **ato de diferenciar tipos de objetos distintos** em uma aplicação, usando-se para isso de **técnicas de codificação**. As técnicas mais comuns são: **cor, forma, tamanho, aspecto, intensidade, textura e espessura/estilo de linha**.

Cada técnica de codificação tem um número diferente de categorias que ela pode diferenciar, lembrando que o mau uso da codificação pode causar confusão e erro. O ser humano médio consegue distinguir a seguinte quantidade de categorias sem dificuldade (Van Cott e Kinkade, 1972 apud Foley *et al*. 1990):

**- 10 cores**
**- 6 tamanhos (área ou comprimento)**
**- 4 intensidades**
**- 24 ângulos**
**- 15 formas geométricas**

Algumas codificações devem, quase sempre, ser **usadas com redundância** (dois códigos ao mesmo tempo). Um exemplo deste tipo de codificação é a cor que, se usada isoladamente, pode confundir muito os indivíduos daltônicos. Entretanto, **não se deve usar mais que duas ou três técnicas simultaneamente**, para evitar confusão.

A **seleção** do tipo de código mais adequado leva em conta a **quantidade de categorias** de discriminação necessárias, bem como os **tipos de informação: nominativas** (sem distinção de ordem), **oridnárias** (com ordme, mas sem relação métrica) e **proporcionais** (com ordem e métrica comparativa).

**Informações nominativas**: não é bom usar códigos que indiquem ordem ou relação métrica. Assim, deve-se usar **formas, estilos de linhas, texturas**, etc.

**Informações ordinárias**: deve-se usar sugestão de ordem, mas não de relação métrica. Assim, deve-se usar **texturas com densidade variável, tamanhos de texto, linhas de densidade variável**, etc.

**Informações proporcionais**: tanto o sentido de ordem quanto de proporção métrica deve ser representado. Deve-se usar **variações de tamanho, extensão, orientação, etc**.

É **importante** lembrar sempre de **verificar regras e similaridade e agrupamentos lógicos** no uso de técnicas de codificação visual, lembrando também os **cuidados com os mecanismos de atenção** (cores e formas especiais para chamar atenção, cursor rotativo ou piscante, vídeo reverso, etc).

## 4. Consistência Visual

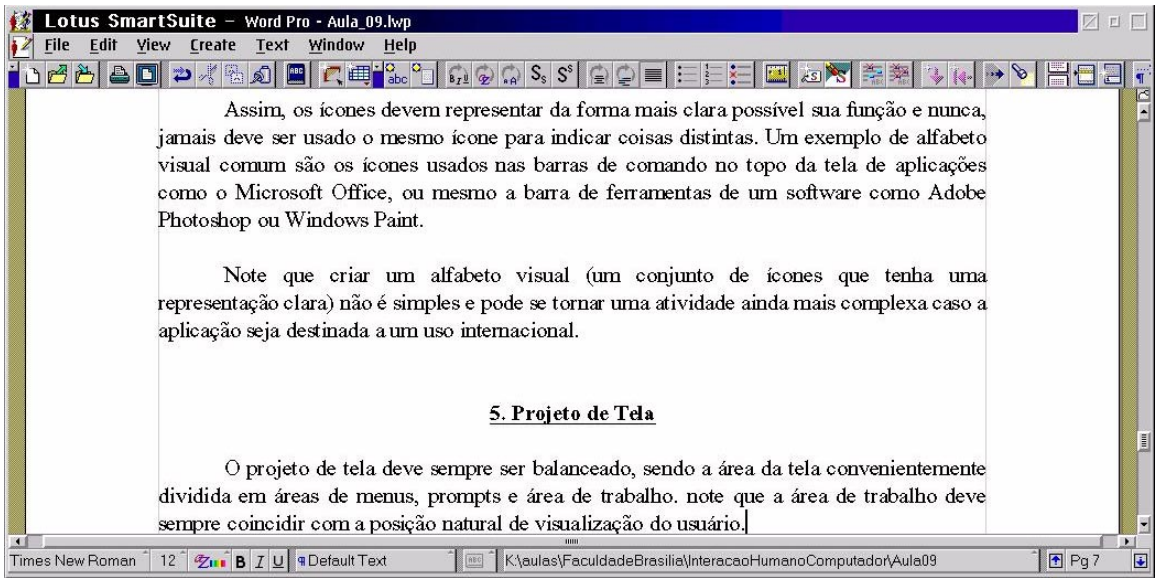
Assim como na interface de ação, no projeto visual a consistência está **relacionada ao uso consistente de elementos como imagens, códigos e ícones** ao longo da aplicação, ou seja, **usando sempre o mesmo alfabeto visual**.

Assim, os **ícones devem representar da forma mais clara possível sua função** e nunca, **jamais deve ser usado o mesmo ícone para indicar coisas distintas**. Um exemplo de alfabeto visual comum são os ícones usados nas barras de comando no topo da tela de aplicações como o Microsoft Office, ou mesmo a barra de ferramentas de um software como Adobe Photoshop ou Windows Paint.

Note que **criar um alfabeto visual** (um conjunto de ícones que tenha uma representação clara) **não é simples** e pode se tornar uma atividade ainda mais complexa caso a aplicação seja destinada a um uso internacional.

## 5. Projeto de Tela

**O projeto de tela deve sempre ser balanceado**, sendo a área da tela convenientemente dividida em **áreas de menus, prompts e área de trabalho**. Note que a **área de trabalho deve sempre coincidir com a posição natural de visualização** do usuário. Observe a figura abaixo:



Nesta figura, é possível observar a área de menu no topo, a área de prompt na região inferior, juntamente com uma segunda (e menor) área de menus. A área de trabalho fica centralizada, tornando-se mais agradável ao uso.

## 4.Bibliografia

FERREIRA, M.A.G.V. *Notas de Aula - Tópicos de Comunicação Homem-Máquina*, EPUSP, 2003.

FOLEY, J. D. et al. *Computer Graphics Principles and Practices*. Addison-Wesley, 1990.

SHNEIDERMAN, B. *Designing the User Interface*. 3rd. Ed. Addison-Wesley. Reading, Ma. 1998.

Bibliografia Complementar:

NETTO,A.A.O.Modelagem e Gerência de Interfaces com o Usuário. Visual Books, 2004.

Notas da Aula 10: Introdução às WUIs
Prof. Daniel Caetano

**Objetivo**: Apresentar Diferenças entre WUIs e GUIs e algumas dicas iniciais de projeto.

## 1. Introdução

Desde o início do curso, muitos tópicos sobre interfaces foram abordados. Até agora, porém, foram apresentados conceitos em uma forma genérica e, em alguns casos, focados no desenvolvimento de GUIs (Graphical User Interfaces).

A partir de agora, o foco será maior nas WUIs (Web User Interfaces), diferenças na concepção e projeto das mesmas, passando pelas diferenças inerentes a seus usuários.

## 2. Paradigma WUI

As **Web User Interfaces são**, como apresentado nas primeiras aulas, interfaces **voltadas a facilitar a obtenção de informações** por parte do usuário, ao consultar um sistema qualquer. É importante ressaltar esta característica, em oposição ao objetivo de **facilitar o processamento de informações que é característico das GUIs**.

O **usuário** também é **diferente no paradigma WUI**. Enquanto **nas GUIs o usuário é mais ou menos conhecido** (embora não completamente), **nas WUIs** os usuários **são completos desconhecidos**. Em outras palavras, isso significa que no projeto das GUIs é possível determinar com alguma precisão o que os usuários esperam do software, mas no caso das WUIs essa determinação é quase impossível.

Além disso, existe uma outra questão envolvida no caso do usuário das WUIs. Enquanto as **GUIs normalmente são parte de softwares caros e que precisam ser adquiridos *à priori***, o usuário acaba ficando cativo de um software por já ter gasto dinheiro no mesmo. Em outras palavras, nas GUIs o usuário só conhece a interface depois que comprou o produto. **No caso das WUIs** a relação se inverte: **a interface é usada para adquirir produtos**. Desta forma, o usuário não se fixa em um determinado site e, se não gostarem de sua interface, além de não adquirirem qualquer produto, com grande probabilidade jamais voltarão àquele site.

Uma quarta consideração é quanto à complexidade da navegação. Enquanto nas **GUIs a navegação é bem simples** (quando não inexistente), sendo basicamente direcionada por um menu suspenso, **nas WUIs a navegação é bastante complexa**, com elementos de navegabilidade em todos os cantos e sendo possível chegar a uma determinada página por dezenas de trajetos.

Como é possível observar, há **grandes diferenças** entre o "problema" de se projetar uma GUI ou uma WUI. É claro que **conceitos de ergonomia** (como mínima carga de memória, máxima realização, etc) **continuam válidos**. O **mesmo vale para os conceitos da interface de percepção**, como elementos de atenção, tempos de feedback e até mesmo projeto visual. Entretanto, as diferenças de funcionamento e dos usuários de cada uma das duas levam a ênfases em aspectos distintos.

As principais **diferenças entre GUIs e WUIs** podem ser **resumidas** da seguinte forma:

**GUIs:**
**1) Centradas em tarefas.**
**2) Anseios do usuário razoavelmente conhecidos.**
**3) Usuário "cativo".**
**4) Navegação simples.**

**WUIs:**
**1) Centradas na informação.**
**2) Usuários e seus anseios são desconhecidos.**
**3) Usuário "nômade".**
**4) Navegação complexa.**

## 3. Evolução da Web

Antes de sentar e projetar um website, entretanto, é interessante avaliar as **tecnologias** que estão **disponíveis e** quais serão **usuadas**. É possível classificar as evoluções da Web e suas tecnologias em **fases**:

**Web Estática**: Primeira fase da web, em que os **conteúdos eram estáticos**, ou seja, eram criados todos manualmente e quase nunca atualizados. Esta fase passou por dois estágios: um primeiro onde as formatações visuais eram mínimas e uma segunda, em que se introduziram tabelas, mudanças de cores e fontes, etc.

**Web Dinâmica**: Segunda fase da web, em que os **conteúdos** passaram a ser dinâmicos, ou seja, a **se modificarem sozinhos** com o tempo. As tecnologias acrescentadas, como forms, uso de linguagens script (Javascript, PHP, ASP, etc) em conjunto com bancos de dados tornaram possível a **criação de páginas personalizadas** e com características muito mais atrativas aos usuários.

**Web Ativa**: Fase atual, em que foram acrescentados **elementos que são executados no computador do usuário**, com informações sendo apresentadas e escolhidas sem a intervenção do servidor. Fazem parte desta era tecnologias como Java e Flash.

É importante lembrar que usar recursos de **Web Dinâmica** implicam em **exigências específicas com relação ao servidor** onde a página estará hospedada (como a existência de

um interpretador PHP de uma determinada versão, por exemplo). Uma vez que esta exigência seja resolvida, todos os usuários terão acesso ao conteúdo.

No caso da **Web Ativa**, entretanto, a **exigência é com relação à existência de um determinado produto** (Flash, por exemplo) de uma determinada versão **no micro do cliente**. A vantagem quando os requisitos são atendidos é que o usuário tem uma experiência muito mais rica e rápida. A desvantagem é que o usuário pode não querer instalar o software em sua máquina (ou mesmo não poder instalá-lo, por uma série de razões) e seu site acabar ficando inacessível para estes usuários.

A dica aqui é sempre **usar o menor número de recursos necessário** para que o website seja atrativo e, sempre que for usado algum recurso extra de Web Dinâmica ou Ativa, procurar requisitar sempre a versão mais antiga possível de softwares a serem instalador no servidor ou computador do cliente, dado que a probabilidade de uma versão antiga ou mais nova já estar instalada é maior a cada versão do produto que entra no mercado.

## 4. Dicas de Projeto

É possível oferecer algumas dicas para o projeto de WUIs. Embora elas não solucionem automaticamente a difícil tarefa de projetar um web site, elas ajudam no processo.

Primeiramente, deve-se **definir o objetivo do site**, ou seja, sua missão: que tipo de informação eu desejo passar, que tipo de produto eu desejo vender. Esta é uma característica muito importante porque influencia não só no conteudo que deve ou não estar presente, mas também na organização do site.

Uma segunda preocupação é, de alguma forma, **estimar os prováveis usuários do site**, criando alguns modelos de tipos de usuário padrão. Esta determinação é importante para que as diversas partes do site seja planejadas levando em conta estes diferentes tipos de usuário.

Um terceira determinação, derivada da segunda, é uma **determinação das necessidades de cada usuário**, diante de seu site. Ou seja, que tarefas e informações ele provavelmente deseja encontrar, quais são os interesses deste usuário, objetivando atendê-los da melhor forma possível.

A partir destes dados, deve-se projetar o web site. Este **projeto pode ser dividido em** duas partes: o **projeto de página** e o **projeto de site**. Ambos devem **seguir uma organização lógica da informação** a ser apresentada (lógica segundo o ponto de vista do usuário, não segundo o ponto de vista do fabricante do produto apresentado no site) e **apresentados na forma de esquemas**, incluindo **conteúdos e elementos de apresentação**, construindo um **mapa do site**. A hierarquia de páginas será uma conseqüência dos conteúdos, que muitas vezes precisam de várias páginas para serem apresentados em um formato correto.

É sempre importante que este **mapa** desenvolvido esteja, de alguma forma, **disponível no site**, facilitando aos usuários encontrarem alguma informação. O mesmo vale para uma **função de busca**. Vale ressaltar que o ideal é que o usuário não precise usar tais ferramentas, mas se ele sentir a necessidade das mesmas, é bom que elas estejam lá, caso contrário ele pode sair de seu site frustrado e nunca mais voltar.

No caso específico da função de busca, é interessante que ela seja feita de forma a permitir **buscas locais** (em uma dada seção do site) **ou globais** (no site como um todo), com esta seleção sendo explícita para o usuário, de forma a ajudá-lo a encontrar o que procura. Quando nenhuma informação for encontrada em uma busca local, o site pode oferecer uma busca por todo o site automaticamente, facilitando as tarefas do usuário.

Finalmente, depois de implementado o sistema projetado, deve-se realizar inúmeros **testes com diversos tipos de usuários**, para eliminar quaisquer problemas que tenham passado pela etapa de projeto.

Uma proposta de desenvolvimento de interface é a seguinte, conhecida como "Ciclo Iterativo de Engenharia da Usabilidade":

**Fase 1: Projeto**
**Entradas: Padrões, Diretivas, Experiências, Requisitos, Avaliações Anteriores.**
**Saídas: Modelos de Avaliação e Especificação.**

**Fase 2: Implementação**
**Entradas: Especificação.**
**Saída: Protótipo ou Sistema completo.**

**Fase 3: Avaliação**
**Entradas: Padrões, Requisitos, Modelos de Avaliação, Protótipo/Sistema.**
**Saída: Sistema pronto ou Avaliação de Implementação (com retorno à fase 1)**

## 5. O Que Não se Deve Fazer em Um Projeto de Website

Historicamente, grande parte dos websites são horríveis em diversos níveis. Em parte isso veio do fato que, desde o Geocities, **qualquer um pode projetar um website** e colocá-lo no ar. Desta forma, muitas pessoas que jamais estudaram qualquer coisa sobre comunicação visual e interfaces colocam online páginas com usabilidade inimaginavelmente ruim.

Entretanto, mesmo **grandes empresas constroem muitas das piores páginas** de que se tem notícia na Web. As razões para que mesmo equipes profissionais construam websites terríveis são muitas. A seguir serão citadas algumas, como exemplo de **direcionamentos** comuns **que não devem ser seguidos**:

**1) Projetar um website como uma brochura de marketing** - a web tem suas próprias diretrizes de design;

**2) Projetar um website com foco nos anseios internos na empresa** - ele deve ter foco nas necessidades do cliente;

**3) Estruturar o site de acordo com a estruturação interna da empresa** - a estruturação deve seguir a lógica dos documentos e de busca do usuário;

**4) Criar páginas pesadas, enormes e cheias de animação, ainda que elas fiquem rápidas na intranet da empresa** - as páginas devem ser rápidas nos computadores dos clientes;

**5) Escrever conteúdo em estilo linear, como revistas e livros** - usar muito hipertexto;

**6) Tratar seu site como o único do mundo, sem links externos** - usuários web não gostam de se sentir presos!

## 6. Bibliografia

NIELSEN, J. *Projetando Websites*. Ed. Campus, 2000.

FERREIRA, M.A.G.V. *Notas de Aula - Tópicos de Comunicação Homem-Máquina*, EPUSP, 2003.

NERURKAR, U. *Web User Interface Design: Forgotten Lessons.* IEEE Software. Nov/Dec 2001, p. 69-71.

Notas da Aula 11: Projeto de Sites Web I
Prof. Daniel Caetano

**Objetivo**: Apresentar características de projeto Web e dicas na elaboração de sites.

## Introdução

Como visto na aula passada, embora muitos dos conceitos apresentados sirvam para GUIs e WUIs, estas últimas requerem considerações adicionais devido ao seu foco: a informação.

Existe ainda uma dificuldade adicional, devido ao fato que o desenho de páginas Web é, hoje, impregnado por maus hábitos dos projetistas, que iniciaram na tarefa de criar páginas Web muito antes de qualquer informação ser estudada ou conhecida a respeito de usabilidade em Web.

Algumas aula serão voltadas, então, a **auxiliar no projeto de páginas Web**, bem como apresentar **sugestões no desenho do site e das páginas**, evidenciando más práticas que devem ser evitadas.

**Não serão apresentados** aspectos específicos de **aplicações web**, já que o projeto destas segue basicamente os mesmos princípios das GUIs, por serem voltadas às tarefas a serem executadas e não à transmissão de informação

## 1. O Projeto do Site

Em primeiro lugar, é preciso deixar claro que a **Engenharia para a Web** não é um clone perfeito da Engenharia de Software, embora procure seguir os mesmos princípios. Neste espírito, as **características** mais importantes (mas não únicas) que norteiam um projeto são:

- **Confiabilidade**
- **Usabilidade**
- **Adaptabilidade**

**Busca-se** um **desenvolvimento bem sucedido**, bem como uma **implementação e manutenção simplificados**, ao mesmo tempo em que se obtém um resultado de **alta qualidade sob o ponto de vista do usuário**.

A **ponto de vista de Lowe** (1999) facilita a compreensão da maneira como devemos enxergar o desenvolvimento de um web site: "O desenvolvimento de Websites está muito mais relacionado à criação de uma infraestrutura (as linhas mestras do jardim) e depois ao "cultivo" da informação que cresce e floresce dentro deste jardim. Ao longo do tempo, o

jardim (ou seja, o site na Web) vai continuar a evoluir, a se modificar e a crescer. Uma boa arquitetura inicial deve permitir que este crescimento ocorra de forma controlada e consistente."

Existem algumas outras **diretivas** que impulsionam o processo de projeto de um WebSite:

- **Imediatismo**, uma necessidade de que o site em poucos dias ou semanas no ar.
- **Segurança**, para limitar o acesso a algumas informações para alguns tipos de usuários.
- **Estética**, fundamental para vender produtos ou idéias, fundamental para o sucesso.

Segundo Pressman (2002), os principais **atributos de qualidade** de um site são:

- **Usabilidade**, relativa à compreensão geral do site, ajuda online, interface e estética.
- **Funcionalidade**, relativo à buscas, navegação e características da aplicação.
- **Confiabilidade**, relativo a links corretos, recuperação de erros, validação de entradas de usuário.
- **Eficiência**, relativo ao tempo de resposta, tempo de renderização da página e gráficos.
- **Manutibilidade**, relativo à facilidade de correção, adaptabilidade e extensibilidade.

## 1.1. Diferenças no Ciclo de Projeto de Sites

Em geral, o **ciclo de projeto de um Site é** exatamente **o mesmo de qualquer aplicação**, com **planejamento, análise, engenharia, testes, avaliação** e assim por diante. Da mesma forma que em projetos de software usual temos uma equipe cuidado das funções básicas do aplicativo e uma outra equipe cuidando da interface com o usuário, **no projeto Web também temos duas frentes**.

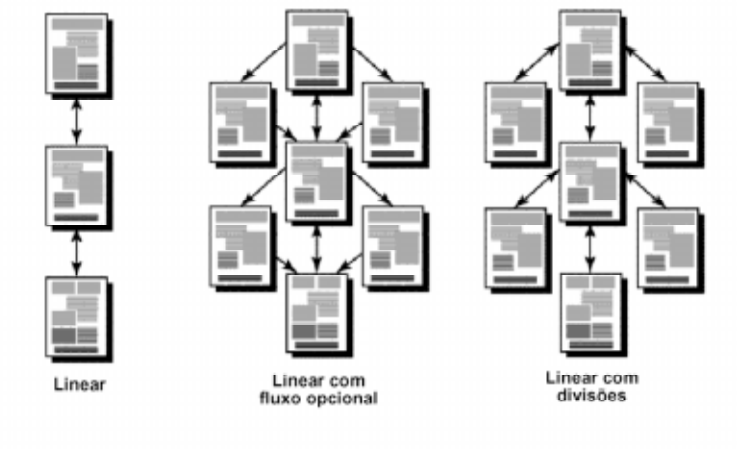
A **primeira** destas frentes cuida da **arquitetura** do site, o **projeto da navegação** e da **interface**. A **segunda**, deve se preocupar com o **projeto de conteúdo** e a **produção**.

## 2. Dicas de Projeto do Site como um Todo

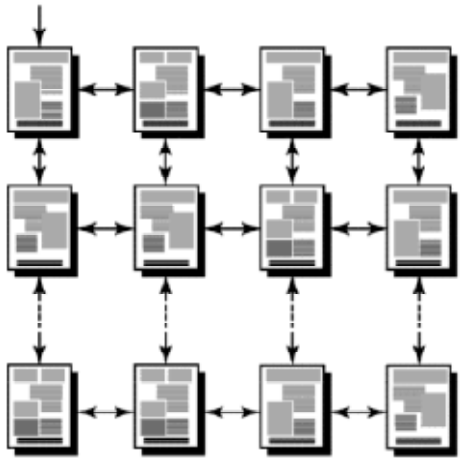
ESTRUTURA DO SITE

Talvez a primeira coisa a ser definida, deve ser a estrutura do site. De forma geral, esta estrutura é baseada no **conteúdo** a ser apresentado, nas **classes de usuários** que devem acessar a página e na **filosofia de navegação** proposta. As **filosofias** de navegação podem ser: **lineares, em malha, hierárquicas, em teia, dentre outras** mais caóticas.

Estruturas **lineares** são de **fácil compreensão**, porém **pouco flexíveis**.

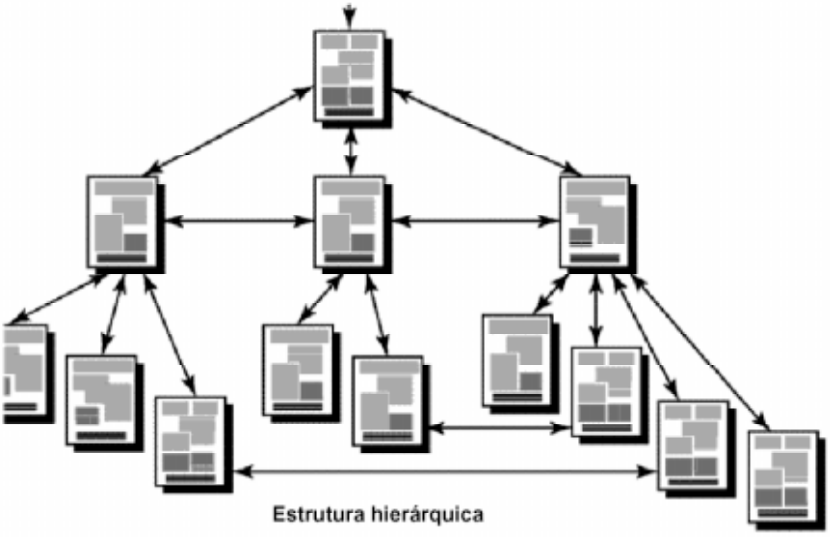



Estruturas **em malha** são de **de compreensão um pouco complexa**, porém **mais flexíveis**.



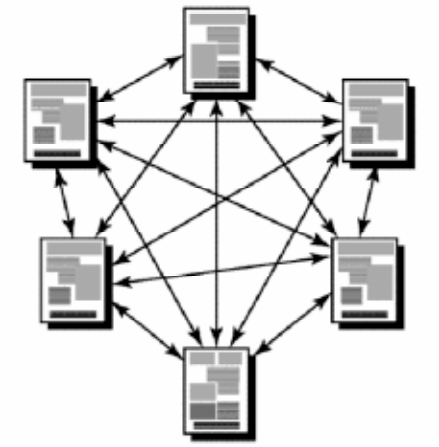
Estrutura em malha com categorias bidimensionais

Estruturas **hierárquicas** são de **de compreensão complexa**, porém são **flexíveis**.



Estrutura hierárquica

Estruturas **em teia** são de **de compreensão muito complexa**, porém são **totalmente flexíveis**.



Estrutura em Teia (Web)

**- A estrutura escolhida deve ser o menos confusa** possível, além de **refletir como o usuário enxerga o conteúdo** do site e não a forma como a empresa é dividida.

- **Não use telas Splash** (aquelas telas de abertura), a menos que necessário. Exemplos destas situações são mudanças do endereço principal, páginas com conteúdos inapropriados para menores, etc.

- **Não force o usuário a entrar pela homepage**. Os links das páginas internas devem estar **sempre** acessíveis e, dentro do possível, não devem ser alterados.

DESENHO GLOBAL

- **A largura da página deve ser dinâmica**, sempre que possível. Se não puder, ela deve ser corretamente apresentada mesmo em baixas resoluções, ou seja, para boa **visualização em torno de 600 pixels**.

- **Cuidado com o uso de metáforas** no projeto visual geral! Metáforas equivocadas podem ser "bonitinhas" mas dificultarem o uso por parte do usuário. Lembre-se, por exemplo: Web não é TV! Metáforas consagradas (como a do carrinho de compras, por exemplo) devem ser usadas, entretanto.

- **Região de navegação clara**, indicando "onde o usuário está", "onde estava" e "onde pode ir", fazendo indicações com relação à Web e ao Site como um todo.

DESENHO DA HOMEPAGE

A homepage é a **página principal**, onde o usuário entra se digitar o link principal de sua página.

- **Deve ter design diferenciado**, visando captar a atenção do usuário.

- **Não deve ter botão "home"**, mas se tiver, deve ficar "desligado".

- **Deve ter um logotipo maior**, respondendo à pergunta "onde estou" que o usuário possa se fazer ao chegar em seu site.

- **Deve passar a idéia do que o site faz**, mas nunca através das enfadonhas "missões" que são apresentadas em alguns sites.

- **Área de notícias restrita**, a menos que seu site seja um site de notícias. A maioria dos usuários entra em uma página buscando uma informação específica, não as últimas novidades que ocorreram em sua empresa.

- **Deve ter um mecanismo de navegação por menus**, se possível indicando também um mapa do site. Muitos usuários preferem a navegação link-a-link ou por mapa de site.

- **Deve ter um mecanismo de busca**, incluindo busca avançada. Nem todo usuário gosta de navegação link-a-link e sempre vai direto no mecanismo de busca.

DESENHO DAS PÁGINAS INTERIORES

Diferentemente da homepage, as páginas interiores são direcionadas à **apresentação de conteúdo**.

- **O logotipo deve ser menos proeminente**, já que aqui ele é apenas uma informação adicional e não o foco da página interior.

- **O logotipo deve linkar para a homepage**, para que o usuário que entrou "pelo meio" da página, por um mecanismo de busca, possa descobrir mais sobre a empresa que criou aquela página.

- **Deve mostrar conteúdo específico e direto**, não ficar aborrecendo o usuário com mensagens de boas vindas.

DESENHO DE SUBSITES

Subsites são uma **forma interessante de categorizar informações** de maneira que apenas usuários realmente interessados as acessem. Entretanto, alguns cuidados adicionais devem ser ressaltados.

- **Podem ter diferenças do site principal**, mas é bom evitar diferenças demais.

- **Devem ter coerência visual com o site principal**, para manter a identidade do site como um todo.

- **Devem ter navegação similar**, para não frustrar o usuário por ter que aprender a navegar em mais um site diferente.

- **As buscas devem oferecer opções de escopo**, deixando claro quando uma busca é local (no subsite) ou global (no site todo).

DESENHO DE URLS

A URL é a parte que é mais divulgada de seu site, de forma que as **pessoas possam acessá-lo**. Por esta razão, vale tomar alguns cuidados:

- **Escolha nomes claros**, sem caracteres malucos.

- **Escolha nomes breves**, já que nomes grandes são mais difíceis de guardar e mais propensos a erros de digitação

- **Não use mistura de caixa alta e baixa**. Dentro do possível, use todo o nome em letras minúsculas e não use, em hipótese alguma, acentuação.

- **Use endereços estáveis**. Esta dica é importante para sites com conteúdo que mudam em determinados períodos de tempo. Apesar de ser "bonitinho" ter um link como "edicaoatual.html", isso é péssimo para quem quer linkar um artigo. A pessoa faz um bookmark para um artigo no mês de agosto e ao tentar reler o mesmo artigo no mês de setembro ele sumiu.

- **Produtos devem indicar a URL de sua página web**. Embora seja comum as empresas colocar o site da empresa em seus produtos, é tão ou mais importante indicar a URL da página do produto na embalagem/manual. Poucas coisas irritam mais um usuário do que ter que "achar" a página do produto num site, quando precisa de drivers ou suporte.

## 6. Bibliografia

NIELSEN, J. *Projetando Websites*. Ed. Campus, 2000.

PRESSMAN, R. Engenharia de Software. Ed. McGraw-Hill, 2002.

FERREIRA, M.A.G.V. *Notas de Aula - Tópicos de Comunicação Homem-Máquina*, EPUSP, 2003.

Notas da Aula 12: Projeto de Sites Web II
Prof. Daniel Caetano

**Objetivo**: Apresentar características e dicas na elaboração de páginas Web.

## Introdução

Como visto em aulas anteriores, o projeto de interfacs é sempre voltado às necessidades do usuário. No caso da Web, o usuário é, muitas vezes, desconhecido. Entretanto, uma característica marcante é conhecida: o usuário de Web busca informações.

Sendo assim, o **projeto das páginas e de seu conteúdo** deve ser **focado na facilidade de compreensão e da navegabilidade**, sendo a **linguagem visual** utilizada **muito importante** para uma boa aceitação da mesma.

Em última análise, **essas são características que** praticamente **definem o sucesso ou fracasso de** uma determinada **página ou site**. Por esta razão, continuando as dicas da aula anterior, serão apresentadas dicas relativas ao desenho da página, em termos de diretrizes gerais de desenho) e do conteúdo como um todo.

Lembre-se que **credibilidade é tudo** e, portanto, se a página a ser criada for séria, nada de firulas como ícones animados para mandar e-mail e coisas do tipo, se quiser que sua página tenha sucesso.

## 1. Dicas para o Projeto da Página

O **projeto da página** envolve basicamente **como os elementos da página estão dispostos** e **a finalidade de cada uma das áreas**.

ÁREA DA TELA E LAYOUT

- **Deixe a maior parte da tela para conteúdo**. Lembre-se que o usuário não entra em uma página por causa da beleza do logotipo, nem mesmo pelo incrível mecanismo que foi criado para os menus... Muito menos para ver propagandas.

- **Não abarrotar a tela com informações**, em especial se elas forem inúteis. Se há muitas informações a serem apresentadas, segmente-as em conjuntos coerentes e apresente-as de maneira separada. Abarrotar a tela com informações só fará com que o usuário perca interesse no conteúdo como um todo.

- **Evite gastar área da tela com propaganda**. Embora nem sempre seja possível, o ideal é nem existir propaganda alguma. Se for decidido que haverá propagandas, nunca coloque mais que uma, pois elas disputarão atenção (e espaço) entre si e contra o conteúdo da

página, irritando o usuário. Se for usá-las, procure colocar propagandas que tenham alguma correlação com o conteúdo da página.

- **O layout deve ser independente de resolução**, ou seja, deve ser redimensionável, como já dito anteriormente. Caso não seja possível, deve ser fixo em um tamanho adequado para visualização no máximo de dispositivos.

- **O layout deve ficar bom em diferentes navegadores e plataformas**, por isso, realize testes em todas elas! Não há nada mais desagradável do que um usuário visualizar uma página toda distorcida porque a equipe que a projetou só testou em um único navegador. Página distorcida ou não funcional significa usuário perdido... e usuário perdido significa negócios não concretizados!

- **Lembre-se dos softwares antigos**. Muitos usuários ainda usam versões antigas de navegadores e extensões. Procure desenvolver seu site para duas versões mais antigas que a recente. Se fizer para a "última moda" em navegador ou plugin, lembre-se de criar uma versão que funcione bem com versões antigas dos mesmos.

- **Separe programação de layout do conteúdo**. Lembre-se: conteúdo é especificado no HTML, que indica o significado de cada elemento. O layout é definido nos arquivos de CSS (Cascade Style Sheets), ou seja, separados do conteúdo. Isso facilita a manutenção do site, tanto na alteração de conteúdo quanto na criação de novos layouts.

- **Não use frames**. Embora em alguns raros casos o uso de frames seja aceitável (e talvez até mesmo requeridos), os frames criam problemas sérios de navegação, dificuldades para linkar partes específicas do site, além de terem uma visualização muito dificultada em dispositivos de navegação com baixa resolução.

- **Lembre-se da impressão**, ou seja, que os usuários podem querer imprimir sua página. Se sua página tiver fundo escuro com texto claro, crie uma versão alternativa de CSS para impressão, com fundo claro e texto escuro, sem figuras de fundo, de preferência. Se não quiser, adicione links para a matéria em formato postscript (.PS) ou portable data file (.PDF).

TEMPOS E TAMANHOS

- **Os tempos de resposta devem ser mínimos**, já que ninguém gosta de ficar esperando olhando para uma ampulheta.
  - **0,1s é o ideal**, para que o usuário sinta uma navegação em tempo real.
  - **1s é razoável** para que não atrapalhe o fluxo de pensamento do usuário.
  - **10s é o limite** de usabilidade.

- **Pense nos usuários de modem**. Estes tempos não são relativos ao seu próprio micro ou à uma conexão banda larga. Desprezar usuários de modem é desprezar uma grande parcela da população.

- **Cuidado com figuras e textos muito grandes**, pois eles aumentam **muito** o tempo de download. Os tamanhos ideais são:

| **CONEXÃO** | **IDEAL** | **MÁXIMO** |
|---|---|---|
| **- Modem:** | **até 2KB** | **34KB.** |
| **- ADSL até 1Mbps:** | **até 8KB** | **150KB** |
| **- ADSL rápidas:** | **até 100KB** | **2MB.** |

USO DE LINKS

- **Use links de navegação estrutural ao longo do texto**, que são aqueles que interligam partes de uma única página. Eles facilitam ao usuário encontrar o que procuram e ajudam a manter os textos enxutos, já que explicações detalhadas de alguns aspectos podem ter suas próprias páginas.

- **Use links associativos ao longo do text**o, que ligam partes de sua página com páginas que expliquem termos ou assuntos citados. Não incorra no erro de querer prender o usuário em sua página. Não é assim que se conquista usuários de internet.

- **Use links de referência adicional ao fim do texto**, ligando sua página a páginas que falem de assuntos semelhantes e que possam ser de interesse de pessoas que tenham lido sua página do começo ao fim.

- **Use palavras significativas nos links**, não use palavras como "clique aqui" ou algo assim. Isso dificulta o uso por deficientes visuais e dificulta a geração de um menu do tipo "Links" onde o usuário possa navegar sem precisar ler a página.

- **Use títulos nos links**, dentro do possível, explicando de forma resumida para onde os links levam e qual o conteúdo da página indicada.

- **Evite mudar as cores padrão dos links**, já que os usuários geralmente estão acostumadas a elas. Se mudá-las, seja coerente.

- **Use sempre a mesma URL na indicação de links para uma mesma página**, de forma que se ela já houver sido visitada (mesmo que através de outro link), o link fique em cor diferente.

- **Diferencie links externos**, se possível, indicando-os com um estilo diferente ou alguma marca (por exemplo, escrevendo em itálico).

## 2. Dicas para o Projeto do Conteúdo

TEXTO

O texto é, em geral, a **parte mais importante** de um site. Entretanto, a **leitura na tela tem seus problemas** e medidas especiais devem ser adotadas para tornar a experiência do usuário mais agradável.

**- Seja sucinto**, é a principal dica com relação ao texto. A leitura na tela é cerca de 25% mais lenta que no papel, devido à maior dificuldade de leitura oriunda do brilho e a baixa resolução da tela. Por esta razão, textos longos são cansativos e desagradáveis. Na web os textos devem ter em torno de 50% do tamanho que eles teriam em uma publicação impressa.

- **Use e abuse do hipertexto**. Isso significa que se uma dada palavra do seu texto merece um "artigo" para explicá-la, transforme-a em um link para uma página onde exista a explicação para a mesma. Lembre-se: web não é livro. O uso de hipertexto facilita a redução do tamanho dos textos, já que os termos não serão explicados detalhadamente dentro do texto principal, como ocorreria na mídia impressa.

- **Divida o conteúdo em trechos**, apresentados em páginas diferentes. Observe que isso deve ser feito com critério e não simplesmente cortando um texto no meio, em qualquer parte, apenas para dividir em mais de uma página.

- **Evite erros ortográficos**, em especial em páginas de empresa. Embora em páginas pessoais isso não seja grave, há poucas coisas que tiram mais credibilidade de uma empresa que uma página web cheia de erros ortográficos.

- **Facilite a leitura...** os usuários raramente lêm uma página toda. O uso de elementos especiais ajudam ao usuário focar naquilo que é mais importante. Ou seja: destaque o que é mais importante.

**- Estruture o texto em 2 ou 3 níveis de títulos.**
- **Use títulos significativos**, ao invés de títulos "bonitinhos".
- **Quebre blocos de texto em trechos menores**. Bullets ajudam nisso.
- **Enfatize palavras importantes**, usando negrito, mudanças de cor, etc.

- **Use linguagem simples**, uma vez que o usuário pode não ser um especialista no assunto sendo tratado. Como em qualquer interface, evite ao máximo o uso de termos computacionais.

- **Cuidado com o uso de humor**, alguns usuários podem não compreender ou se ofender com as brincadeiras e trocadilhos.

TÍTULOS

- **Seja descritivo e sucinto**, usando de 2 a 6 palavras. Lembre-se que os títulos são, muitas vezes, apresentados fora de contexto, em bookmarks ou em sites de busca.

- **Use linguagem simples**, sem trocadilhos que possam dificultar usuários de outras nacionalidades.

- **Evite atrair usuários que não são público alvo**, ou seja, evite títulos que possam ter uma interpretação dúbia e trazer usuários que não se interessariam por sua página. Tais usuários certamente se decepcionarão com o conteúdo encontrado.

- **Pule artigos definidos e indefinidos iniciais** do título, lembrando que muitos sites de diretório e busca organizam os links para os sites pelo título e você não gostaria que o seu site estivesse misturado em meio a um monte de sites começando com "O", "A", "Um" ou "Uma", por exemplo.

- **Use uma primeira palavra bem descritiva**, já que, como dito acima, muitos sites de diretório e busca organizam os links pelo título dos sites. Usando uma primeira palavra indicativa, você facilitará ao usuário encontrar seu site.

- **Não use o mesmo título para todas as páginas** do site, e evite que todas as páginas comecem com título igual. Lembre-se que os sites de busca apresentam o **título** da página como representação do seu conteúdo.

LEGIBILIDADE

- **Use cores de alto contraste**, ou seja, se usar fundo claro, use texto escuro. Se usar fundo escuro, use texto claro.

- **Evite figuras de fundo**, mas se usá-las, use figuras com padrões sutís.

- **Use fontes de tamanho suficiente**, e nunca fixe o tamanho da fonte, permitindo que usuários com deficiências ampliem o tamanho da mesma.

- **Mantenha os textos parados!** Embora isso pareça estranho, há pessoas que adoram usar "marquees", com textos correndo, subindo e descendo, piscando... isso atrapalha **muito** a legibilidade.

- **Não use textos "TUDO EM MAIÚSCULA".** Além de ter uma aparência agressiva, o ser humano lê com muito menos velocidade textos totalmente em maiúsculas, causando um maior cansaço visual e irritação.

MULTIMÍDIA

- **Evite vídeos**, já que eles são demasiado lentos para as redes atuais. Entretanto, quando usá-los, indique o tamanho e tempo estimado de download.

- **Reduza as imagens de forma adequada**, para reduzir seu tamanho sem perda do significado. Normalmente ela deve ser cortada (crop), ressaltando a área de interesse, e depois redimensionada (resize) para reduzi-la a um tamanho adequado.

- **Evite animações**, pois elas costumam tirar a seriedade do site, além de aumentar os tempos de download. Seu uso excessivo também pode causar irritação no usuário.

- **Use áudio com parcimônia**, e sempre permita que o usuário o desligue. Evite música de fundo automática e efeitos sonoros que não possam ser desligados. Audioclipes podem, entretanto, dar uma maior dimensão à apresentação de conteúdo, mostrando um exemplo de um trecho de um CD ou uma ópera. Nestes casos, indique o tamanho e o tempo estimado de download.

- **Evite o uso de navegação 3D**, a menos que isso **realmente** facilite a compreensão. Usar 3D para ficar "bonito" em geral tem o péssimo efeito de tornar a navegação muito lenta. Navegação por ambientes virtuais (uma cidade onde cada prédio é uma página) é também uma péssima idéia, pois é um tipo de navegação muito lento, o que vai na contra-mão dos interesses do usuário típico da web.

## 3. Bibliografia

NIELSEN, J. *Projetando Websites*. Ed. Campus, 2000.

Notas da Aula 13: Páginas Web Dinâmicas
Prof. Daniel Caetano

**Objetivo**: Apresentar características das páginas Web Dinâmicas e situações de uso.

## Introdução

Apesar das WUIs serem relativamente mais jovens que as GUIs, a **WUIs** também **já passaram por diversos estágios** de desenvolvimento, amadurecendo a cada geração em direção à sua vocação atual, que **combina** a **transmissão de informações**, presente na idéia original da Web, com o **comércio eletrônico**, uma área em larga expansão já há alguns anos.

Pode-se dizer que a web passou pelos **estágios** conhecidos como "**Web Estática**", "**Web Dinâmica**" e atualmente tenha surgido o conceito de "**Web Ativa**". Ainda que as **mudanças** sejam apresentadas como gerações distintas da web, é importante ter em mente que elas **ocorreram de forma contínua e suave** ao longo dos anos. Além disso, uma "geração" não substituiu a outra: pelo contrário, **sites** com características das mais diferentes etapas **de cada uma destas fases** estão ainda no ar e **convivem pacificamente**.

## 1. Páginas Web Estáticas

A **primeira fase** da Web, como idealizada por Tim Berners Lee, é chamada hoje de "Web Estática". As páginas Web Estáticas são **compostas** apenas **por elementos básicos** como **tabelas**, **imagens**, **listas** e **links**.

Seu **uso é bastante limitado**, servindo basicamente à **difusão de conteúdo que não muda** (ou muda muito pouco) ao longo do tempo, com **pouca capacidade de busca** (limitada à capacidade do navegador do usuário). A questão do conteúdo de pouca variação (daí o nome: estático), oriunda da **dificuldade de atualização de páginas** construídas neste formato, foi ignorada no começo e, como resultado, há uma infinidade de "páginas fantasma" na Web, com conteúdo bastante desatualizado e que não verão jamais um atualização por todo o resto de sua existência.

Seu uso acertado foi basicamente para a **descrição dos produtos ou serviços** de uma empresa (coisas que, espera-se, não sofram tanta variação ao longo do tempo), além de **informações sobre objetivos de empresa** (que costumam mudar apenas em intervalos de alguns anos, quando a empresa tem sua diretoria ou presidência alterada).

Entretanto, a **dificuldade de atualização e gerenciamento** (incluindo adição de novas informações), **dificuldades para o cliente encontrar as informações** que procura (pela ausência de um sistema de busca adequado) e a **dificuldade inerente modificar o layout** de um site (na primeira fase, as informações de visualização eram inseridas **dentro** das páginas) fizeram com que este **formato fosse evoluído em poucos anos**. A viabilidade comercial das páginas puramente estáticas teve uma vida de menos de dois anos.

Uma das inovações mais recentes da Web que poderiam ter trazido sobrevida à Web Estática foi o advento das **Cascade Style Sheets**, ou CSS.

A criação das Cascade Style Sheets trouxe um **novo conceito** ao projeto de páginas Web, em que o **conteúdo é separado** da definição **das características visuais** de uma página. Isso significa que a página descrita em **HTML não deve conter** qualquer informação sobre **a "forma" da página**: deve **apenas** especificar o **significado** do conteúdo (o que é título, o que é texto, qual é a figura a mostrar, o que é legenda, etc). Toda **a especificação da forma**, que é atrelada ao significado do conteúdo, **é definida** externamente **em um arquivo do tipo CSS**.

Embora esta tecnologia **não tire as limitações de uso das páginas web estáticas**, seu uso **traz maior flexibilidade visual**, **permitindo a criação de layouts muito mais elaborados**, bonitos e - bastante importante - de **carregamento bastante rápido**. Além disso, a separação dos aspectos visuais do aspecto de conteúdo - algo que vem de encontro à arquitetura MVC, amplamente utilizada na Engenharia de Software - **facilita muito o gerenciamento de páginas e seu conteúdo**, bem como a **remodelagem visual de sites** inteiros.

Infelizmente, no que concerne a **negócios**, os **problemas** das páginas Web Estáticas vão além dos já citados, que são basicamente aspectos técnicos.

Primeiramente, as páginas web estáticas **não são capazes de receber entradas dos usuários**. Os contatos com as empresas (para adquirir produtos, por exemplo) precisam ocorrer por e-mail e, em geral, não podem ser automatizados. Além disso, elas **são pouco personalizáveis**, **comprometendo a usabilidade** e **dificultando a criação de usuários cativos**.

## 2. Páginas Web Dinâmicas

A primeira evolução da Web veio basicamente para **atender a** uma onipresente **necessidade de ferramentas de busca** mais poderosas do que as fornecidas pelos navegadores, além de **possibilitar o uso da web para negócios**.

O **ponto fundamental** desta primeira evolução foi a criação dos "**forms**", os populares formulários, que nada mais são do que uma adaptação de **elementos de entrada de dados** de uma GUI para uso em uma WUI. Os formulários devem ser usados com cuidado, **seguindo** basicamente as **regras** prescritas para **as GUIs**, ao mesmo tempo em que não deve ignorar as **regras de desenho das WUIs**.

As páginas Web Dinâmicas podem ser construídas de acordo com **3 conceitos distintos**, dependendo dos requisitos da aplicação/página sendo desenvolvida. Cada um destes conceitos possui tecnologias próprias, vantagens e defeitos:

- **Server Side, externo** à página: **CGIs**, **Servlets**. Usado quando se deseja uma **aplicação veloz** e **alta compatibilidade** com clientes.
- **Server Side, interno** à página: **ASP**, **JSP**, **PHP**. Usado quando se deseja **praticidade de desenvolvimento**, além da **alta compatibilidade** com clientes.
- **Client Side**: **JavaScript**, **VBScript**. Usando quando se deseja **reduzir a carga no servidor**, sob pena de uma **compatibilidade limitada**.

Adicionalmente, algumas outras tecnologias são utilizadas:

- **Banco de Dados**, para armazenar **perfil** de usuário, **conteúdos** em diversas línguas, últimos **lançamentos**, etc.
- **Cookies**, para armazenar **dados temporários** como **página visualizada** atualmente, **variáveis de sessão**, etc.

Talvez a **principal vantagem** das páginas web dinâmicas sobre as estáticas é que as dinâmicas **permitem a realização de negócios por meio eletrônico**. Mas isso não é tudo, já que elas também podem ser usadas para **melhorar o tratamento do cliente** por parte da empresa, através de **tratamento personalizado**. Isso inclui não apenas **chamar o usuário pelo nome**, mas também **adaptar as informações** ao gosto do usuário: **listar os últimos produtos** comprados pelo usuário, **apresentar novos produtos** que podem ser de interesse do usuário, **propor promoções especiais** de acordo com o perfil do usuário, etc.

Além disso, as tecnologias por trás da web dinâmica permitem uma **personalização das páginas** por parte do usuário, que pode **modificar o layout** de uma página, **escolher** quais **informações** serão apresentadas em sua página principal, **mudar o idioma** quando desejado, etc.

É importante lembrar que este tipo de característica **facilita a fidelização do cliente**, através de uma **melhor qualidade de serviço**, por **resolver os problemas do cliente de forma mais simples e rápida**, por **criar condições especiais para bons compradores**, etc.

Infelizmente a **Web Dinâmica também tem seus problemas**. As novas tecnologias e mecanismos de funcionamento fazem com que as **cargas no servidor sejam** bastante **maiores** que na web estática. Além disso, entre o comando do usuário e a resposta do servidor existe um tempo de tráfego das informações na rede, o que se traduz em uma **lentidão geral da interface**, perante o usuário. Para finalizar, as **interações são** bastante **limitadas**, resumindo-se ao preenchimento de formulários e seleções em mapas.

## 3. Páginas Web Ativas

A **segunda evolução** da Web veio basicamente para solucionar alguns problemas da web dinâmica: **reduzir os tempos de resposta** e **aumentar as possibilidades de interação**.

O **ponto fundamental** desta segunda evolução foi o uso dos "**plugins**", em especial os de máquina virtual, que possibilitaram a execução de programas na máquina do usuário,

quase que de maneira independente do equipamento e sistema operacional usados pelo usuário. As **regras de uso e design** para estes casos **são as mais variadas** e dependem do tipo de site e aplicação que se deseja criar.

As páginas Web Ativas são **sempre "client-side"**, ou seja, rodam na máquina cliente. Isso significa que as respostas às **interações** não dependem mais do tempo da rede, tornando-as muito **mais rápidas**, além de **reduzir a carga no servidor**. As **tecnologias** básicas do desenvolvimento atual de páginas ativas são o **Java**, **Flash** e, mais recentemente, o **Ajax**. **Bancos de Dados** também são usados, embora de forma menos extensiva (sob pena de aumentar muito a carga no servidor).

As **vantagens das páginas Web Dinâmicas** estão praticamente todas presentes nas páginas Web Ativas. Além disso, são vantagens adicionais a **total flexibilidade de interação e layout**, o que combinado à **velocidade de resposta** às ações do usuário e **uso extensivo de áudio e vídeo** podem melhorar muito a experiência do usuário.

Um uso que tem se tornado comum destas tecnologias é na **criação de aplicativos sob demanda**, em que se paga pelo tempo de uso do aplicativo, por exemplo (ou não se paga nada, como nos milhares de jogos em flash disponíveis na internet).

Infelizmente a **Web Ativa também apresenta alguns problemas**. O principal deles, sob a óptica do usuário, é a **falta de padronização das páginas** feitas usando as tecnologias de Web Ativa. Cada página tem um tipo de interação, um tipo de visual, etc... obrigado que o usuário tenha de "aprender" a usar cada nova página que encontrar.

Além disso, como as aplicações rodam em "máquinas virtuais" no lado do computador cliente, é necessário que a máquina virtual adequada, da versão correta, esteja instalada no equipamento do usuário, o que nem sempre é possível, gerando um **problema de compatibilidade**. Por esta razão, é comum que sites do tipo "ativo" possuam também uma versão do tipo "dinâmico".

O **uso de banco de dados** também **não costuma ser expressivo** nestas aplicações, para evitar o problema da sobrecarga do servidor. Entretanto, esta postura costuma reduzir a qualidade deste tipo de site, sob a óptica do usuário.

## 3. Bibliografia

NIELSEN, J. *Projetando Websites*. Ed. Campus, 2000.

TANENBAUM, A. S. *Redes de Computadores*. Editora Campus, 2003.

DUNCAN, C.B. *Flash MX: Criação e Desenvolvimento de Web Sites*. Editora Futura, 2003.

Notas da Aula 14: Eficiência e Avaliação de Interfaces
Prof. Daniel Caetano

**Objetivo**: Apresentar, resumidamente, formas comuns de testes de avaliação de UIs.

## Introdução

Durante todo o curso **foram citados testes** que devem ser realizados, medidas que devem ser feitas, dentre outras coisas... mas **pouco foi dito sobre as maneiras com que estes testes podem ser conduzidos**, **problemas potenciais** e **custos envolvidos**.

A seguir serão respondidas algumas destas questões de uma forma geral, que **servem** para testes de avaliação **tanto de GUIs quanto de WUIs**, **sejam** sistemas de funcionamento **interno ou externo**.

## 1. Testes de Interface

O **objetivo** dos testes é, com o menor trabalho e custo possível, **identificar se um sistema funciona a contento** e quais são as falhas existentes. Para que este objetivo possa ser alcançado, são necessárias duas **condições básicas**:

**- Encontrar usuários representativos**
**- Programar tarefas representativas**

Nenhuma destas duas condições é realmente simples de atender. No caso específico da **seleção de usuários representativos**, existe pelo menos uma indicação: podem ser **clientes** ou **funcionários**, dependendo do público alvo do sistema, atraindo a atenção dos mesmos para os testes através de **campanhas externas** de publicidade ou **campanhas internas** respectivamente.

As **camapnhas externas** normalmente **são mais complicadas** porque **não se deseja uma** verdadeira **torrente de solicitações** de pessoas querendo testar. Normalmente os testes externos **são um pouco mais direcionados**. As **camapnhas internas** são mais simples e vão desde a **seleção direta** dos usuários até **informativos nos murais** da empresa.

Os **métodos de seleção** mais comuns são o questionamento **direto aleatório** e a **escolha de indivíduos** para questionamento. A seleção aleatória é problemática porque nem sempre se consegue atingir um público de fato representativo (em termos de características, não em termos numéricos). Este problema pode ser solucionado se houver informações disponíveis sobre o público a ser testado, como no **RH** de uma empresa ou em institutos de pesquisa como o **IBGE**. É importante lembrar que, principalmente no caso de um sistema interno de uma empresa, tanto usuários **experientes** quanto **iniciantes** devem ser testados.

**Tipos de Teste**

Há basicamente dois tipos de teste: **Testes em Laboratório** e os **Testes em Campo**. Os **testes de laboratório** são aqueles em que se **projeta um teste específico** em um **local especial** e **usuários são convidados** para a realização daquele teste, sejam eles **filmados ou não**.

Os **testes de campo** possuem a característica inversa: o **teste** costuma ser **feito em cima do próprio software**, em que o **avaliador vai até a sala do funcionário** e **o observa trabalhando**, **interrogando o usuário apenas quando necessário**. É sempre importante deixar claro que o **avaliador não deve fazer maiores explicações sobre o design** do sistema nos momentos iniciais, deixando este tipo de comentário para o fim do teste. É importantíssimo, neste tipo de teste, que o avaliador **observe atentamente todos os usos inesperados** que o usuário faz do sistema, pois esta é a maior contribuição com relação aos testes em laboratório.

O avaliador deve **verificar** se, para o usuário, os **desenhos e elementos de interface são de fácil compreensão**, se o **fluxo de tarefas é simples**, se as **tarefas possuem uma exigência mínima de etapas adicionais** e, finalmente, se **existe a possibilidade de executar toda as tarefas** que o usuário tem necessidade de realizar.

## 2. Testes de Usuário Internacional

Este tipo de teste envolve um **número muito grande de questões** de compreensão e aceitação, e pode ter resultados distintos em países diferentes. Por esta razão, deve sempre ser **aplicado em diferentes países**.

Quando são testes **em aplicações do tipo "Web", é** bastante **simples disponibilizar** instrumentos de teste **internacionalmente**, já que a internet alcança a todos igualmente. Quando se tratam de **aplicações GUI**, entretanto, **nem sempre** os testes internacionais **são simples**, muitas vezes envolvendo **versões especiais** do sistema, exclusivas para tste.

**Contornando o Problema Lingüístico**

A primeira barreira existente é a questão do "**problema lingüístico**", não só por **diferenças entre** a língua do **sistema** e a língua do **usuário**, mas também relativo à dificuldade de **comunicação entre o usuário e o avaliador**. Existem três maneiras de contornar este problema.

Um **método aceitável** para fazer a avaliação é a produção de um **sistema no idioma do público do teste** e, indo até o local, **usar intérpretes para as conversas** entre o usuário e o avaliador. É importante lembrar ao **intérprete** para **não auxiliar o usuário**, apenas fazer seu papel de intérprete. Infelizmente, é sabido que algumas **sutilezas de comunicação são**

**perdidas nas traduções** de uma língua para outra, o que prejudica o desempenho deste tipo de avaliação.

Um **bom método** para a avaliação seria a **seleção de usuários que se comunicam** pelo menos minimamente **na língua nativa do avaliador**, quebrando a barreira da tradução. Entretanto, o **problema** aqui fica com relação à **representatividade da amostra**, já que usuários que se comunicam também na língua nativa do avaliador já possuem, por definição, um conhecimento diferenciado.

Uma **maneira melhor** de realizar a avaliação é pela **contratação e o uso de profissionais de usabilidade no país dos testes**, que falem a língua dos usuários e estejam habituados com as peculiaridades das interfaces locais. Neste caso, o desafio é **explicar claramente** aos profissionais de usabilidade que farão o teste sobre **as metas do teste** e também **o escopo das tarefas** a serem testadas.

**Em quantos países realizar o teste?**

Como dito anteriormente, realizar **testes em vários países pode ser positivo**, já que existem vastas diferenças culturais entre os países do mundo. Entretanto, **por questões financeiras e de tempo**, **geralmente não é possível** testar um sistema em um número ilimitado de países. Neste caso, em quantos testar?

Primeiramente, o cenário ideal seria **testar o sistema em todos os países que têm "importância comercial"** para a empresa produtora do software. Isso significa testar em todos os países em que o sistema é bem vendido ou nos quais se deseja ampliar o *market share*.

Caso o número de países da situação anteior seja muito grande, considerando-se a verba disponível, **pelo menos um país de cada região importante do mundo deve ser testado**, tentando evitar grandes equívocos na interface que causem embaraço futuro.

Finalmente, se nenhuma das alternativas anteriores forem possíveis, **pelo menos um país estrangeiro deve ser alvo de testes** do sistema, sob pena de permitir que o sistema tenha um funcionamento incompatível internacionalmente.

**Métodos para Teste**

- **Vá ao país estrangeiro**: é um método **muito bom** em termos de qualidade de resultado. Entretanto, os **problemas de língua** já citados aparecerão e, em geral, **não é** um **dos métodos mais baratos** de teste.

- **Faça o teste remotamente**: é um método que pode ser **bastante barato**, mas muitas vezes **exige adaptações do sistema** para que o acompanhamento possa ser feito. Ademais, é

um tipo de teste que nem sempre resolve completamente o **problema da diferença de língua**.

- **Contrate consultor local**: é **um dos melhores métodos**, embora seja também um **método caro**. Pode ter resultados superiores a todos os outros, se feito com cuidado.

- **Uso do pessoal da filial (sem treinamento):** caso a sua empresa tenha filial no país do teste, usar o pessoal de lá é uma alternativa para **reduzir custos**. Entretanto, pelo fato de o pessoal não ser especializado, os resultados **dificilmente terão uma qualidade surpreendente**.

- **Teste auto-administrado**: os testes auto-administrados podem ser aplicáveis em alguns casos e podem ter um **custo bastante acessível**. Entretanto, eles **exigem um planejamento especial**, em específico com relação à **compreensão das questões** colocadas pelo teste. Uma interpretação ruim de uma das atividades do teste pode turvar os reais resultados do teste.

**Agradecimentos de Testes**

**Agradecer com presentes** aos usuários que se propõem a participar dos testes é uma **prática bastante comum**, mas que deve ser considerada com o devido cuidado.

É importante lembrar que estamos testando interfaces em diferentes países porque suas **culturas podem ser substancialmente diferentes**. Da mesma forma, a maneira com que diferentes tipos de presentes são recebidos pode variar bastante de um local para outro.

Não deixe de agradecer os usuários de alguma forma, mas procure avaliar qual é a melhor forma para cada país em específico.

## 3. Bibliografia

NIELSEN, J. *Projetando Websites*. Editora Campus, 2004.

Notas da Aula 15: Web Semântica e Interfaces para o Futuro
Prof. Daniel Caetano

**Objetivo**: Apresentar o conceito de Web Semântica e conceitos de futuras UIs.

## Introdução

Durante todo o curso **estudamos as interfaces como se encontram** nos dias atuais. Mas o que está por vir? **Que mudanças** devemos **esperar para um futuro** próximo e, talvez, para um futuro não tão próximo?

É certo que haverá evoluções: **as interfaces estão** (e sempre estiveram) **em contínua evolução**, embora em alguns momentos este desenvolvimento tenha ocorrido de forma bastante lenta.

Entretanto, a maioria das **mudanças aparentes devem ser precedidas por** profundas **alterações nas estruturas internas** dos sistemas, softwares e até mesmo nas estruturas dos documentos.

Como um exemplo de um período de transição, temos as mudanças que levarão futuramente ao conceito da **Web Semântica** e, como exemplo de mudanças que ainda estão por vir, embora a maioria dos componentes base já estejam em nosso dia-a-dia, são as interfaces **sem comandos** e as **interfaces ubíquas**.

## 1. A Web Semântica

**Na atualidade, a World Wide Web é uma mídia** bastante desenvolvida, como foi visto anteriormente, **com** recursos avançados que compõem tecnologias como a Web Dinâmica. Entretanto, apesar do advento das Cascade Style Sheets e a conseqüente separação entre conteúdo e características visuais, **a parcela de conteúdo** continua **sendo** escrita como sempre foi feito: **uma descrição de informações focada apenas na leitura humana**.

Entretanto, a mecanização das tarefas do dia-a-dia tem sido quase que uma necessidade na sociedade moderna e, no que se refere à atividades realizadas na web, muito pouco se evoluiu neste sentido. Em outras palavras, é **quase impossível mecanizar** a maioria das **tarefas que as pessoas realizam na web**, como procurar um preço mais baixo de um produto ou mesmo **agendar uma viagem, incluindo hotel e vôo automaticamente**... é preciso que alguém realize esse processo **manualmente**. Quando ele é feito por softwares, na forma como a Web é apresentada hoje, o resultado é quase sempre aquém do esperado pelo usuário.

Como uma proposta para este tipo de problema, foi desenvolvido pela IBM e Microsoft uma tecnologia chamada "**Web Services**", que é **baseada nas tecnologias da web**,

compartilha recursos com a web, **mas está fora da World Wide Web**, no sentido que **não se acessa um Web Service pelo navegador**!

Especificamente, as **tecnologias padrão usadas** pelos Web Services são **XML e HTTP**, sobre os quais **foram construídas tecnologias como SOAP, WSDL e UDDI**. O que essa tecnologia implementa de diferente do proposto pelo HTML? A diferença básica é que são **tecnologias voltadas à compreensão dos dados por parte do computador**. A rede constituída pelos "Web Services" é como se fosse **uma espécie de World Wide Web dos** chamados **"agentes de software"**, para que esses agentes possam realizar tarefas pelos seus "donos" de forma mecânica.

Como é possível observar, **vivemos hoje num mundo segmentado**: temos **dados** disponíveis na forma **para a leitura humana e dados** disponíveis na forma **para leitura dos equipamentos**. A **proposta da Web Semântica** é exatamente a **unificação destas duas categorias de dados em uma única**: documentos web que possam ser lidos facilmente tanto por humanos quanto por agentes de software.

O mecanismo de implementação é um **documento Web normal**, **acrescido de "tags" que indiquem o significado semântico** das palavras, para que eles possam ser compreendidos por humanos e máquinas. Por exemplo:

```
<USUARIO>
        <NOME>Fulano DeTal</NOME>
        <IDADE>18</IDADE>
        <ENDERECO>
                <RUA>Rua das Garças</RUA>
                <NUMERO>17</NUMERO>
                <BAIRRO>Bairro do Limoeiro</BAIRRO>
        </ENDERECO>
</USUARIO>
```

Os "**agentes de software**" **são**, então, **programas capazes de ler páginas web** descritas desta forma, que entram na mesma página web do hotel e do aeroporto que um ser humano entraria, **encontram as informações que interessam ao seu "dono" e tomam atitudes necessárias**, como informar ao usuário um resumo das melhores opções ou ainda tomando decisões ativas, como marcando uma viagem ou uma consulta com o médico.

## 2. Interfaces Sem Comandos e Interfaces Ubíquas

As **interfaces dominantes nos dias de hoje** são as conhecidas pela alcunha de "**Interfaces WIMP**", ou seja, aquelas constituídas basicamente por janelas, ícones, menus e ponteiro de mouse. Estas têm sido a interface dominante das aplicações pelos últimos 20 anos e devem continuar a ser por algum tempo. Entretanto, **a evolução e integração dos equipamentos que vem ocorrendo** nos últimos anos trouxeram novas necessidades que **exigirão novos paradigmas**.

A maior aposta dos estudiosos de interfaces, hoje, é que no futuro nos depararemos com as "interfaces sem comandos" e as "interfaces ubíquas", provavelmente ambas ao mesmo tempo. Mas o que querem dizer estes termos?

Bem, uma "**interface sem comandos**" **não quer dizer exatamente "sem comandos"**, **mas sim sem rigidez sintática**. A idéia é que **as pessoas se expressem como quiserem** e **os equipamentos terão de entender o que elas querem**.

Antigamente existia uma piada, que vale ainda hoje, que dizia que "os computadores não entendem errado; o problema é que eles fazem o que o usuário manda, e não o que o usuário quer". Esta afirmação - verdadeira, por sinal - é uma brincadeira com uma característica da comunicação humana, que é a de se expressar com ambigüidades. No paradigma proposto das "interfaces sem comandos", no fundo, as máquinas terão que começar a agir de acordo com o que o usuário quer, e não de acordo com o que ele diz.

Algumas linguagens de programação, como o **InterLisp**, já possuem **mecanismos de reinterpretar entradas do usuário que**, em princípio, **não fazem sentido**, buscando encontrar um significado na maneira com que o usuário se expressou, ao invés de retornar simplesmente um "erro de sintaxe".

Uma "**interface ubíqua**" ou "**interface defenestrada**" **é uma interface que não é localizada, é espalhada pelo ambiente**. Por exemplo: os dados informados não estarão sempre em um monitor e a entrada de dados não será necessariamente num periférico chamado teclado. O usuário irá falar, se mover para se comunicar com os equipamentos e irá visualizar informações no lugar mais conveniente a cada momento.

Mas o que leva os especialistas a prever interfaces tão complexas e diferentes das atuais? Vários fatores. Vamos a alguns deles:

**- Popularização de alguns sistemas de realidade virtual;**
**- Reconhecimento e interpretação de sons e vozes já são uma realidade;**
**- Reconhecimento de gestos já é uma realidade;**
**- Muitos avanços na área de inteligência artificial;**
**- Popularização de computadores altamente portáteis (celulares, PDAs...)**

Algumas "**aplicações conceituais**" **já existem**, como por exemplo um **Museu Virtual** que envolve **uso de realidade virtual com uma interface** altamente **defenestrada**, **controlada por gestos**.

**Do ponto de vista do usuário, a mudança será radical**. O usuário poderá **deixar de se preocupar com a sintaxe** dos comandos para **se preocupar apenas com a tarefa** que precisa realizar.

Isso será possível porque **não existirá uma unica aplicação rodando**, na qual o usuário teria foco, como acontece hoje em dia. **Todas as aplicações estarão rodando ao**

**mesmo tempo**, **todas elas "lendo" o usuário - através de diversos dispositivos** - e **interpretando suas ações**, compreendendo quando o usuário quer que elas façam alguma coisa ou não.

Um filme que apresenta **exemplos** de como os estudiosos estão visualizando o futuro das interfaces é o filme **Minority Report**. O sistema de visualização de imagens da polícia, o sistema de visualização de filmes em hologramas, as lojas que "lêem" os usuários e apresentam-lhes ofertas automaticamente...

Certamente parece um mundo repressor. Mas o mundo de hoje, altamente controlado por computadores também pareceria repressor a pessoas do início do século XX. Assim, existe uma função até mesmo para a lentidão deste tipo de transição, que é a de permitir que todos nós nos adaptemos a esta nova realidade para que, quando estivermos diante dela, ela nos pareça tão natural que não consigamos imaginar como vivemos antes de sua criação.

## 3. Bibliografia

NIELSEN, J. *Noncommand User Interfaces*. Comm. of the ACM. Abril de 1993, v36, n4, p83-89.

FERREIRA, M.A.G.V. Notas de Aula - Tópicos de Comunicação Homem-Máquina, EPUSP, 2003.

CAETANO, D. J. *Grids e Web Services: Uma combinação de futuro*. Trabalho de disciplina EPUSP, 2006.