



Desenvolvimento de Sistemas

Processo de *design*

Quando um desenvolvedor de sistemas inicia um projeto de *software*, ele deverá, em algum momento, pensar na interação do usuário com o seu produto, a qual ocorre por meio de uma interface gráfica. A interface gráfica, por sua vez, é desenvolvida com atenção aos aspectos da usabilidade, da beleza e da praticidade, e é o que se chama de *design*. Logo, um desenvolvedor de sistemas precisa ser um pouco *designer* também, para entender e garantir que o seu *software* terá boa aceitação e usabilidade pelo seu público-alvo.



Nem sempre o desenvolvedor de *software* trabalhará sozinho em um projeto de interface e *design*. Por isso, é importante que ele entenda o processo e as etapas que o nortearão e o ajudarão a cumprir seu objetivo.

Mas o que é um processo de *design*? Trata-se de um modelo mental que os desenvolvedores de sistema encarregados do *design* do *software* utilizam para resolver problemas com foco no usuário.

Sempre que se fala em abordagem de *design* centrado em usuário, fala-se em uma atividade criativa que considera os problemas que o usuário pode encontrar e as soluções a serem implementadas para resolver esses problemas, sendo essas soluções viáveis e/ou desejáveis.



Etapas

Imersão ou entendimento

Na primeira etapa de criação de interface para um projeto de desenvolvimento de *software*, o programador pode participar do processo criativo de criação dessa interface. Para isso, ele precisa entender todos os elementos do problema a ser resolvido e conhecer o público-alvo do aplicativo, assim como quais são as soluções para o problema em questão, pois já existem no mercado aplicativos que cumprem funções parecidas com o que foi proposto ou até iguais a ele.

Nessa etapa, o desenvolvedor deve focar o tema do projeto, entendendo as situações e as particularidades do problema. Nesse caso, o problema poderia ser o *design* de um aplicativo para celular, para *desktop* ou até mesmo para um *website*.

A primeira etapa pode ainda ser dividida em outras duas partes, que são:

Imersão preliminar

Na imersão preliminar, o desenvolvedor de sistemas e/ou a equipe de desenvolvedores de sistema encarregados do *design* do *software* pensam nos problemas a serem resolvidos e na direção que o projeto tomará para resolvê-los.

Imersão profunda

Na imersão profunda, a visão sobre o problema deve ser mais detalhada e as soluções mais específicas. Podem-se gerar todos os pontos de partida que ajudarão a desenvolver toda a construção do projeto.

Essas imersões podem ocorrer de várias maneiras, entre as quais estão as seguintes:

- ◆ *Brainstorm*
- ◆ *E-mails*
- ◆ Reuniões
- ◆ Entrevistas com o público-alvo

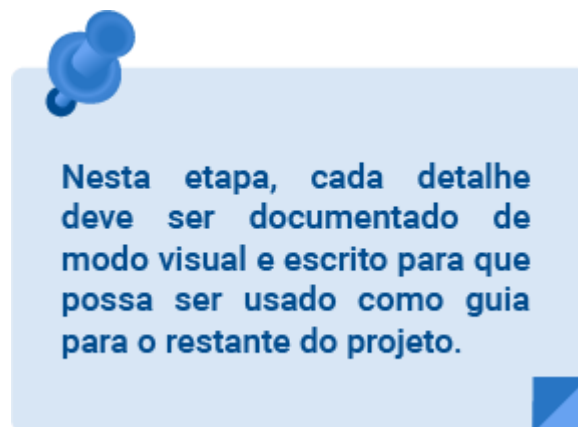
As imersões podem acontecer também com equipes de tamanhos variados; inclusive desenvolvedores que trabalham sozinhos em projetos pessoais podem beneficiar-se dessas etapas para sua criação. Por exemplo, um desenvolvedor solo pode utilizar grupos de discussão na Internet.



Análise e síntese

Depois de conhecer todos os detalhes do problema, o seu escopo, o seu público e quais possibilidades estão disponíveis, chega a hora de analisar esses dados e criar um plano de execução e síntese para desenvolvimento desta solução.

É nesta etapa que são determinadas as ações e as pessoas que desenvolverão a solução para o problema apresentado.



Ideação

A ideação é a etapa na qual o *brainstorm* acontece, pois agora o problema já foi entendido, existe análise e síntese dele e tudo está coletado e documentado. Resta então colher ideias com a equipe para a criação da solução, considerando sempre o seu público-alvo.

Nessa etapa, todos os envolvidos no projeto podem, e devem, apresentar ideias e sugerir propostas que podem intervir em ideias já existentes. Isso é essencial, pois, quanto mais pessoas se envolverem, mais posições e ideias diferentes sobre a solução do problema surgirão e poderão ser estudadas e incluídas no projeto.



Figura 1 – *Brainstorm* de ideias com participação de toda uma equipe
Fonte: Arena Marcas e Patentes (s. d.)

Prototipagem

Após as três primeiras etapas, é na prototipagem que as ideias se transformam em soluções reais para o problema. A criação de protótipos permite que a equipe possa testar as ideias junto a ela mesma ou ao público-alvo.



Uma boa estratégia é sempre fazer a seguinte reflexão: “Qual é o mínimo necessário para resolver este problema de modo eficiente e prático?”.



Figura 2 – Exemplo de um protótipo de aplicativo móvel

Fonte: Dantas (2018)



Abordagens centradas no usuário, empatia e descoberta

Abordagens de *design* centradas no usuário, do inglês *user centered design* (UCD), é uma abordagem de criação de *design* que tem como centro das decisões de solução o usuário. O objetivo disso é garantir que, durante todo o processo de criação do *design*, o foco permaneça no usuário-alvo.

Seres humanos criam *design* e seres humanos usam esse *design*.

Considerando isso, é possível afirmar que essa metodologia contém quatro características principais, podendo ser:

- 1 Empática: usa empatia para focar nas necessidades do usuário alvo.
- 2 Colaborativa: envolve o usuário no processo de desenvolvimento.
- 3 Otimista: sempre acredita que uma mudança para melhor é possível.
- 4 Experimental: valoriza a prática como principal ferramenta de aprendizado.

O UCD é muito utilizado quando se precisa converter usuários casuais em usuários fixos de um produto.

Muitos aplicativos para *desktop*, *sites* e aplicativos móveis utilizam essa metodologia para criar vínculo com os seus usuários por meio do *software*. Resumindo, **UCD refere-se à construção dessa empatia com o público-alvo**, com uma equipe pensando sempre que pode melhorar a interação homem-máquina, criando protótipos e experimentando ideias inovadoras com foco nessa interação.

A importância do UCD já cresce em nível exponencial há algum tempo, o que dificultou muito a distinção entre experiência física e experiência digital, dado o quanto o digital está inserido no mundo físico. Com a ajuda do UCD, essa mudança de paradigmas da relação homem/máquina foi bastante espontânea.

Empresas que criam soluções utilizando essas técnicas conseguem fornecer resultados reais e mensuráveis e acabam tendo vantagem competitiva diferenciada, tornando-se, com isso, menos penoso transformar dificuldades em oportunidades.



Os **10 princípios do UCD** podem ser definidos como:

1. Compreensão das necessidades do público

Se você pensar bem, o usuário nunca está interagindo diretamente com o *hardware* e sim com a sua marca e/ou produto. É disso que ele lembrará e é a isso que ele associará sua solução.

Para auxiliar na criação de uma interface centrada no usuário, pode-se refletir sobre algumas questões, tais como “quem é o usuário?”, “por que ele está usando sua solução?”, “qual é a maneira mais intuitiva de ele fazer isso?”.

2. Teste de usabilidade

Nada deve ser lançado antes de ser testado. Portanto, é sempre bom se colocar no lugar do usuário para entender os problemas que ele enfrentará ao usar sua interface. Pessoas do público-alvo podem ser expostas à interface para o colhimento de *feedback*. Pessoas reais e *feedbacks* reais são sempre o melhor.

3. Boa usabilidade

Qual é o principal trunfo da empresa Apple na criação de seus produtos?


 Ela torna tudo simples para o seu usuário final. Em seus produtos, tudo é fácil de usar, basta arrastar, clicar, apertar e soltar. A Apple não cria nada sem antes considerar o usuário final, e essa filosofia sem dúvida é um trunfo da marca. Uma interface fácil e amigável aumenta muito a retenção do usuário.

Figura 3 – Sistema operacional móvel da Apple chamado de iOS, fácil e amigável a todos os usuários

Fonte: Furquim (2021)

A imagem mostra uma mão humana segurando um iPhone. O aparelho está com a tela ativa apresentando a tela inicial do iOS 15 com seus ícones.

4. Consistência

Quanto mais consistência o seu *design* tiver, mais fácil será o usuário manter-se orientado espacialmente. Com consistência, é possível unificar a maneira com a qual o usuário consome o seu *design*.

Por exemplo, usuários cujos idiomas são escritos da esquerda para a direita tendem a navegar entre telas nessa mesma direção; já usuários de idiomas como o árabe e o persa, que são escritos da direita para esquerda, tendem a navegar de maneira contrária.

Figura 4 – Exemplos de telas de uma aplicação móvel, em que a orientação está disposta para a escrita árabe (da direita para a esquerda)

Fonte: Marketania (c2007-2019)

A imagem mostra variadas telas de aplicativos em árabe: seus menus e suas imagens encontram-se à direita da tela.

5. Diálogo simples e natural

Todo usuário entende a função do botão **home** em um aplicativo, mas menus podem se tornar confusos rapidamente. Um bom exemplo de diálogo simples e natural é quando um usuário preenche um formulário e clica em **Enviar**. Caso ele não receba um *feedback* que o envio ocorreu satisfatoriamente, pode ficar confuso quanto à funcionalidade do *software*.

6. *Feedback* adequado

Mensagens de agradecimento e de confirmação de envio e leitura são uma ótima maneira de mandar um *feedback* para o seu usuário. Considerando o exemplo do diálogo simples citado, um aviso na tela de que o formulário foi recebido pelo aplicativo é uma ótima maneira de apresentar um *feedback* em que o usuário pode confiar. Outro exemplo é quando uma ação no seu aplicativo leva algum tempo para ser executada. Nesse caso, uma barra de progresso, uma ampulheta ou até mesmo um círculo com uma flechinha indicando atividade é uma ótima maneira de apresentar *feedback* ao usuário.

7. Redução do esforço mental do usuário

Qualquer dificuldade de interação ou de procura do usuário com o seu aplicativo ou sistema pode causar frustração e abandono. Lembre-se de que o usuário gosta de acessar o que precisa, sem ter que pensar ou esforçar-se minimamente. Quanto mais fácil e simples, melhor.

8. Garantia de confiança

Um bom *design* é, acima de tudo, honesto. A clareza na forma de se comunicar com o usuário e as intenções claras do aplicativo para com ele são fatores significativos na retenção e satisfação do usuário. Um *site* que tenta

“empurrar” uma venda extra pode ser visto com maus olhos e o usuário pode perder a confiança nele.

9. Lei de Hick

A lei de Hick afirma que o excesso de ícones e menus torna a navegação entre eles mais lenta, tediosa e frustrante. Ou seja, exagerar na quantidade de informação pode estar atrasando a busca do usuário pelo objetivo.

10. Lei de Fitt

Já na lei de Fitt, determina-se o tempo que o usuário leva para navegar entre conteúdos e ícones. Quanto maior a distância, menor será a precisão. Um bom exemplo é a tela de um celular, pois nela estão pontos de difícil acesso aos polegares do usuário.



Principais recursos de um projeto que usa UCD



Persona

A persona é um personagem fictício usado para ilustrar o que seria o usuário de verdade, com os dados de verdade de um deles. A persona reúne as características reais de usuários reais e que são definidas nas fases de pesquisa e de planejamento da criação da solução.



Cenário

Nesse recurso, tem-se como objetivo construir cenários fictícios de uso da solução, baseados em eventos reais e situações reais de uso dessa solução. Assim, a persona é parte fundamental da criação de tais cenários.



Mapa da jornada do consumidor/usuário

O mapa da jornada do consumidor/usuário reúne os documentos que registram todas as ações do usuário ao efetivamente usar a aplicação. Com esses documentos, é possível inclusive utilizar aprendizagem de máquina para se estudar melhorias no design do projeto.



Encerramento



FINALIZANDO



Você aprendeu, portanto, que a relação entre usuário e máquina ocorre por meio de um *design* de interface. Além disso, percebeu também que o uso de um *design* centrado no usuário é de fundamental importância para a retenção e a satisfação desse usuário.