

INTERAÇÃO HUMANO- COMPUTADOR – AULA 4

Prof^a. Ana Cláudia Guimarães Santos
1º semestre de 2024



TÓPICO DA AULA

Agora que já vimos na Unidade 1 sobre os Fundamentos da IHC, entraremos a partir dessa aula na Unidade 2: Construção de uma interface gráfica simples.

Temas da aula:

1. Requisitos

- Entendendo usuários e necessidades;
- Quem são os usuários? Quais são as necessidades dos usuários?
- Identificação de **necessidades** e **requisitos**;
- Questionários;
- Entrevistas;
- Aspectos Éticos;

quem faz o levantamento de requisito? StackHolders; gerentes

2. Construção de um benchmark.

USUÁRIOS

Quem são os usuários? A resposta óbvia é: **quem usa (ou usará) o sistema.**

Mas também chamamos de usuário quem relação direta com quem usa o sistema (por exemplo: chefe ou clientes do usuário).

“Usuários são aqueles que gerenciam usuários diretos, aqueles que recebem produtos do sistema, que testam o sistema e aqueles que utilizam produtos de concorrentes.”

Holtzblatt e Jones 1993

De acordo com Eason (1987), nós temos:

Usuário primário: aqueles que serão usuários frequentes do sistema;

Usuário secundário: aqueles que utilizarão o sistema às vezes;

Usuário terciário: aqueles que serão afetados pelo sistema.

QUEM SÃO NOSSOS USUÁRIOS?



Pode ser literalmente qualquer pessoa no mundo!

USUÁRIOS

O design de IHC precisa pensar, analisar e implementar as necessidades do usuário para o produto em desenvolvimento. Para isso, podemos responder:

- O que o usuário está tentando alcançar com o software?
- Como a atividade é realizada atualmente?

Para além disso, devemos pensar: quais são as novas necessidades que o usuário terá quando o sistema for implementado?

Essas necessidades são pensadas inicialmente durante o **levantamento de requisitos**.

REQUISITOS DE SOFTWARE

Requisito: Condição básica e necessária para se obter alguma coisa ou para alcançar determinado propósito.

- **Requisitos de software são as condições básicas para um sistema, ou seja, o que o sistema deve fazer.**

TIPOS DE REQUISITOS

REQUISITOS FUNCIONAIS
Especificam uma ação que o sistema precisa ser capaz de realizar.

TIPOS DE REQUISITOS

REQUISITOS NÃO FUNCIONAIS
Pode ser considerado um atributo de qualidade, segurança ou desempenho.

ENGENHARIA DE REQUISITOS

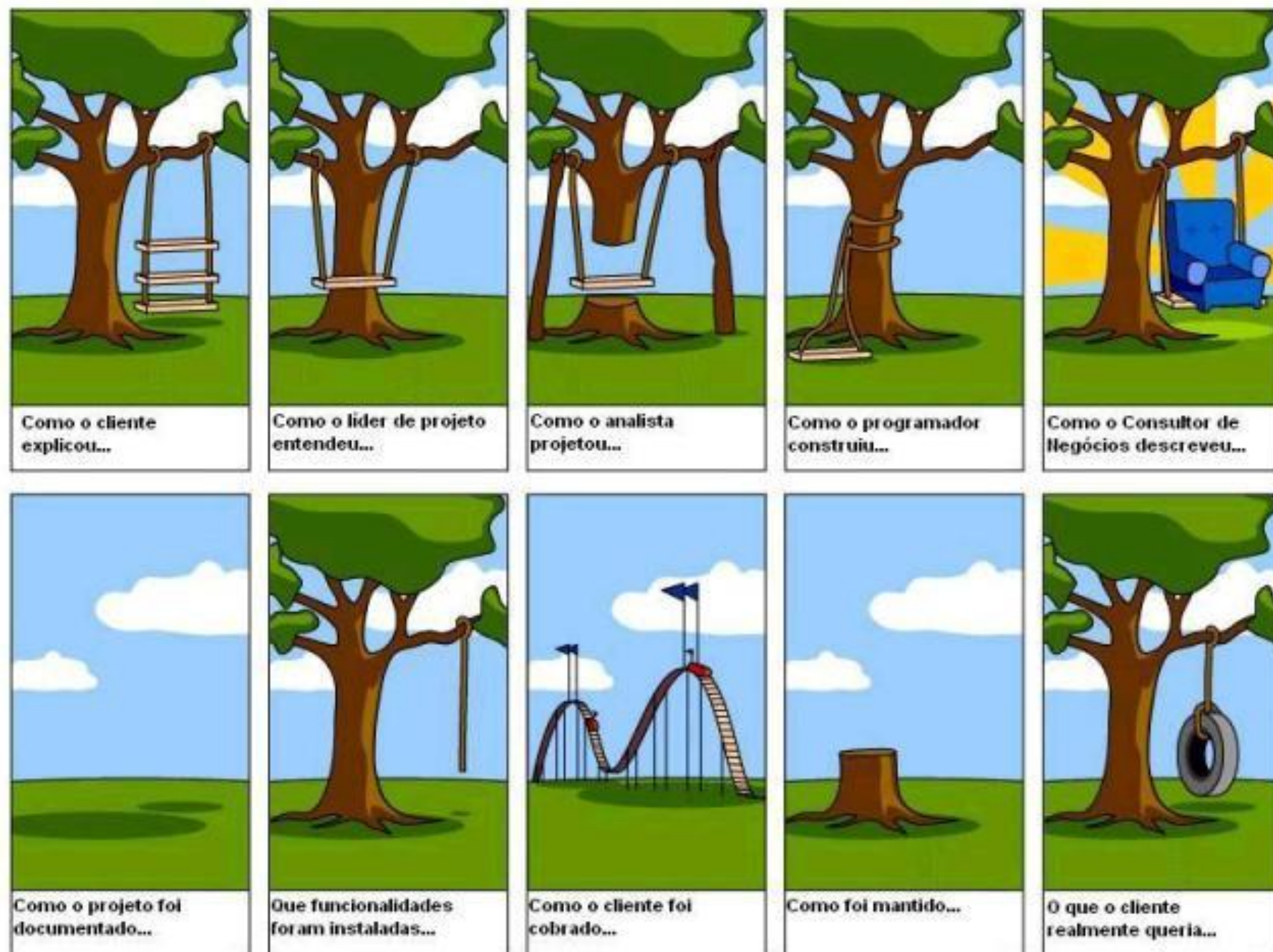
Segundo Pressman, a engenharia de requisitos abrange sete tarefas distintas:

1. Concepção;
2. Levantamento (ou elicitação);
3. Elaboração;
4. Negociação;
5. Especificação;
6. Validação; e
7. Gestão.

ELICITAÇÃO DE REQUISITOS

- O sucesso de um software de excelência depende da capacidade do engenheiro de software de compreender as necessidades e expectativas dos *stakeholders* (partes interessadas).

ELICITAÇÃO DE REQUISITOS

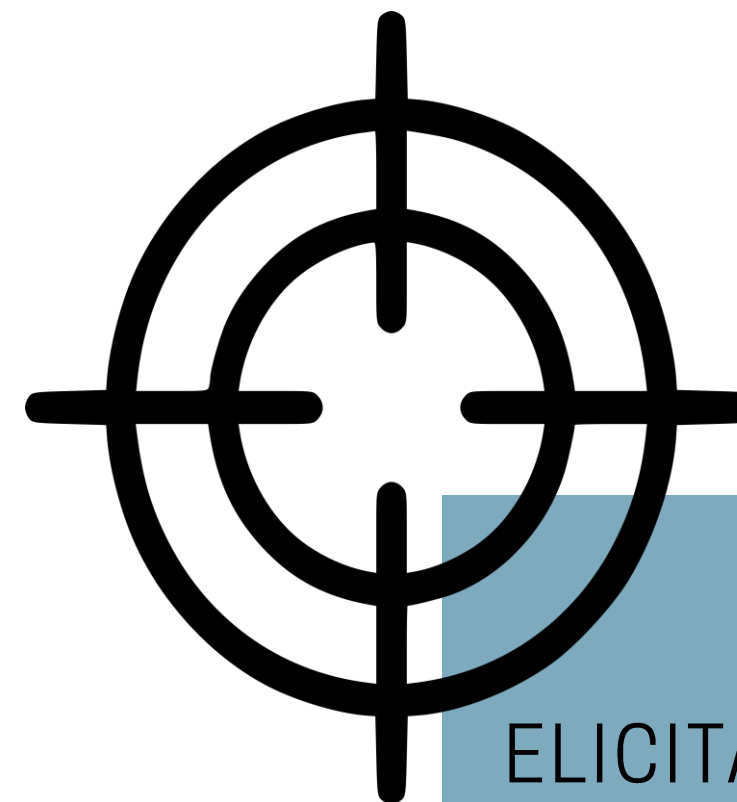


“The users cannot describe what they really do because they are not conscious of it and do not reflect on it.”
Holtzblatt & Beyer, 1993

ELICITAÇÃO DE REQUISITOS

Quem participa da fase de elicitação de requisitos?
Os Stakeholders.

1. Gerentes;
2. Especialistas de domínio;
3. Usuários finais do software.



ELICITAR= Extrair, criar, descobrir.
No caso da engenharia de requisitos, obter o máximo de informação sobre o projeto/objeto em análise.

ELICITAÇÃO DE REQUISITOS

Problemas que podem ser encontrados durante o levantamento de requisitos:


1. **Problemas de escopo:** ocorrem quando os **limites** do software são definidos **precariamente**;
2. **Problemas de entendimento:** ocorrem quando os **stakeholders** não possuem entendimento completo sobre o negócio e suas necessidades; sem entendimento
3. **Problemas de volatilidade:** ocorrem quando os **requisitos mudam** muito rapidamente.

ELICITAÇÃO DE REQUISITOS



IDENTIFICAÇÃO DAS FONTES DE INFORMAÇÃO

O **stakeholder** não é a única **fonte de informações** para o **levantamento de requisitos**. Outras fontes são:

- 
1. Documentação;
 2. Legislações;
 3. Contratos;
 4. Livros sobre a área do sistema;
 5. E até mesmo outros sistemas da empresa!

PRINCIPAIS TÉCNICAS DE ELICITAÇÃO

Diferentes técnicas podem ser utilizadas na fase de levantamento de requisitos:

1. **Técnicas Tradicionais**, as formas mais comuns de levantamento de requisitos. São exemplos: Entrevistas, questionário e observação;
2. **Técnicas de Elicitação de Grupo**, incentivam a geração de ideias pelos stakeholders. São exemplos: Workshops e brainstorming;
3. **Prototipação**, envolve a criação de protótipos para auxiliar a visualização do sistema;
4. **Técnicas Contextuais**, envolvem um entendimento mais profundo do contexto onde o sistema será utilizado. Exemplo: etnografia.

COMO ESCOLHER A MELHOR TÉCNICA?

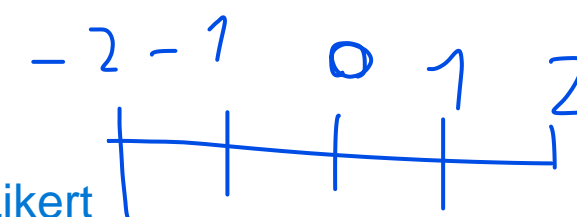
O engenheiro de software deve analisar diferentes aspectos antes de selecionar a técnica empregada na elicitação de requisitos, por exemplo, **tempo** e **recursos** disponíveis, **complexidade** do projeto, quantidade de informações que precisam ser obtidas.

QUESTIONÁRIOS

Questionários podem ser uma ótima forma de elicitação de requisitos. Eles podem ser aplicados de forma impressa ou online, contendo perguntas fechadas ou abertas. Entretanto, as questões devem ser formuladas de **forma clara**, ou ter instruções de como o usuário deve responder.

QUESTIONÁRIOS

ex.: Uso da escala Likert



A hand-drawn diagram of a Likert scale. It consists of a horizontal line with five vertical tick marks. Above the tick marks are the labels -2, -1, 0, 1, and 2. The line starts with a vertical bracket on the left and ends with a vertical bracket on the right.

O questionário precisa ser organizado de forma que contenha:

1. Uma Introdução (o que é, os objetivos e quem deve responder);
2. Perguntas de **natureza demográfica** (idade, gênero, escolaridade, etc);
3. Perguntas sobre a **experiência** do usuário;
4. Perguntas **específicas** do que se pretende descobrir.

ELICITAÇÃO DE REQUISITOS - ENTREVISTAS

Em uma entrevista, o engenheiro de software formula questões para os stakeholders sobre o negócio, o software a ser desenvolvido e os futuros usuários do software.

Tipos de entrevistas:

1. **Entrevistas fechadas (estruturadas):** o engenheiro de software elabora um conjunto predefinido de perguntas;
2. **Entrevistas abertas (não estruturadas):** não há um roteiro predefinido, perguntas são realizadas de maneira menos estruturada, os envolvidos falam abertamente;
3. **Entrevistas mistas (semiestruturadas):** o engenheiro de software realiza uma mistura entre entrevistas abertas e fechadas.

PLANEJAMENTO DA ENTREVISTA

É possível dividir a entrevista em três partes:

Quebra de gelo

Questões livre de contexto. Ajudam o stakeholder a se sentir confortável na entrevista, “traz a mente” dele para o entrevista.

“Quem vai usar a solução?”

Faço

Entendimento do problema

Questões que ajudam a entender o problema que o software visa resolver.

“Você pode me mostrar ou descrever o ambiente no qual a solução será usada?”

ENTENDIM

Finalização (efetividade)

Questões sobre a efetividade da entrevista.

“Alguém mais pode fornecer informação adicional?”
“Tem alguma questão que não fiz que você julga pertinente?”

EFETIV.

ELICITAÇÃO DE REQUISITOS - ENTREVISTAS

Em entrevistas, o engenheiro de software deve ouvir os stakeholders e não ter ideias preconcebidas sobre os requisitos.

Um problema das **entrevistas** é que elas **podem não ser úteis** para a compreensão de requisitos de domínio: os stakeholders podem estar tão acostumados com algum termo ou conhecimento do domínio que eles não conseguem explicar ou nem ao menos os mencionam na entrevista.

ASPECTOS ÉTICOS

O entrevistador deve se preocupar com aspectos éticos (ou deveres morais) durante a entrevista:

1. Respeitar a privacidade do entrevistado;
2. Evitar danos (ex: perda de emprego; perda de bens; perda de informação; danos a propriedades);
3. Honrar a confidencialidade das informações recebidas.

É de responsabilidade da equipe de design proteger o participante (bem estar físico e mental) durante a realização da entrevista/pesquisa.

ATIVIDADE

Vamos pensar em uma entrevista?



BENCHMARK

Benchmark, em português “ponto de referência” é o processo de pesquisa (ou comparação) entre produtos, serviços e práticas similares. É uma ferramenta de gestão, que além de ajudar a aprimorar produtos e serviços, nos dá uma noção de como estão outras empresas do mesmo setor.

Em computação também pode ser a avaliação de performance de código.



BENCHMARK

E porque fazemos isso em IHC?

Para comparar soluções de design já existentes com a nossa solução. Isso nos leva à:

- Identificação de novos requisitos;
- Identificação de tendências de mercado;
- Avaliação de mercado.

BENCHMARK

Benchmarking nos ajuda a entender o impacto dos nossos produtos e também indica itens para melhoramento. Também pode ser útil quando precisamos tomar decisões sobre o design de um produto.

Também pode envolver dados quantitativos, por exemplo:

- Tempo levado para fazer uma compra;
- Número de cliques até conseguir enviar/comprar algo;
- “Eight-week retention rate”: percentual de usuários que continuam a usar o aplicativo depois de oito semanas.

BENCHMARK

UX Benchmarking

UX benchmarking is the process of evaluating a product or service's user experience by using metrics to gauge its relative performance against a meaningful standard.



BENCHMARK

Para o trabalho do semestre, cada grupo deverá fazer um benchmark.

Passos:

- Pense no seu produto;
- Escolha três soluções para o mesmo problema que o seu produto visa atender;
- Faça uma análise das seguintes perguntas:
 - Quem é o público-alvo das três soluções?
 - Como a solução se comporta em diferentes dispositivos (desktop/celular)?
 - Quais são as funcionalidades demonstradas?
 - Como é o layout das soluções? (elementos da interface, localização dos menus, cores, etc).

BENCHMARK

Elabore um relatório (postar no AVA) e uma apresentação sobre as repostas das perguntas anteriores. Outras coisas que podem ser incluídas: idioma da funcionalidade; se possui instruções de uso; se possui um canal de atendimento e se é possível aumentar a fonte.

O grupo deve decidir quais itens a nova solução irá considerar e quais não irá.
Inclua prints no seu relatório!

Data de entrega: ~~23:59 do dia anterior à próxima aula;~~ 10/04/2024 - 11h59

Data de apresentação: Próxima aula.

DÚVIDAS?

REFERÊNCIAS

- Infanti, Luciano. **Uma ampla lista de métodos e entregáveis de UX design.** Acesso em: 29/03/2024. Disponível em: <https://brasil.uxdesign.cc/uma-ampla-lista-de-m%C3%A9todos-e-entreg%C3%A1veis-de-ux-design-7b83a859d234>
- Moran, Kate. Benchmarking UX: Tracking Metrics. Acesso em: 30/03/2024. Disponível em: <https://www.nngroup.com/articles/benchmarking-ux/>
- Joyce, Alita. 7 Steps to Benchmark Your Product's UX. Acesso em: 30/03/2024. Disponível em: <https://www.nngroup.com/articles/product-ux-benchmarks/>
- PRESSMAN, Roger S. Engenharia de software uma abordagem profissional. 9ª ed. Porto Alegre: AMGH, 2021.
- Holtzblatt, K., & Beyer, H. (1993). Making customer-centered design work for teams. Communications of the ACM, 36(10), 92-103.
- CHRISTEL, Michael G.; KANG, Kyo C. Issues in requirements elicitation. 1992.