**PROTOTIPAÇÃO DE SOFTWARES**

**Veja as vantagens da prototipação, os desafios do processo e que boas práticas e alternativas ajudam a evitá-los**.

A prototipação de software é um conceito bem conhecido dentro do desenvolvimento de produtos, com a propularidade de metodologias como **design sprint.** Faz todo sentido, ao ter uma ideia de negócio, iniciar por um protótipo. Sobretudo depois de tantas histórias de projetos bem sucedidos apenas para os desenvolvedores, mas não para os **stakeholders** e usuários.

O protótipo é um modelo funcional e interativo de um produto. Em outros termos, é uma visualização de como o produto vai funcionar, de alguns fluxos e do layout inicial.

A prototipação é uma maneira de testar se uma ideia pode e deve se tornar em um **MVP**, antes de decidir ir em frente com o projeto ou não. Protótipos ajudam a trazer respostas a questões como as seguintes:

* Como essa ideia se torna um produto?
* Temos condições de fazer dessa ideia um produto?
* Temos mercado e clientes potenciais para um possível produto?

Porém, dizer que o conceito de prototipação é conhecido não é o mesmo que dizer que é praticado. Poucos prototipam de maneira sistemática ou até no verdadeiro sentido do termo. Muitos saem direto da ideia para a definição de escopo e, daí, para o código, sem pensar muito em disgn inicial e testes de usuários.

Os motivos para justificar o salto são vários. Alguns mencionam a suposta falta de tempo para fazer o protótipoe os custos do processo. Há quem tenha a impressão de que o produto não precisa ser prototipado quando os criadores já sabem quais são os requisitos.Alguns pensam que o protótipo é desnecessário se ja têm um **wireframe**. Outros arriscam afirmar que os usuários se comportam de maneira diferente diante de protótipos e de produtos reais, o que invalida os resultados. Por fim, tem quem considere desnecessário prototipar dentro do contexto de metodologias ágeis.

**A prototipação de softwares: os prós**

O protótipo é uma versão-teste e, portanto, incompleta de um produto. Apesar da incompletude de um protótipo já desqualificar o processo como um todo para alguns, ele é útil. Sobretudo em relação a softwares, em que é comum que haja uma compreensão apenas parcial ou suposições sobre tudo o que uma solução precisa.

A prototipação tangibiliza ideias com baixo custo, tempo e energia, ajudando na análise dos requisitos, da aceitação de usuários, da complexidade do projeto, da capacidade técnica da equipe, da organização, dos custos e do tempo de desenvolvimento.

A prototipação de softwares, portanto adquire valor quando a organização precisa determinar a factibilidade de uma ideia de negócio em várias frentes, ou seja se ela se transforma em produto e se esse produto é capaz de ser desenvolvido pela organização.

A prototipação não se esgota apenas nos aspectos relativos a capacidade da organização de executar o projeto. Ela também pode responder se a organização deve desenvolver essa ideia. Nesse caso, o protótipo também serve de prova de que a organização deve tocar um projeto adiante e como, apesar de possíveis limitações.

No longo prazo, a prototipação está por trás de economia de tempo e dinheiro no desenvolvimento de softwares. Mas sobretudo, está por trás de um produto melhor. Ela evita que suposições ruins avancem no desenvolvimento do produto, diminuindo o custo da correção, mas também traz ideias mais refinadas, não concebidas inicialmente. Apesar de todas as vantagens, o processo não é isento de desafios.

**Os desafios da prototipação de software**

Apesar de beneficiar projetos, a facilidade e os casos de sucesso da abordagem podem levar a aplicações descuradas, que afetam a efetividade ou, no limite, levam a resultados inócuos ou errados.

Por exemplo, um desafio comum é determinar a medida certa de tempo e esforço que a equipe deve dedicar a criação de protótipos. Normalmente, elas investem muito mais tempo do que precisam.

Isso pode produzir o chamado efeito posse, isto é, um apego exagerado pelo protótipo. Esse excesso torna a ideia ou o protótipo preciosos demais para falharem ou para serem abandonados. Como consequência, pode gerar vieses e má vontade em relação a resultados de testes.

Outro problema é o excesso de confiança gerado pelos primeiros resultados bem-sucedidos. Isso pode levar a equipe a não testar suficientemente. É bem comum uma postura de confirmação das suposições iniciais, para dar o processo por encerrado rapidamente. No final, a equipe fica com uma definição apenas parcial do produto.

**Outro desafio** é usar protótipos muito diferentes da solução final. Protótipos definitivamente não são o produto final. Embora haja protótipos com diferentes níveis de fidelidade ao resultado, essa escolha pode levar a resultados enviesados, tanto para o lado positivo quanto negativo.

Os desafios são muitos. Porém não são suficientes para atestar o insucesso da abordagem, mas a necessidade de fazer uma apropriação consciente.

**Boas práticas de prototipação de softwares**

1. Tenha um propósito para a prototipação

Responda com clareza **por que você está criando um protótipo**. Saiba o que você precisa validar e, a partir disso, determine o que o protótipo precisa ter para responder a essa pergunta.

1. Expore diferentes modelos de protótipo

Um software pode ser desenvolvido de inúmeras formas. Descartar possibilidades já na prototipação para apostar todas as fichas em apenas uma pode ser contraproducente. Os efeitos disso podem ser apego demais, sem uma resposta definitiva contra ou a favor de sua ideia.

Não dá para prototipar demais, porém **partir das várias ideias da equipe até reduzir a algumas opções** para transformar em protótipo, para serem testados concomitantemente ou em sequência, evita esses problemas.

1. Comece com protótipos baratos e rápidos

**Manter-se simples e focado no essencial para ser rápido e barato, mas altamente efetivo:** essa é a chave da prototipação em estágio inicial, quando você não precisa fornecer alta fidelidade.

1. Mostre em vez de dizer

Não é raro que os criadores passem muito tempo explicando sua ideia, mas pouco tempo mostrando e dando tangibilidade, pondo-as à prova na prática. Esse foco na teoria pode ser evitado se você der **primazia à demonstração**. Explique sua ideia enquanto a mostra em funcionamento. Além de facilitar a compreensão, você dará abertura a novas ideias.

1. Preparece para destruir o protótipo

Você deve ter desapego suficiente em relação à sua linha de raciocínio para estar preparado para abandonar ou destruir protótipos, quando necessário. Essa mentalidade parece simples, mas não é. **Exige o grau de comprometimento, de esforço e de tempo certo** para a equipe não mergulhar no projeto antes mesmo de entender se ele pode dar certo.

1. Prepare-se para falhar

Protótipos são testes e, como tal, é comum que falhem, seja por suposições erradas ou por não responderem de fato à questão levantada. **Falhas consecutivas, ideias que não funcionam ou percepção negativa podem levar a equipe à frustração** e, finalmente, ao desânimo.

Na prototipação de softwares, é preciso adotar uma **mentalidade de aprendizado** e um pouco de insistência. As falhas devem ser vistas positivamente, como oportunidades de aprendizado. Elas podem conduzir a refinamentos ou a novas ideias que, essas sim, tenham sucesso. Além do mais, abandonar ideias que pareciam boas na teoria é bem melhor do que ter um projeto inteiro mal-sucedido.

1. Tenha uma visão de longo prazo

Falta de tempo para prototipar é uma das razões mais usadas para justificar o salto dessa etapa de desenvolvimento de softwares. Não para quem tem visão de longo prazo. **Se prototipar é aprender sobre como o produto deve ser para ser bem aceito pelos usuários, fazê-lo é que leva a economizar tempo.**

Não são raros escopos de projetos longos que vão mudando ao longo da execução, levando a atrasos e gastos extras.

**Existem alternativas à prototipação?**

A prototipação não é a única forma de validar a factibilidade e o potencial de uma ideia sem investir grandes esforços ou orçamento.

Há alternativas, com **diferentes enfoques de análise**. A escolha de um método ou da combinação deles vai depender de aspectos como maturidade do projeto e alcance do produto. Veja duas outras formas de validar uma ideia antes de ir para o projeto:

**Prova de conceito**

A PoC é um projeto pequeno, normalmente utilizado apenas internamente, para **validar se uma ideia é viável em termos de desenvolvimento**. Basicamente, a PoC responde se ela é tecnicamente viável, se funciona e pode ser feita, enquanto o protótipo responde de que maneira ela pode ser feita e se terá público.

Na prova de conceito, consequentemente, ainda **não é considerada a usabilidade** de uma possível solução**ou se há mercado**para ela, por exemplo. Por isso, tampouco faz parte do escopo de uma prova de conceito levá-la a público.

A PoC é usada, sobretudo, em pesquisas para **explorar tecnologias** dentro de centros de inovação, mas também pode ser uma forma de **provar um conceito para stakeholders**. Nesse caso, ela ajuda tanto no processo de assessment quanto em estimativas.

Então, se a organização ainda não sabe se sua ideia pode ser feita, é bom começar pela PoC antes de prototipar.

**MVP (produto mínimo viável)**

O MVP é um passo seguinte em relação a um protótipo: é a **mínima forma de um produto completo e funcional, capaz de ser testado e validado no mercado.**

Assim, enquanto um protótipo é relacionado ao primeiro estágio de desenvolvimento e apresentação aos usuários, mas sem ter grande valor de mercado, o MVP já é desenhado para o mercado. Ele é uma proposta de valor, embora ainda utilize a **inteligência dos adotantes iniciais para tomar melhores decisões sobre incrementos de produto**. Em cada MVP, portanto, o produto avança em maturidade.

**Vocabulário**

**design sprint:** O Design Sprint foi uma metodologia desenvolvida dentro do Google pelo designer Jake Knapp para prototipagem de soluções digitais. É um processo exclusivo de cinco dias em que ideias são mapeadas, desenhadas, estruturadas e testadas.

**stakeholders:** Podemos definir o que são stakeholders como, conforme já dizia Robert Edward Freeman, grupos e indivíduos que, de uma forma ou de outra, apresentam algum nível de interesse nos projetos, atividades e resultados de uma determinada organização.

**wireframe:** Podemos definir como um esqueleto, um protótipo ou uma versão bastante primitiva do visual de um projeto. Ele consiste na representação da diagramação e das estruturas macro do site, ou seja, apresentamos por meio de formas geométricas e linhas como pensamos a divisão da interface em seções.