UniFil
EDUCAÇÃO A DISTÂNCIA

Unidade 4

Design e avaliação de usabilidade

Abertura

Olá, amigo(a) discente! Seja bem-vindo(a)!

Imagino e espero que você esteja ansioso pelo conteúdo dessa aula, que é, finalmente, o design de interação e de interfaces, a atividade fim dessa disciplina. Nessa aula você vai aprender técnicas que ajudam na sistematização do design desses artefatos extremamente complexos. Começaremos elaborando abstrações que nos ajudam a organizar as informações trocadas entre usuário e sistema, e a ordenar os momentos em que cada troca ocorre. Em seguida, definiremos as cenas e as transições entre elas. Finalmente, estudaremos sobre os variados tipos de interfaces, os componentes existentes, a padronização estilística e de plataforma e a elaboração de esboços e protótipos. Encerraremos o ciclo do processo de design de sistemas interativos abordando três técnicas de avaliação de usabilidade, que são preciosas como faróis em uma viagem em meio a uma neblina densa. Bons estudos!

Objetivos

- Aprender a elaborar o design de interação conceitual;
- Conhecer os tipos de interfaces, os componentes widgets e os padrões estilísticos e comportamentais de plataformas de sistemas;
- Aprender as formas de elaborar representações de interfaces para antecipação do design e para construção do sistema interativo;
- Conhecer as principais técnicas de avaliação de usabilidade, e o motivo de serem necessárias.

Conteúdo Programático

Aula 01 – Design de interação

Aula 02 – Avaliação de usabilidade

Quer assistir às videoaulas em seu celular? Basta apontar a câmera para os **QR Codes** distribuídos neste conteúdo.

Caso necessário, instale um aplicativo de leitura QR Code no celular e efetue o login na sua conta Gmail.

Design de interação

Na aula anterior expliquei a importância de elaborarmos modelos e representações de nossos objetos de trabalho, que são as pessoas, suas atividades, os contextos onde ocorrem e as tecnologias envolvidas ou disponíveis. Abordei algumas das principais técnicas que permite expressar nosso entendimento, compartilhando-o com todos os envolvidos, e, principalmente, melhorando-o iterativamente e aumentando sua fidedignidade para nós mesmos.

A partir do momento em que nós começamos a compreender o domínio do sistema interativo a ser trabalhado, é necessário iniciar as atividades de design de interação, que consiste na elaboração de modelos conceituais de entidades e atributos do sistema, na estruturação das tarefas dos usuários a serem apoiadas pelo sistema, e na projeção da interface do sistema interativo.

Em um mundo ideal, essas atividades acontecem nessa exata ordem, mas estamos tratando de uma atividade holística, em que a perfeição é inatingível, e não há demarcações exatas para nos dizer que uma etapa está completamente correta e, portanto, terminada. Por esse motivo, é necessário começá-las o quanto antes e dar-lhes tempo suficiente para que se possa praticar a antecipação e o design participativo. Antecipação é a elaboração de designs ainda antes de se ter todo o conhecimento e especificação necessários sobre o domínio. Já o design participativo é a inclusão de usuários finais no processo de design, testando, avaliando e sugerindo melhorias nos modelos elaborados.

Design Conceitual

O design conceitual é a elaboração de modelos representativos das atividades e dos objetos envolvidos nos objetivos dos usuários que serão apoiados pelo sistema interativo. Esses modelos começam em um nível de abstração quase independente de meios e tecnologias disponíveis e acessíveis. À medida que são refinados e o processo de design evolui, eles são transformados em modelos específicos e representações concretas, tornando-se um conjunto de especificações de software, a ser utilizado pela equipe de engenharia e desenvolvimento do sistema.

Cenário de interação

Voltando à aula anterior, abordei a técnica da escrita de cenários, com o objetivo de descrever a situação atual dos envolvidos na realização de suas atividades. Alguns autores, como Barbosa e Silva (2010), chamam esse tipo de cenário de “cenários de problema”, enquanto outros, como Benyon (2011), o chamam de *user stories*, que não deve ser, por sua vez, confundido com as *user stories* típicas das metodologias ágeis da engenharia de software. Como você deve notar, a nomenclatura de técnicas no desenvolvimento de sistemas é, por vezes, confusa, principalmente por se tratar de uma área de estudos e atuação bastante recente.

Isto posto, prossigo com a primeira etapa do design conceitual, que é a escrita de cenários. Porém, neste caso, são “cenários de interação” ou “cenários conceituais”, dependendo do autor de seu livro de referência, que são escritos exatamente como os cenários de problema, mas com contexto e atividades fictícias, representando aquelas que imaginamos envolverem a utilização do sistema que estamos projetando para nossos usuários. Além disso, devemos

evitar entrar em detalhes de interface do sistema, pois mais adiante utilizaremos outras técnicas para esse tipo de especificação.

A melhor maneira de escrevermos cenários de interação é justamente partir dos cenários de problema. Tipicamente, você terá escrito vários cenários que ilustram a realidade dos usuários, de forma que há uma certa sobreposição nas atividades relatadas entre cenários. Além disso, é difícil não imaginar como as atividades podem ser aperfeiçoadas, ou até mesmo reprojetadas completamente, após a leitura. Dessa forma, uma coleção de cenários de problema provavelmente dará origem a uma coleção de cenários de interação menor, reduzindo e agregando atividades necessárias para o cumprimento dos objetivos.

A fim de contextualização, volte à aula anterior e leia novamente o cenário intitulado "Aprendizagem de uma estratégia vencedora para a posição 'Mid' em *League of Legends*", pois no quadro 1 apresento um cenário de interação resultante dele, para ilustração do conceito.

Quadro 1: Um cenário de interação para o *LoLpendium*, correspondente ao cenário descrito na unidade anterior.

Objetivo: Aprendizagem de uma nova estratégia em *League of Legends*
Atores: Josias Fabre Sabala (jogador), Daniel Jerome César (líder de comunidade)
Josias deseja expandir sua variedade estratégica no LoL, pois seus amigos o trocaram de posição no jogo. Para tal, ele utiliza o *LoLpendium* para acessar a biblioteca de receitas estratégicas do Daniel, seu *influencer* e líder de comunidade preferido, e avalia suas estratégias mais novas e populares, que são específicas para a nova posição em que vai jogar. Lendo um resumo sucinto de vantagens, desvantagens e estilo de jogo, ele seleciona uma estratégia ganhadora baseada na personagem Akali. A seguir, o *LoLpendium* indica um vídeo gravado por Daniel, com uma apresentação, instrução e comentários sobre a estratégia escolhida, onde é explicado o passo-a-passo das habilidades a serem escolhidas e as armas a serem compradas e combinadas. Como o vídeo o convenceu de que a estratégia era promissora, Josias a adiciona em sua coleção de estratégias favoritas. Chegada a hora de jogar uma partida de LoL, Josias escolhe a *Akali* e jogará no meio de campo. Durante o intervalo entre as definições dos times e o início da partida (LoL sempre dá pelo menos um minuto nesse momento), Josias o *LoLpendium* em seu dispositivo móvel preferido e utiliza um atalho para escolher a estratégia que marcou como favorita recentemente. Nela, ele aciona a função de instrução de jogo em tempo real, similar a um navegador por GPS para carros. Seu dispositivo móvel é colocado sobre sua mesa, próximo ao teclado do computador. Assim que a partida começa, Josias marca o início

no *LoLpendium*, que, a partir de então, vai instruindo-o nos itens a serem comprados, habilidades a serem adquiridas, nos momentos certos, e dando-lhe dicas escritas por Daniel para serem lembradas no decorrer da partida. Dessa forma, Josias não precisa tirar sua atenção da tela de jogo.

Fonte: o próprio autor.

Representação do cenário como conversa

Após escrito e aprovado, cada cenário de interação pode ser analisado e esquematizado de forma a ficar claro quais são as atividades envolvidas, as ações realizadas pelos usuários, os objetos e unidades de informação manipulados, e o papel do sistema. Simone e Silva (2010) sugerem, em sua metodologia do design centrado na comunicação, tratar a esquematização como um diálogo entre os usuários e um prestador de serviços, enquanto Benyon (2011) apresenta diversas técnicas similares, mas sem nenhuma doutrina específica. Por questão de maior familiaridade, vamos seguir com as técnicas especificadas pelos dois primeiros.

Nessa técnica, representamos o cenário como uma conversa, com tópicos, subtópicos, falas e signos. Os tópicos são as atividades enquanto subtópicos são ações que compõem as atividades. As falas são projeções de interação entre o usuário (U) e o sistema (D) – aqui imaginado como um prestador de serviços – enquanto signos são os objetos e as unidades de informação trocadas entre eles. No quadro 2, apresento o cenário do quadro representado em forma de conversa.

Quadro 2: Cenário de interação do *LoLpendium* representado como conversa.

Tópico > Subtópico	Falas e signos
Encontrar uma estratégia nova	U: Oi! Preciso de uma nova **estratégia** de LoL.
> Analisar estratégias disponíveis	D: Aqui estão as estratégias **mais novas e populares** de seus **canais preferidos**, com o **estilo de jogo**, **prós e contras**.
> Refinar análise	U: Quero restringir as estratégias mostradas por **estilo de jogo** e **posição**. D: Aqui estão.

> Aprender sobre uma estratégia	U: Quero aprender sobre essa estratégia em mais detalhes D: Aqui está seu **vídeo de apresentação** e **o plano**, com a **lista de ações** e **tempos**.
> Escolher estratégia	U: Quero adicionar essa estratégia à minha **coleção pessoal**. D: Ok, ela está adicionada.
Jogar LoL com guia estratégico	U: Agora, quero ser guiado em uma estratégia durante uma **partida de LoL**.
> Encontrar estratégia pré-marcada	D: Em qual posição você jogará? U: Esta. D: Aqui estão suas estratégias relacionadas a esta posição.
> Escolher estratégia para o guia	U: Quero utilizar esta estratégia D: Certo. Me avise quando a partida começar.
> Revisar plano estratégico	D: Enquanto a partida não começa, você pode revisar os **principais momentos** da estratégia escolhida, com os referidos **tempos de ação**.
> Indicar início de jogo	U: O jogo começou. D: Ok, vou iniciar meu **cronômetro**
> Obter instruções em tempo real	D: Hora de fazer essa ação! U: Feita! D: Cuidado! Você está **45 segundos atrasado** em **relação ao plano**.

Fonte: o próprio autor.

Nesse esquema, os elementos mais complicados de modelar me parecem ser os signos, que estão marcados em negrito. De maneira geral, você deve elaborar falas que contenham os detalhes informacionais mínimos para que a conversa faça sentido, mas ao mesmo tempo preocupando-se em não torná-la prolixa. Em geral, os signos são os substantivos das orações, e não se preocupe em marcar um mesmo signo duas vezes. Por fim, para validação, deve ser possível ler toda a conversa, sem fazer saltos, e que ela permaneça fazendo sentido.

A esquematização dos cenários em forma de conversas nos provê informações ricas e estruturadas para continuarmos a elaborar o design conceitual. Os signos identificados e destacados serão nossa fonte principal para elaboração do esquema entidade-relacionamento, diminuindo o risco de perdermos, esquecermos ou deixarmos de considerar informações importantes no cumprimento dos objetivos dos usuários. Paralelamente, os tópicos e subtópicos demarcam e ordenam os momentos de ação e comunicação entre o usuário e o sistema, e serão fonte para o design do mapa de transições entre cenas de interação.

Videoaula 1

Utilize o QR Code para assistir!

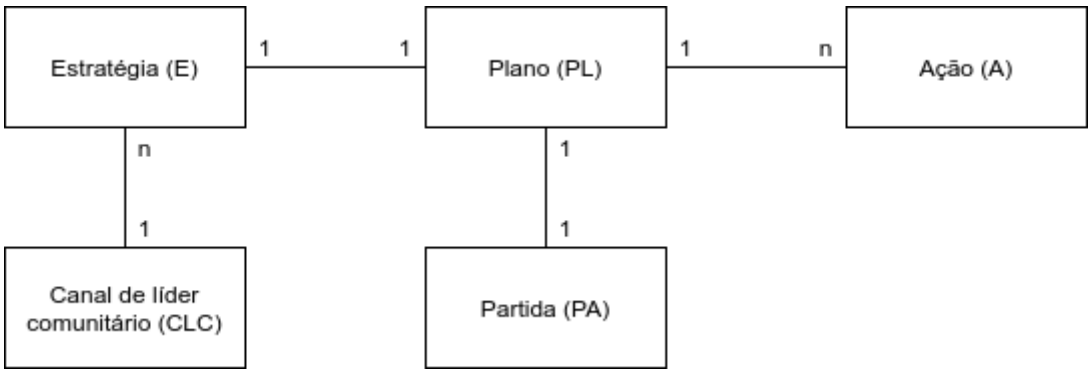
Agora, assista ao vídeo em que abordo a elaboração de cenários de interação e esquematização em conversa.

Esquemas entidade-relacionamento de signos

Esquemas e diagramas entidade-relacionamento são bastante comuns no desenvolvimento de sistemas de informação. Caso você já tenha estudado disciplinas como Engenharia de Software, Análise de Sistemas, Bancos de Dados, Programação Orientada a Objetos ou Teoria das Categorias, certamente já encontrou esse tipo de representação de dados. E na atividade de design de sistemas interativos, você a encontrará mais uma vez, apesar de ser menos protocolar e sistemática neste caso, como é com quase toda ferramenta utilizada em Interação Humano Computador.

Um esquema entidade-relacionamento (ER) tem como propósito a elucidação e definição dos objetos de informação que são manipulados na atividade ou processo que se está projetado, dando-lhes nome e descrevendo as relações conjuntivas entre eles. A fim de exemplificação, considere a figura 1, que traz uma modelagem ER a partir da conversa do quadro 2.

Figura 1: Esquema entidade-relacionamento que modela a conversa do quadro 2.

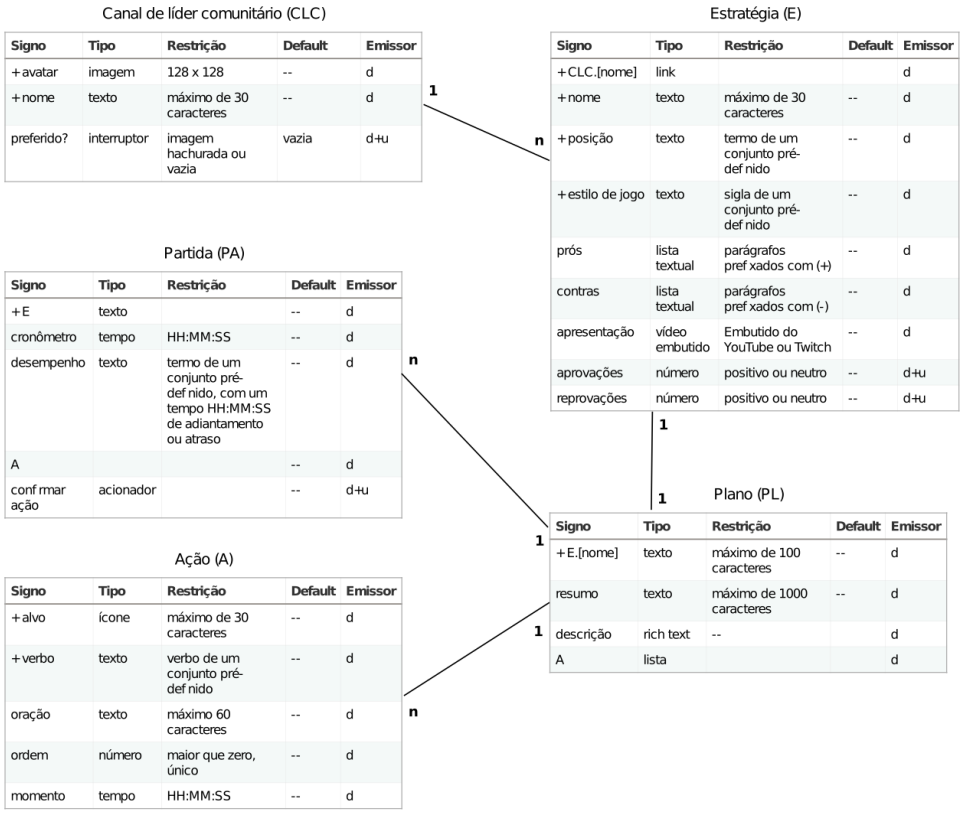


Fonte: o próprio autor

Este diagrama da figura 1 informa que há cinco unidades de informação chamadas “canal de líder comunitário” (CLC), “estratégia” (E), “plano” (PL), “ação” (A) e “partida” (PA). Ele também nos diz que um canal de líder comunitário único está relacionado a múltiplas, ou *n*, estratégias. Cada estratégia, por sua vez, está relacionada a um único plano, e vice-versa. Um plano único está relacionado também a uma partida e a múltiplas ações.

A elaboração desse tipo de diagrama não requer necessariamente o esquema da conversa, mas tê-lo torna esse processo de design mais fácil e assertivo, quase sistemático. No caso desse ER da figura 1, os nomes foram definidos a partir dos signos identificados na conversa do quadro 2. Em uma interpretação pessoal, modelei que alguns dos signos se referem a objetos de informação, ou seja, um conjunto de atributos coesos e com uma relação de pertinência entre si. Já os outros signos, os que não estão nesse ER, deverão constar como atributos desses objetos, o que nos leva à forma detalhada do ER, exposto na figura 2.

Figura 2: Esquema entidade-relacionamento com atributos detalhados



Fonte: o próprio autor.

Neste ER detalhado, já nos forçamos a definir muitos detalhes importantes do design da interface. Os atributos de cada entidade não apenas foram definidos e nomeados, mas também qualificados com tipo (ou forma), restrição de representação, forma padrão e emissor. Este último pode ser “d” ou “d+u”, que significam, respectivamente, que o atributo é definido pelo sistema ou pelo usuário. Neste caso, como nosso ator principal deste cenário é um jogador, que tem um papel mais predominantemente de consumidor de informação, a maioria desses atributos são tipo “d”, emitidos pelo sistema. Caso estivéssemos modelando um cenário do líder de comunidade, predominantemente produtor de conteúdo, decerto estes atributos seriam “d+u” em sua maioria.

Outra notação que merece destaque é o sinal de mais (+) que precede alguns dos nomes de atributos, e indicam que estes atributos são identificadores únicos para os usuários, permitindo-os distinguir entre as diferentes instâncias de uma mesma entidade. Para ilustrar o conceito, considere a entidade “canal de líder comunitário”, que tem os atributos avatar e nome como chaves identificadoras. Neste caso, de acordo com o modelo do designer, um usuário identifica um canal de líder comunitário olhando sua imagem de avatar e seu nome. Para quem já estudou Bancos de Dados, a ideia é similar à da chave primária de tabelas, mas essas são sempre um código

numérico, o tipo de dados preferido para computadores, enquanto nós, seres humanos, preferimos atributos visuais e sonoros.

Este modelo de ER que apresentei é o utilizado por Barbosa e Silva (2010), mas outros autores utilizam esquematizações ER similares, mas com notações variadas. Diferentemente do que acontece na modelagem com as técnicas diagramáticas da engenharia de software, minha recomendação é que você não se prenda aos detalhes protocolares e sinta-se à vontade em inserir novas colunas ou elementos que o permitam se expressar melhor, que é o objetivo final de nosso design conceitual. Os próprios Barbosa e Silva (2010) propõem várias outras colunas, que você pode aprender a respeito lendo o material original, enquanto Benyon (2011) propõe modelos muito mais simples, apenas denotando o diagrama ER, e os nomes dos atributos.

Videoaula 2

Utilize o QR Code para assistir!

Agora, assista ao vídeo em que abordo a elaboração de diagramas entidade-relacionamento de interação.

Modelagem da Interação

Além de nos servir como fonte rica de informação para o design de esquemas ER, a representação de cenários como conversa nos serve também como ponto de partida para o design da modelagem de interação, que trata da definição das cenas de nossa interface de usuário. Uma cena de interação é um estado ou variação da interface do sistema projetado para permitir ao usuário o cumprimento de um objetivo específico, e a realização de várias ações de transição a outras cenas.

Para ajudar em seu entendimento, pense no *iFood*, famoso aplicativo de encomenda de refeições. Quando se inicia o programa, ele logo mostra uma tela com o objetivo de identificação, ou *log-in*. Esta é uma cena. A partir dela, o usuário pode criar uma conta nova, caso não tenha, tentar recuperar sua senha perdida, ou identificar-se. Cada uma dessas ações leva o usuário para uma outra tela distinta, com objetivos específicos. E cada uma delas igualmente oferecerá ações de transição para outras telas com novos objetivos. Neste caso do *iFood*, cada tela corresponde a uma cena de interação.

Por seu tamanho reduzido e pela baixa precisão na captura dos gestos dos usuários, os aplicativos de smartphone costumam ter uma tela de interface específica para uma cena de interação, da modelagem de interação. Mas esse mapeamento direto entre

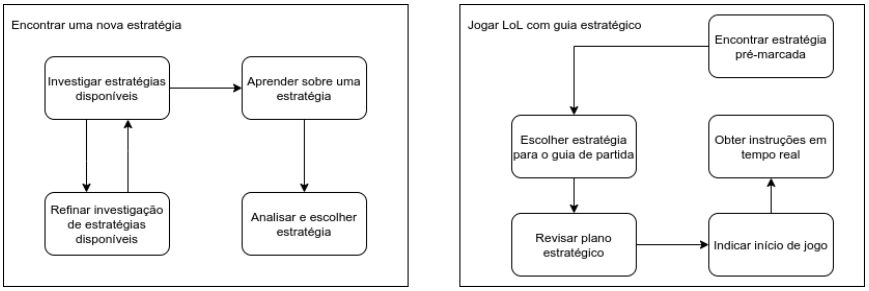
tela e cena não é obrigatório, e uma interface pode oferecer uma tela, janela, ou modo de operação, que implemente múltiplas cenas de uma vez. Porém é mais fácil, a princípio, imaginar que uma cena equivale a uma tela de uma aplicação de interface gráfica.

Há várias linguagens desenvolvidas por diversos autores para a modelagem de interação, algumas delas até mesmo padronizadas e protocolizadas por reconhecidas instituições de fomento e prática de engenharia de software. Há a rede de transição de estado (STN), proposta por Benyon (2011), a linguagem de modelagem de fluxograma de interfaces (IFML), padronizada pela Object Management Group (OMG), famosa fomentadora da UML, a linguagem de modelagem da interação como conversa (MoLIC), proposta por Barbosa e Silva (2010), e muitas outras, como uma rápida pesquisa pela literatura revelará. Mais uma vez, por familiaridade minha, prosseguiremos com a MoLIC. Apenas me sinto no dever de aclarar que não há um padrão vencedor no mercado profissional. Na maioria dos projetos nem mesmo um padrão é seguido em sua completude, exceto quando se trata de licitações públicas ou projetos para empresas auditadas, quando então é necessário utilizar a linguagem determinada pelas regras ou editais.

Feitas as considerações sobre a escolha da linguagem de modelagem da interação, prossigo com as instruções. A primeira etapa dessa atividade é elencar todos os tipos de usuários, ou papéis de usuários como é mais comumente denominado na engenharia de software, e, para cada um deles, elencar todos os seus objetivos. Na sequência, é necessário relacionar os objetivos por afinidade de objetos de informação, definir para cada objetivo quais são os outros que precisam ser cumpridos antes para que se tenha informação o suficiente no sistema de forma que aquele seja viável.

Convenientemente, a esquematização do cenário em forma de conversa, que fizemos no quadro 2, justamente apresenta em sua coluna da direita, os tópicos e subtópicos, estes objetivos, e organizados razoavelmente em uma sequência de dependência, de forma que já temos mais da metade do trabalho feito. Tudo o que nos resta fazer agora é representá-lo em forma de um diagrama de estados, como mostra a figura 3.

Figura 3: Primeira etapa da modelagem de interação tipo MoLIC

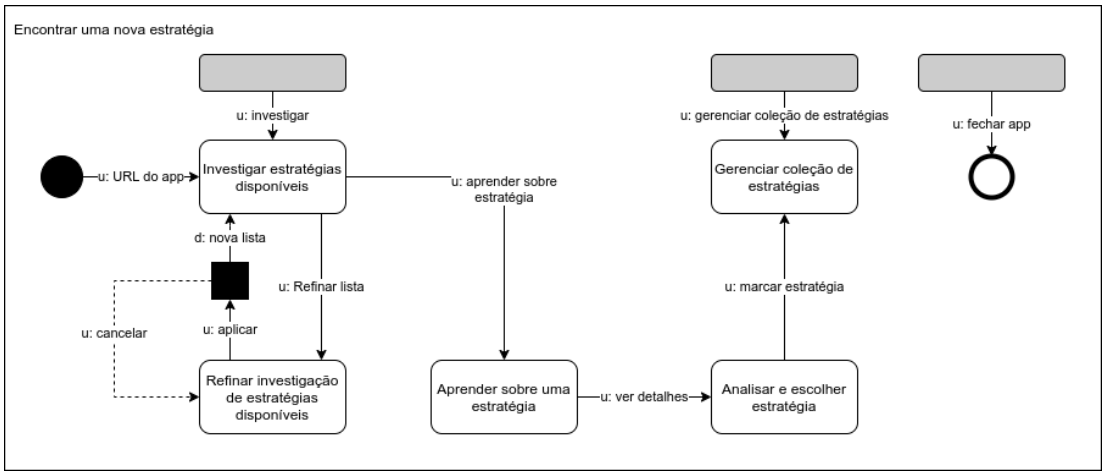


Fonte: o próprio autor.

Essa figura apresenta o MoLIC simples, que identifica as cenas distintas, representadas por cada retângulo com bordas arredondadas, as transições possíveis de uma cena às outras, e a segregação das cenas em interações distintas, neste caso as duas grandes caixas que agregam as cenas. Em um projeto completo, há grandes caixas dessas para cada combinação de tipos de usuário e contexto de uso. No *LoLpendium*, há pelo menos os usuários "jogadores" e "líderes de comunidade", sendo que os primeiros têm os contextos de uso de encontrar nova estratégia e jogar LoL com guia, enquanto os da segunda classe têm o contexto de uso de gerenciar estratégias publicadas.

Após a elaboração desse MoLIC simples, também denominado mapa de objetivos, podemos prosseguir em um trabalho iterativo de detalhamento das interações. A primeira melhoria a ser feita é a nomeação das ações que causam as transições, colocando um verbo ou oração em cada aresta do diagrama, as linhas com setas que relacionam as cenas. Também devemos indicar quais são os pontos de início e de fim da interação, colocando a bolinha anelada e a bolinha preenchida, respectivamente. Depois, vamos querer indicar as transições ubíquas, aquelas que podem acontecer a qualquer momento da interação, independente da cena a qual o usuário se encontra, com os retângulos hachurados. Finalmente, determinamos as transições que contém pausas de sistema, como tempo de computação do resultado de uma busca, e as transições de recuperação, que são aquelas que não fazem parte dos objetivos do usuário, mas são necessárias para oferecer escape ou suporte ao usuário. A figura 4 ilustra todos esses conceitos e linguagens, em uma evolução do quadro da esquerda da figura 3.

Figura 4



Fonte: o próprio autor.

Com a modelagem de interação completada, podemos prosseguir para o design de interfaces, tema de nossa próxima aula.

Videoaula 3

Utilize o QR Code para assistir!

Agora, assista ao vídeo em que abordo a elaboração da modelagem de interação com MoLIC.

Design de interfaces e avaliação de usabilidade

Design de interfaces é a atividade de definição dos recursos físicos dos sistemas, que permitem ao usuário expressar suas intenções e ao computador emitir chamados, instruções ou respostas. Em termos mais populares, é a atividade de definir janelas, botões, formulários, cores, estilo e atalhos.

Apesar de eu ter organizado esse material apresentando primeiro o design conceitual, e antes dele o próprio levantamento de necessidades do usuário, nada impede que o design de interfaces se inicie tão logo o projeto de sistema interativo seja também iniciado. Trata-se da prática de antecipação, que nos dá as vantagens de exercitar a conversa com os materiais e o design participativo, que envolve avaliar a usabilidade de um design de interfaces diretamente com o usuário final, e isso pode ser feito desde antes de termos sequer uma linha de código escrita. Eventualmente, o levantamento de necessidades e o design conceitual vão sendo desenvolvidos em paralelo para, no fim, realizar a síntese em um design de interfaces final, advindo de todas essas atividades.

Mas vamos focar agora especificamente no design de interfaces. Primeiramente, é importante conhecer os diversos estilos de interação que as tecnologias existentes de software e dispositivos físicos nos oferecem. Os estilos de interação mais utilizados são o da linguagens de comando, linguagem natural, interação por manipulação direta e o WIMP.

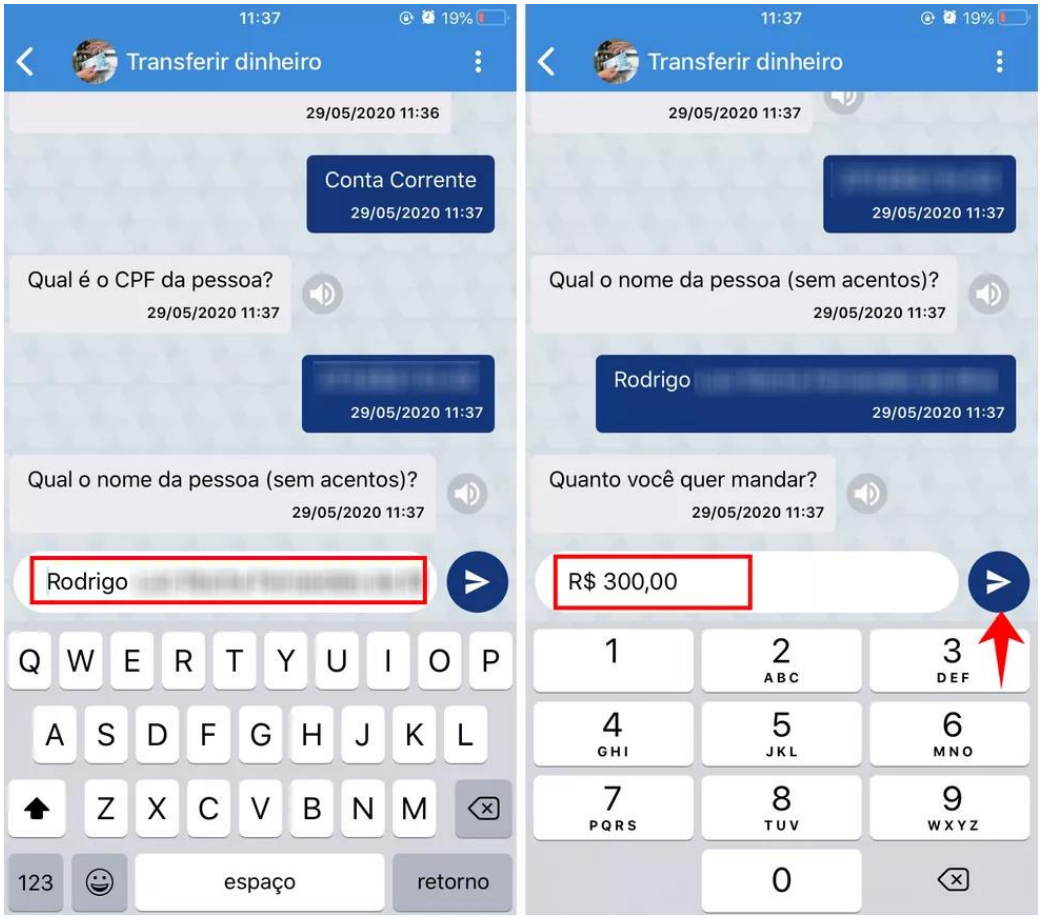
As interfaces de linguagens de comandos são aquelas mais ligadas à perspectiva de sistema, tipo de interação apresentado na segunda aula deste curso. São interfaces em que os usuários digitam comandos em linguagem de computador, como acontece em um terminal de linha de comando, uma linguagem de programação, edição direta de arquivos de configuração em texto e a utilização de atalhos, como o famoso copiar e colar. A utilização desse tipo de interface tipicamente redunda em sistemas com usabilidade mais eficiente, porém de mais difícil aprendizado e com menor segurança de uso. O desenvolvimento de interfaces de linguagens de comando é mais barato e fácil que a maioria dos outros tipos, por isso costuma ser a preferida pelos desenvolvedores. A figura 1 ilustra uma interface de linguagem de comando típica.

Figura 5: Exemplo de interface de linguagem de comando, neste caso, o IOS da CISCO

Fonte: http://geek-university.com/ccna/display-ios-command-history/. Acesso em: 21 mar. 2022.

As interfaces de linguagem natural são aquelas que correspondem à perspectiva do usuário, ou seja, o sistema tenta se expressar com a linguagem tipicamente humana. Esse tipo de interface é composta por mecanismos de fala computadorizada, interpretadores de comandos de voz, expressões faciais e textos em linguagem natural, e a conversa e escrita em linguagem natural. Exemplos de sistemas que utilizam esse tipo de interfaces são os copilotos GPS, como o Waze, os assistentes pessoais como a Alexa e a Siri, os robôs de chat de serviços de atendimento ao cliente, muito comuns nos bancos e operadoras telefônicas. A utilização de interfaces de linguagem natural provê, antes de mais nada, acessibilidade, já que a maioria das pessoas realmente não tem instrução, e talvez nem vocação, para entender a linguagem de comandos. O problema é que essas mesmas pessoas não entendem os limites da tecnologia e podem facilmente ficaram frustradas ou até mesmo nervosas quando o sistema emite uma resposta sem sentido em relação à última fala do usuário, ou ao contexto como um todo. Por isso, a maioria desses sistemas recorre a tiradas humorísticas genéricas e preveem a capacidade de um operador humano assumir a conversa tão logo ela se torne difícil para o computador.

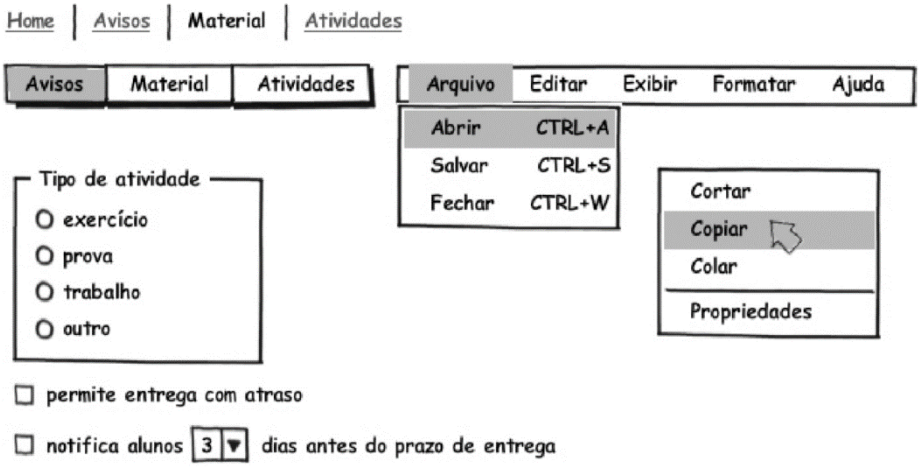
Figura 6: Interface de linguagem natural de *chatbot*, neste caso, do aplicativo Caixa Tem.



Fonte: https://ecminformatica01.blogspot.com/2020/06/como-transferir-dinheiro-do-caixa-tem.html.Acesso em: 21 mar. 2022.

As interfaces que utilizam a manipulação direta são aquelas que modelam os objetos virtuais como uma metáfora dos objetos reais, a fim de causar um *affordance* positivo generalizado, abrangendo até mesmo pessoas com pouca experiência prévia com computadores e sistemas interativos. Por exemplo, nossos sistemas são cheios de *menus* de opções, termo típico dos cardápios de restaurantes, que são acessíveis e consultados logo na entrada do cliente ao estabelecimento, para conhecer os pratos disponíveis, preços e tempo de espera. De maneira análoga, os menus dos sistemas oferecem as funcionalidades disponíveis de maneira organizada por categorias, e podem conter instruções extra como tempo de espera, pré-condições e necessidade de conversa com caixas de diálogo.

Figura 7: Elementos de interface gráfica do tipo menu, da manipulação direta.



Fonte: Barbosa e Silva (2010).

Essas caixas de diálogo são também uma forma de manipulação direta, trata-se da utilização de formulários, típicos no mundo real de quando contratamos serviços ou fazemos *check-in* em hotéis. Nos computadores, preenchemos formulários o tempo inteiro, desde quando pedimos ao navegador para que busque por uma página de internet, ou quando encomendamos um par de livros em uma livraria virtual, ou ainda quando configuramos nosso gerenciador de bancos de dados, como o DBeaver ou o DataGrip. Formulários são muito eficientes pois quase todo mundo já preencheu vários deles em sua vida, e em sua forma digital é ainda mais eficiente, com a utilização do teclado, mesmo que seja o virtual de dispositivos móveis, ou da dicção. Um cuidado importante com formulários é sempre indicar para o usuário a formatação da escrita, como ao indicar “dd/mm/aaaa” no caso de datas, respeitar a ordem típica esperada de entrada de dados e configurar corretamente a navegação automática entre os campos do formulário. Pouca coisa deixa um usuário mais furioso do que um formulário em que, ao apertar-se a tecla <tab>, o cursor do teclado vá para um campo de entrada do outro lado da tela, ou, pior, escolha um elemento de interface que nem da entrada de dados participa.

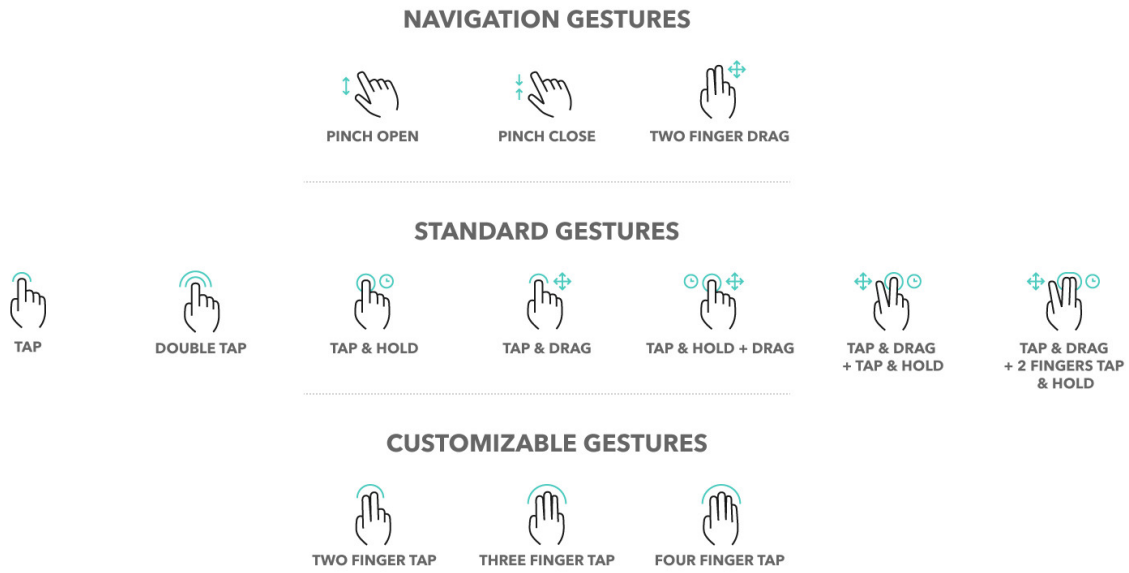
Ainda sobre as interfaces que se utilizam da manipulação direta, há os mecanismos de interpretação de gestos humanos, tipicamente manuais, mas ultimamente envolvem também as expressões faciais. A interface de gestos mais comum em sistemas de informação é o arrastar-e-soltar com o *mouse* de computador, que trata-se de uma metáfora para algo que hoje já está bem raro, que é abrir uma gaveta de arquivo físico, segurar e puxar um maço de papéis e colocá-los em outra gaveta, em outro separador.

O ponteiro do *mouse*, na interface gráfica, é como nossa mão se deslocando dentro do sistema, e quando seguramos o botão do *mouse*, o ponteiro muda seu ícone para o de uma mão fechada. Pelo fato de a analogia ser clara, simples e popular, quase todo mundo consegue mexer em um computador e arrastar arquivos depois de ver uma outra pessoa fazendo igual. Esse é o poder da acessibilidade de bons *affordances*, advindo de analogias completas. O problema surge quando a interface extrapola a analogia, como é o caso de se poder fazer uma cópia de qualquer arquivo ao arrastar-e-soltá-lo segurando junto a tecla <ctrl>.

Atualmente, as interfaces do tipo manipulação direta mais comuns e populares são as de gestos manuais, aquelas que utilizamos diariamente em nossos *smartphones* e *tablets*. Por

causa da alta capacidade computacional desses dispositivos e pela precisão mais que razoável das superfícies sensíveis a múltiplos toques, desde o lançamento do iPhone em 2007, a indústria de design de sistemas interativos desenvolveu e sedimentou uma linguagem de gestos manuais bastante natural para nós, seres humanos. A figura 4 ilustra os principais gestos, mas vale ressaltar que alguns deles acabam não tendo nenhuma analogia com o mundo real, e sofrem de problemas de acessibilidade e usabilidade, como é o caso do comando “arrastar com três dedos para trocar de aplicação”.

Figura 8: Ilustração de comandos de gestos, para interfaces de manipulação direta por meio de superfícies sensíveis a múltiplos toques.



Fonte: https://www.graphic.com/docs-ipad/gestures. Acesso em: 21 mar. 2022.

Por fim, há de se destacar que a maioria dos sistemas interativos utiliza uma composição de todos estes estilos de interfaces, em uma tentativa de agregar as qualidades que cada uma delas provê, e oferecer caminhos alternativos para usuários que encontrem dificuldades na manipulação de alguma das modalidades específicas. Sistemas interativos que usam essa confluência de estilos de interfaces são denominados sistemas com interface WIMP, da sigla *Windows, Icons, Menus and Pointers*.

Videoaula 1

Utilize o QR Code para assistir!

Agora, assista ao vídeo em que abordo os tipos de interfaces.

Padrões de interfaces

Vimos, na aula 04, que o design de interação não se beneficia muito com soluções pré-documentadas e reaproveitadas, porém, neste caso das interfaces, há a grande dificuldade de construção de elementos de interface arbitrários. O desenvolvimento de um novo objeto gráfico capaz de receber a entrada do usuário e dar-lhe resposta de forma correta, eficiente e compreensível, é, por si só, um projeto de software à parte. Portanto, dada a complexidade de construção de interfaces WIMP, a engenharia de software, em comunhão com o design de interação, produziu técnicas de estilização, padronização, documentação e reaproveitamento de elementos de interfaces, chamados de widgets.

Pense em um sistema interativo qualquer que você utiliza diariamente, como o WhatsApp. Todos os seus elementos de interface advém da utilização de widgets bem conhecidos. O formulário onde você escreve suas mensagens é um widget chamado “Text Field”. O botão com ícone de aviãozinho de papel que você pressiona para enviar a mensagem é o widget “Button”. A lista de contatos e grupos de conversa é o widget “Table View”. O efeito sonoro que você ouve quando recebe uma mensagem advém do widget de alarme sonoro. Nessa toada, há certamente milhares de widgets documentados e pré-programados para uso, porém conhecendo algumas dezenas deles você já terá coberta quase a totalidade das necessidades funcionais dos usuários, na produção cotidiana de sistemas interativos.

Há várias bibliotecas de widgets, algumas em nível conceitual, outras em nível de implementação. As primeiras oferecem documentação e manual de instruções de aplicação, mas cabe ao programador implementar os widgets, enquanto as bibliotecas em nível de implementação oferecem widgets em forma de módulos de código para serem prontamente incorporados na construção de qualquer projeto de sistema.

Indicação de Leitura

Conheça algumas das mais eminentes bibliotecas de widgets: (i) Qt, em https://www.qt.io/; (ii) React Native, em https://reactnative.dev/docs/components-and-apis; (iii) Flutter, em https://docs.flutter.dev/development/ui/widgets.

Plataformas e famílias de produtos

Outro problema enfrentado na produção de sistemas interativos é o fato de que os computadores modernos rodam as aplicações em plataformas, com o objetivo de permitir ao usuário operar vários programas ao mesmo tempo e, ainda ainda por cima, de forma integrada, mesmo que advenha de equipes de desenvolvedores distintas, pessoas que nunca se viram e trabalham por iniciativa própria ou empresas concorrentes. Por exemplo, em meu Windows, estou neste momento utilizando o navegador Chrome, da Google, o processador de planilhas Excel, da Microsoft, a plataforma de jogos online Steam, da Valve, e o gerente de credenciais 1Password, de autoria da empresa homônima.

Todos eles foram desenvolvidos por equipes diferentes, mas estão funcionando em meu computador ao mesmo tempo, e, mais que isso, exceto pelo feioso Steam, parecem de fato pertencerem à uma mesma família de produtos, pois utilizam um mesmo estilo de identidade de interação. Dessa forma, o copiar e colar funciona igual em todos eles, inclusive posso copiar uma tabela de uma página no Chrome e colar em uma planilha do Excel. Posso acessar a senha de uma conta de jogo online no 1Password, e pedir ao programa que escreva as credenciais automaticamente em um jogo lançado pelo Steam. Todos esses programas podem ser minimizados ao apertar no botão “X” de suas janelas, e posso imprimir documentos deles acessando o menu “Arquivos” e escolhendo a opção “Imprimir”.

Como é possível que esses programas tão distintos, feitos por equipes diferentes, tenham tantos comportamentos em comum e até mesmo capacidade de integração? Trata-se de uma necessidade do design de interação, afinal os usuários ficam mais produtos e satisfeitos quando suas aplicações apresentam coesão visual e comportamental. Por isso, criou-se o conceito de plataformas, ou famílias, de produtos, que são grandes projetos de design de sistemas interativos que documentam, padronizam, auditam e implementam bibliotecas de widgets, de tal forma que toda aplicação que for feita seguindo os padrões e utilizando os componentes da biblioteca parecerão terem uma origem comum.

Atualmente, as principais plataformas de aplicações são os sistemas operacionais macOS e iOS, da Apple, o Windows, da Microsoft e o Android, da Google. Como menções honrosas, há o Gnome, da Red Hat e o Eclipse da IBM, muito menores, mas razoavelmente conhecidos e utilizados em algumas aplicações historicamente importantes. Há também padrões de família de produtos sem plataforma, como é o caso do Material Design, da Google.

Se há uma coisa que todo usuário detesta é utilizar uma aplicação que se parece e se comporta como um verdadeiro peixe fora d’água. Quem nunca abriu um programa em seu *smartphone* Android com aparência e comportamento de aplicações iOS? A aplicação ignora o botão de menu, e sempre apresenta uma desnecessária seta para voltar à tela anterior. Ou pior ainda, quando você instalou aquela aplicação no

smartphone que parece ser uma tradução direta de um programa de Windows, com formulários prolixos, minúsculos, que se escondem por detrás do teclado virtual. Por mais que ofereça todas as funcionalidades necessárias, nenhum usuário se contenta com esse tipo de aplicação.

Por tanto, é importante definir, desde o princípio de um projeto de sistema interativo, quais serão as plataformas em que ele estará disponível, e implementar interfaces específicas, distintas para cada uma delas. Há algumas ferramentas de mercado que prometem, a partir de uma única implementação genérica, converter a interface gráfica automaticamente para os padrões de cada plataforma. São opções interessantes quando se trata de um projeto pequeno, com equipe e recursos reduzidos. Mas à medida que o projeto ganha complexidade e tamanho, as falhas desse tipo de ferramenta vão se evidenciando.

Videoaula 2

Utilize o QR Code para assistir!

Agora, assista ao vídeo em que abordo os padrões de usabilidade e as plataformas e famílias de produtos.

Indicação de Leitura

Conheça os guias de estilo das principais plataformas modernas: (i) Human Interface Guidelines, da Apple, em https://developer.apple.com/design/human-interface-guidelines/; (ii) Windows App Development, da Microsoft, em https://docs.microsoft.com/en-us/windows/win32/uxguide/how-to-design-desktop-ux; (iii) Material Design, da Google, em https://material.io/design.

Representações da interface

Tanto na prática da antecipação quanto no projeto de interfaces, é necessário produzir representações da interface com o usuário, que são de fato manifestações físicas ou gráficas das telas do sistema interativo. Para tal, há duas classes de representações, a mais simples chamada *lo-fi*, acrônimo em inglês para o termo “baixa fidelidade”, e a

mais completa chamada *hi-fi*, acrônimo para “alta fidelidade”. Nas representações tipo *lo-fi* estão os esboços e wireframes, enquanto nas representações *hi-fi*, temos os protótipos. Esclareço desde já, que os termos “esboço”, “wireframe” e “protótipo” são usados por diversos autores com sentidos trocados, ou então como sinônimos, especialmente entre aqueles mais próximos da engenharia de software do que do design de interação. E não podemos dizer que estão errados, pois essas três palavras realmente podem assumir o sentido umas das outras, exigindo do leitor atenção especial para distinção.

Um esboço é tipicamente um desenho manual de interfaces gráficas, feito em papel com lápis. Esboços têm como principal vantagem o custo baixo, pois qualquer pessoa com pouco treinamento pode desenhar interfaces com um grau de fidelidade. Além disso, por sua aparência simples e acessível, outras pessoas, inclusive usuários, podem contribuir diretamente sobre o artefato, fazendo correções e adicionando elementos. Se um esboço não estiver bom, é fácil descartá-lo por completo e produzir outro. Por fim, um esboço tem a vantagem de comunicar claramente que se trata de um projeto em início de design e construção, o que ajuda no envolvimento e participação de todos os interessados.

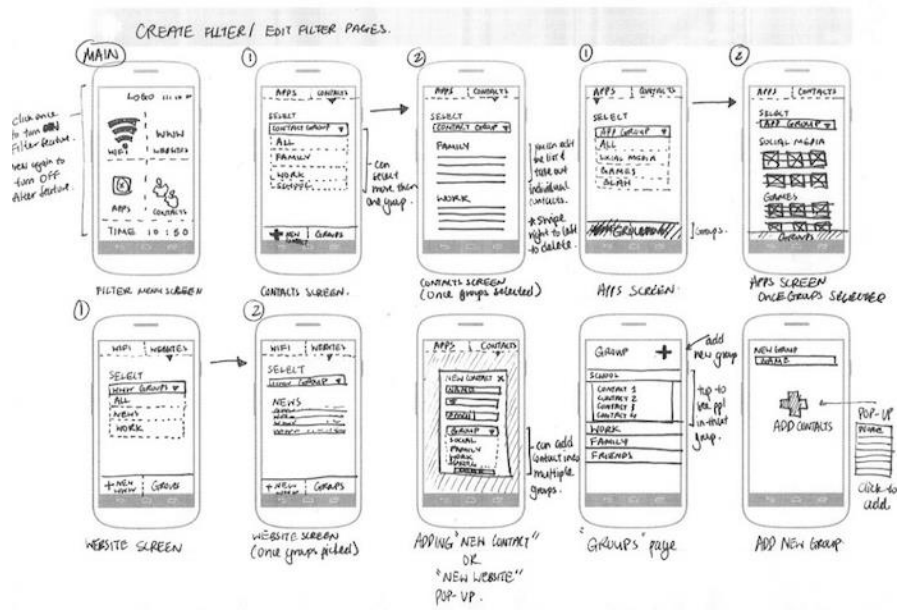
Wireframes são esboços com maior fidelidade, produzidos por programas de computador específicos. Eles têm aparência indistinta de desenho, portanto remetendo a um sistema em estágio inicial de design, mas possuem certas funcionalidades interativas, como o navegar automático entre cenas e a manipulação de certos widgets, como botões e formulários simples. Dessa forma, podem ser utilizados com maior qualidade em testes de antecipação. O problema com wireframes é que são menos acessíveis à contribuição dos envolvidos, já que requerem software específico para o design.

Finalmente, temos os protótipos e representações de alta fidelidade. Sua principal característica é ter aspecto visual ou físico que remetem a sistemas funcionais ou completos. Um protótipo é produzido ou por ferramentas específicas de prototipação ou por ferramentas completas de desenvolvimento de sistemas. Um protótipo pode oferecer algumas das funcionalidades reais do sistema, o que ajuda nos testes de usabilidade e antecipação, para averiguar se o sistema está no caminho certo de cumprir seus objetivos de apoio ao usuário. Protótipos podem ser horizontais, quando oferecem várias funcionalidades simples e superficiais, ou verticais, quando oferecem um caminho específico de funcionalidades complexas.

Um problema comum na produção e utilização de protótipos é que, a depender do momento do projeto, pode enganar os envolvidos quanto ao estágio de desenvolvimento do projeto. Uma pessoa que manipula um protótipo, mas não participa do desenvolvimento, normalmente imagina que o projeto já está em etapa de finalização. Por vezes, isso acontece até mesmo com os gerentes dos projetos, e, por vezes, o protótipo passa a ser produto. Do ponto de vista da engenharia de software, isso pode ser um desastre, já que antecipa-se o lançamento do sistema, mas torna-se a etapa da manutenção do sistema muito mais custosa ou até inviável.

Na elaboração de representações de interfaces, o designer pode produzir o que chamamos de *storyboards*, ou seja, uma coleção de cenas representadas como interfaces e ligadas entre si, para denotar as transições, e apontar os mecanismos que as causam. Se você já tiver elaborado seu MoLIC, basta transformar cada cena do diagrama de estados em uma representação de interface. O storyboard é então passado à equipe de engenharia de software, que se encarregará de transformá-lo em sistema interativo, com a maior fidelidade possível. A figura 5 ilustra um *storyboard* típico.

Figura 9: Exemplo de um *storyboard lo-fi*



Fonte: https://medium.com/@jjman505/how-to-storyboard-an-app-ede5ce249ea5. Acesso em: 21 mar. 2022.

Importante

Há várias ferramentas especializadas na produção de representações *lo-fi* e *hi-fi* de interfaces. Algumas delas são o Balsamic Moqups, o SketchUp, o Adobe Photoshop e InDesign e o XCode. Mas, atualmente, a principal ferramenta utilizada para este propósito é o Figma, e recomendo que você aprenda a utilizá-lo. Este vídeo-curso do freecodecamp é a melhor introdução que conheço:

- UI / UX Design Tutorial – From Zero to Hero with Wireframe + Prototype + Design in Figma, em https://www.freecodecamp.org/news/ui-ux-design-tutorial-from-zero-to-hero-with-wireframe-prototype-figma/

Avaliação de usabilidade

Apesar da natureza holística do design de sistemas interativos, que o envolve em incertezas e faz com que todo projeto seja uma pesquisa, precisamos estar seguros de

que de fato criaremos soluções melhores do que o que já existe. E só há uma maneira de fazê-lo: testar e avaliar constantemente nossos designs, desde o princípio do processo.

Há várias técnicas para fazê-lo, e inclusive algumas das técnicas de levantamento das necessidades dos usuários podem ser também utilizadas para avaliação, especialmente as entrevistas e os estudos de campo. Destaco três técnicas específicas e eficientes na avaliação de designs, que são o teste de usabilidade, a avaliação heurística e o percurso cognitivo.

A primeira delas, o teste de usabilidade, é a mais importante. Trata-se de definir critérios de qualidade a serem mensurados, elaborar perguntas objetivas, hipóteses, a respeito desses critérios, definir quais tipos de dados ou conjunto de informações são necessários para respondê-las, e por fim, elaborar um plano de uso, uma atividade a ser executada por um usuário por meio do sistema. Após estar preparado, coloca-se o usuário para utilizar o sistema, em um ambiente monitorado, gravando suas expressões faciais, o conteúdo da tela e toda entrada de dados no computador que ele estiver utilizando. O usuário então lê o roteiro de atividades e vai tentando cumpri-las da maneira mais natural que puder.

Ao final da sessão, tem-se em mãos dados ricos para serem analisados e medidos. Pode-se, por exemplo, medir quanto tempo levou para o usuário colocar um objeto à venda no site de leilões. Ou então, quantos erros ele cometeu para encontrar a função de entrar equações matemáticas em um processador de textos. Pode-se medir quantas das atividades foram completadas de maneira satisfatória. Pode-se ainda fazer testes A/B, comparando sistemas ou designs concorrentes, para descobrir quais deles se saem melhor em quais aspectos da usabilidade.

O teste de usabilidade é definitivamente o mais rico e produtivo entre todos, mas é também o mais caro de se realizar. Não apenas requer uma preparação cuidadosa, softwares especializados em gravação com múltiplas fontes, e tipicamente um ambiente físico específico, mas também o tempo e disponibilidade de alguns usuários. A verdade é que a maioria dos projetos de software não têm acesso a todos esses recursos.

Logo sobram a avaliação heurística e o percurso cognitivo. Ambas são técnicas baseadas em inspeção, o que significa que são realizadas pelos próprios designers, sem a presença do usuário. Isso imediatamente as torna mais baratas, mas ao mesmo tempo muito menos ricas. Ainda assim, é melhor fazer algum tipo de avaliação do que simplesmente deixar passar.

Na avaliação heurística, método inventado por Nielsen (1994), o designer define uma lista de controle de diretrizes de usabilidade que levam a qualidades desejáveis, de preferência ainda no início do projeto. O próprio Nielsen e vários outros autores propõem listas heurísticas a serem usadas com esse propósito, assim como os guias de plataformas às disponibilizam. Com uma lista dessas em mãos, o designer ou algum outro colaborador inspeciona toda a interface para conferir que a diretriz está sendo

seguida, anotando em quais cenas ela está falha e o grau de severidade de cada falha. Ao final, tem-se um relatório classificado de falhas a serem corrigidas em uma nova iteração do design de interação do sistema.

Por fim, no percurso cognitivo, o designer conhecedor do perfil do usuário simula a utilização do sistema como se ele próprio fosse o usuário. Isso requer, claro, uma boa dose de imaginação, empatia e estudo sobre o usuário. A elaboração de personas certamente ajuda. Para manter a fidelidade da simulação, o designer propõe-se a cumprir um objetivo típico do usuário e, ao utilizar o sistema, narra os pensamentos que ele imagina que o usuário terá em voz alta, sempre se indagando “como o usuário faria para descobrir que precisa acessar a funcionalidade X para atingir o seu objetivo”? “Como ele perceberia que está evoluindo”?

Videoaula 3

Utilize o QR Code para assistir!

Agora, assista ao vídeo que aborda...

Atividades Avaliativas e de Fixação

Prezado(a) aluno(a)!

Chegou a hora de colocarmos em prática os conhecimentos adquiridos até o momento!

As avaliações nos possibilitam perceber se estamos obtendo bons resultados e quais são os conteúdos que requerem nossa atenção.

Clique em “**Módulos**” no “**Menu Lateral**” e acesse a **Atividade Fixação e Atividade Avaliativa**.

Atente-se aos prazos! Após a data limite, a avaliação será encerrada. Bons estudos!

Videoaula Encerramento

Utilize o QR Code para assistir!

Assista agora ao vídeo de encerramento de nossa disciplina.

Encerramento

Nesta unidade você finalmente terminou o ciclo completo do design de sistemas interativos e, principalmente, chegou em sua atividade final, que é o design de interação e de interfaces. Você aprendeu técnicas de sistematização para elaborar o design conceitual, a partir da síntese de análise, feitas na unidade anterior. Você agora sabe que existem variados tipos de interfaces, bibliotecas de componentes de interface chamados widgets, que podem ser padronizados estilística e comportamentalmente, para participarem efetivamente em uma plataforma de aplicações. Você aprendeu sobre os distintos tipos de esboços e protótipos, e o momento correto de utilizar cada um deles. Finalmente, você fechou o ciclo ao conhecer três técnicas de avaliação de usabilidade, que serão seus guias de correção de rumo em sua futura atividade profissional como designer de usabilidade e de sistemas interativos.

Encerramento da Unidade

Prezado aluno, agora que você chegou ao fim deste curso remeto à minha mensagem de abertura da disciplina, quando indaguei sobre os motivos que fazem com que as empresas mais valiosas do mundo, no momento, sejam empresas de tecnologia. Na oportunidade, destaquei o fato de que havia muitas empresas de tecnologia no mundo, a maioria delas certamente sem qualquer grau de sucesso.

Portanto, fiz meu ponto de que não basta ter a tecnologia no cerne da missão do empreendimento para ter sucesso, mas deveria haver algum fator que, em comunhão com a tecnologia, faz o sucesso exponencial, a ponto de que as empresas mais valiosas sejam todas empresas novas, considerando-se a idade do livre mercado e das atividades empresariais. Acredito que agora você entende o motivo de minha aposta ser na abordagem diferenciada quanto às qualidades de uso e acessibilidade dos produtos e serviços dessas empresas.

Agora você entende o quão complexo é um projeto de sistema interativo, cuja interface é o último elemento a ser definido e testado, somente após entendermos as

pessoas, as atividades e o contexto de uso. Neste momento, você é também capaz de entender porque nenhuma interface é boa o suficiente, e porque as empresas, mesmo tendo bilhões de dólares à disposição, não conseguem acertar definitivamente no projeto. Você agora adquiriu a mentalidade crítica e construtiva a respeito do design de interação.

Referências

BARBOSA, Simone Diniz Junqueira; SILVA, Bruno Santana da. **Interação humano-computador.** Rio de Janeiro: Elsevier, 2010.

BENYON, David. **Interação humano-computador**. 2. ed. São Paulo: Pearson, 2011. ISBN 9788579361081.

NORMAN, Don. **The Design of Everyday Things:** Revised and Expanded Edition. Basic Books (AZ), 2013.


UniFil
EDUCAÇÃO A DISTÂNCIA
UNIFIL.BR