

Exame Teórico IHC:

Human Computer Interaction:

Disciplina que se preocupa com o design, avaliação e implementação de sistemas de computação interativa para humanos usarem e com o estudo dos principais fenômenos que os rodeiam.

Sistemas interativos incluem um “módulo que **não podemos controlar**:

O **utilizador** que:

- é muito complexo
- não é bem conhecido
- não podemos controlar

Os utilizadores podem ser bastante diferentes

Não projetar um sistema interativo com cuidado tem um preço elevado:

- Aumento o tempo de desenvolvimento
- Aumento custos e complexidade de manutenção
- Aumento complexidade do código
- Prejudica a satisfação do utilizador

Usabilidade:

Requisito não funcional, diretamente relacionado com a capacidade dos utilizadores atingirem o seu objetivo por si mesmos.

Aspetos fundamentais:

- Fácil de aprender e lembrar
- Fácil de usar – eficiência (realiza de maneira correta as suas funções) e eficácia (qualidade daquilo que cumpre com as metas definidas)
- Satisfação
- Visibilidade de estado
- Poucos erros e fáceis de superar

Princípios de Aprendizagem:

- Possibilidade: propriedade que determina como usar
- Visibilidade: as partes relevantes do problema devem ser visíveis
- Feedback: as opções devem ter efeitos visíveis
- Consistencia: coisas parecidas devem ter o mesmo aspecto e comportamento
- Falar a linguagem do utilizador: não usar termos técnicos
- Usar metáforas: analogia de pastas que contêm ficheiros
- Seguir standards de plataforma

O utilizador:

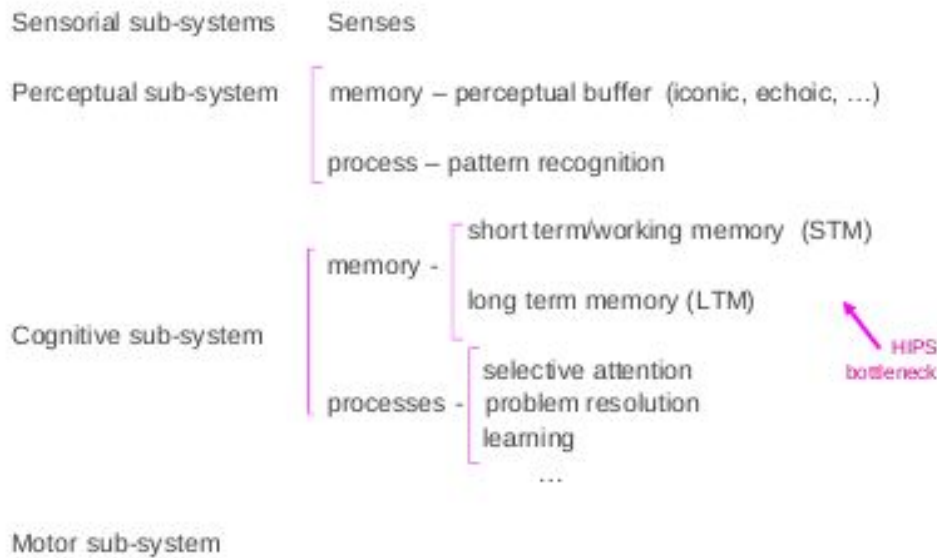
Perfil do utilizador:

- HIPS
- Conhecimento e experiência *
- Trabalho e tarefas *
- Características físicas *
- Ambiente *
- Ferramentas *

* variável entre utilizadores

HIPS – Human Information Processing System:

- Sentidos: Visão, audição, tato, cheiro, sabor
- Memória
- Sub-sistema perceptivo
- Sub-sistema cognitivo



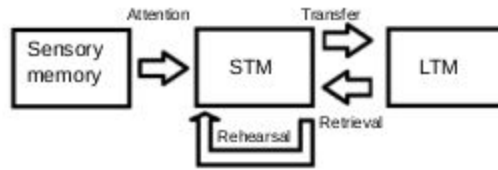
Sub-sistema sensorial I/O:

Input: os 5 sentidos, no entanto alguns são mais relevantes, para IHC a visão é o preferido

Output: sistema de comunicação

Memória:

- Memória sensorial
 - Muito curta – 1 a 2 seg
- Memória de curta duração – memória de trabalho
 - Calculos mentais
 - Leitura
 - +/- 18 seg
 - Fácil de aceder e de esquecer
 - 7+/-2 items
- Memória de longa duração
 - Acesso lento
 - Praticamente infinita em tamanho e duração
 - 3 processos: armazenar, esquecer, recuperar



Como estas características influenciam o design de UI:

- Muita experiência de utilização, mas baixa experiência de tarefas
-> mais ajuda semântica
- Muita experiência de tarefas, mas baixa experiência de utilização
-> mais ajuda sintática
- Elevada frequência de uso -> fácil de usar
- Baixa frequência de uso -> fácil de aprender e lembrar
- Uso obrigatório -> fácil de usar
- Uso opcional -> fácil de aprender e lembrar
- ...

Avaliação Heurística:

Objetivo: Encontrar problemas de usabilidade. Pode ser usado nas várias fases do projeto.

Vantagens: Fácil (mesmo para utilizadores inexperiente), Barato, Rápido

Desvantagens: Subjetivo (depende do avaliador), Dificuldade em arranjar especialistas

1. Visibility of System Status:

O utilizador deve saber o que se passa dentro do sistema. Deve ser recebido feedback dentro de um tempo razoável

2. Match Between System and the real world:

A aplicação deve falar a linguagem do utilizador - deve ser perceptível

3. User Control and Freedom:

Deve ser dada ao utilizador a liberdade para navegar e realizar ações - ex. Liberdade para fazer undo

4. Consistency and Standards:

Se a informação é mostrada de uma certa forma, deve ser sempre mostrada assim

5. Error Prevention:

Ex. autopreenchimento do google ou alertas de erros na escrita de emails

6. Recognition rather than recall:

É sempre preferível dar ao utilizador um conjunto de opções do que obrigá-lo a decorar

7. Flexibility and efficiency of use:

A app deve ser fléxivel , ao ponto de se conseguir transformar entre 1 utilizador antigo e 1 novo - ex. Durante a instalação escolher default ou definições avançadas

8. Aesthetic and minimalist design:

A informação apresentada deve ser toda necessária e útil - ex. ter botão learn more

9. Help users recognize, diagnose and recover from errors:

Deve ser verificado se os erros são corretamente explicado ao utilizador

10. Help and Documentation:

Deve ser fornecida ajuda adequada

Grau de Severidade:

0 - não é um problema de usabilidade

1 - problemas de cosmética - são resolvidos se houver tempo

2 - problemas minimos de baixa prioridade

3 - deve ser dada grande prioridade para resolver

4 - catástrofe de usabilidade - resolver antes do produto ser lançado

Os problemas devem ser avaliados de acordo com: Frequencia - comum ou novo; Impacto - os utilizadores ultrapassam facilmente?; Persistência - é um problema que os utilizadores ultrapassem, ou vão cometê-lo repetidamente?

Cognitive Walkthrough:

- Processo analítico - não envolve users
- Baseado no facto que, normalmente, os utilizadores preferem aprender um sistema usando-o
- Produz resultados rápidos a um preço baixo
- Aplica-se em fases precoces, antes de código

Proceso:

1. Análise de tarefas - ações requeridas por um user para realizar uma tarefa, e as respostas do sistema
2. Os desenvolvedores fazem perguntas ao grupo
3. Responder às questões e ver as problemas de usabilidade encontrados
4. Reportar os problemas principais
5. Redesenhar a UI para responder aos problemas encontrados

Modelos de Design:

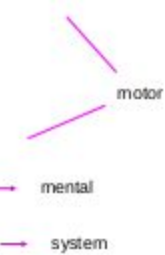
KLM - Keystroke-Level Model:

Modelo de tarefas de iteração unitária (ex: alterar a fonte de uma palavra, usar pesquisa e substituir)

Estas tarefas têm 2 fases:

- Aquisição: construção da representação mental da tarefa
- Execução: usar o sistema

Pode ser decomposta em 7 operadores

- K- Keystroke (varies with typing skill)
 - B- Button press of the mouse
 - P- Pointing at a target (Fitts' law)
 - H- Homming between mouse and keyboard
 - D- Drawing using mouse
 - M- Mentally preparing for physical action → mental
 - R- System Response (often may be ignored) → system
- 

Task Analysis and GOMS:

Task Analysis	GOMS
models: real world aspects not part of the system	user cognitive processes while performing task
Point of view: external	internal
S/W lifecycle: early phases	evaluation

Task Decomposition:

Hierarchical Task Analysis (HTA) - das técnicas de análise mais utilizadas e produz:

- Uma hierarquia de tarefas e sub-tarefas
- Planos com uma sequência e condições de execução





Tipos de planos na HTA:

- Sequência fixa (plano 3 - preparar o bule)
- Tarefas opcionais (5,3 açúcar?)
- Esperar por eventos (4- espera 4 ou 5 minutos)
- Ciclos (plano 5 - servir chá)
- Partilha de tempo (1 e 2 preparam bule de chá, fervem a água)
- Aleatório (salas de limpeza a vácuo)
- Mix de vários tipos

Modelos mentais e conceptuais:

Modelo mental: É a representação interna de como o utilizador compreende o sistema (Exemplo: iPad ebook - Utilizador agora como ler um livro)

Permite fazer presunções e determinar casos de evento MAS são incompletos, instáveis e não científicos.

Modelo conceptual: É o modelo real que é dado ao utilizador através da interface do produto. (Exemplo: ebook iPad - Botão, telas, eventos - interface real representando)

Se o modelo conceitual do produto não corresponder ao modelo mental do utilizador:

- o utilizador acha o produto difícil de aprender
- pode causar frustração.

Metáforas: Tiram partido dos modelos mentais existentes do mundo real. Pode ser difícil, engenhoso e restritivo para o user.

Prototipagem:

Adequado para receber feedback; não precisa de ter muito detalhe ou ser muito realista.

Prototipo em papel:

- Fácil de construir
- Fácil de mudar
- Foca-se a atenção na “big picture” e não se preocupa com detalhes
- Pessoas, que não programadores, podem ajudar

Aprende-se:

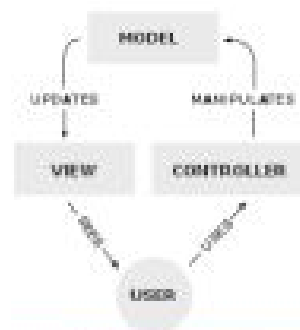
- Modelo conceptual
- Funcionalidades
- Navegação e fluidez
- Funcionalidade
- Terminologia

User Interfaces Software Architecture:

Model-View-Controller (MVC):

Padrão de arquitetura normalmente usado para desenvolvimentos de UI
Expressa o "núcleo da solução" para um problema, permitindo que ele seja adaptado para cada sistema.

Divide a aplicação em 3 partes : Model, View e Controller



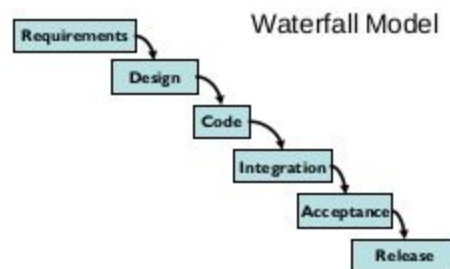
Model – Componente central que expressa o comportamento da aplicação em termos do domínio do problema

View – Output a representar informação

Controller – Aceita um input e converte para comandos da model ou view

Waterfall Model:

Os utilizadores não estão envolvidos na validação até o teste de aceitação
Falhas na interface do utilizador geralmente causam mudanças nos requisitos e no design



User-Centered Design:

- Considera as necessidades dos utilizadores durante todo o processo de design
- Design iterativo
- Foco inicial nos utilizadores e nas tarefas
 - análise do utilizador: quem são os utilizadores
 - análise de tarefas: o que eles precisam fazer
- Envolve tanto utilizadores como avaliadores, consultores e, às vezes, designers
- Avaliação constante
- Utilizadores estão envolvidos em todas as iterações
- Todo protótipo é avaliado de alguma forma

Envolve saber princípios de usabilidade e paradigmas. Deve-se conhecer os exemplos de sucesso, porque eles funcionam, usar os métodos adequados .

Todas as propostas de metodologias UCD são iterativas e incluem avaliação de usabilidade em iterações.

RESUMO

- Modelos para desenvolvimento de software
Modelo de queda de água faz sentido para projetos de baixo risco;
Modelos em espiral ou iterativos são necessários quando os requisitos e o espaço de design são desconhecidos ou arriscados;
Desenvolvimento de interface do utilizador é muitas vezes arriscado;
- Processo de design centrado no utilizador
Iterativo, orientado por protótipo
Foco inicial em utilizadores e tarefas
Avaliação constante

Estilos de Iteração:

- Menus
- Manipulação direta
- Fill-in-forms
- Dialog boxes
- Function keys
- Command languages
- Natural languages
- 3D interfaces

Manipulação direta:

As ações são realizadas diretamente nas representações visuais dos objetos de informação. Caracterizadas por:

- Representação Contínua de Objetos
- Ações físicas em vez de linguagens de comando
- Ações rápidas, incrementais e reversíveis com resultados visíveis

Semantic Distance: Distância subjetiva entre o goal do user e a semântica da interface

Articulatory Distance: Distância entre o significado das ações e a sua forma física

Fill in Forms:

Úteis para trabalhos de rotina, administrativos ou para tarefas que exigem muita entrada de dados.

Atualmente são frequentemente usados com outros estilos

Dialog Boxes

Combinações entre menus e fill-in-form
(FALTA COISAS #9)

Input Interfaces:

Ponting Devices:

Usados para apontar ou selecionar um alvo, desenhar, etc.

A sua eficiência varia de acordo com as tarefas que