

Engenharia de Software

Prof. Vinicius Cardoso Garcia

vcg@cin.ufpe.br :: [@vinicius3w](https://twitter.com/vinicius3w) :: viniciusgarcia.me

[IF977] Engenharia de Software

<http://bit.ly/vcg-es>

Licença do material

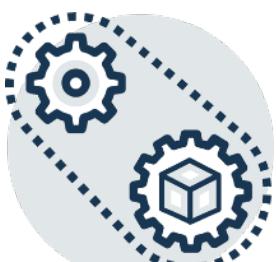
Este Trabalho foi licenciado com uma Licença

Creative Commons - Atribuição-NãoComercial-
Compartilhual 3.0 Não Adaptada



Mais informações visite

[http://creativecommons.org/licenses/by-nc-sa/
3.0/deed.pt](http://creativecommons.org/licenses/by-nc-sa/3.0/deed.pt)



Referências

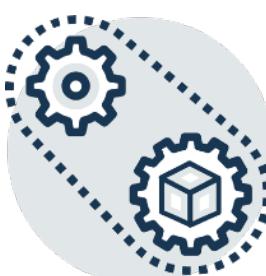
- Matt Watkinson. **The Ten Principles Behind Great Customer Experiences** (Financial Times Series). FT Press; 1 edition (December 16, 2012).
- Jesse James Garrett. **The Elements of User Experience: User-Centered Design for the Web and Beyond** (2nd Edition) (Voices That Matter), New Riders; 2 edition (December 26, 2010).
- [O que é Design de Experiência e por que você deveria prestar atenção nele](#)
- [Afinal, o que é Design de Experiência e por que meu negócio precisa dele?](#)
- [Design de Experiência: Como trabalhar as necessidades do usuário a favor dos negócios](#)
- [Social Experience Design: one method, two tools, three tips, the lecture, by Erin casali](#)
- [A Startup Expansiva, by Frederick van Amstel](#)

**Fica,
vai ter
bolo!**

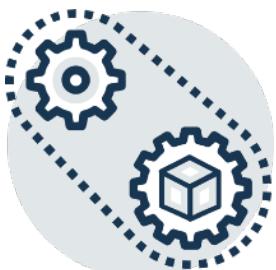


Fica, vai ter bolo!

O que faltou a gente conversar a respeito?



**E agora, já sabem o que é
a Engenharia de Software?**





Você está errado sobre o que é Engenharia de Software

Fonte: <http://bit.ly/2WxPlIP>



É o quê então?

- Se você acha que é apenas programação, design de algoritmos e estruturas de dados
 - então você está **errado**
- Se você acha que não tem nada a ver com programação, design de algoritmos e estruturas de dados
 - então você está **errado**



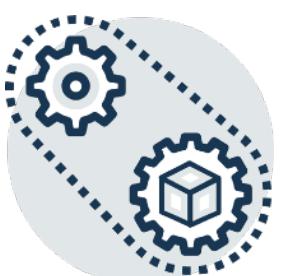
Vó, vem ver, eu fiz um app!

- Atualmente, qualquer pessoa que constrói um aplicativo móvel bonito ou cria um "site" afirma ser um engenheiro de software
- Só porque você conhece uma linguagem de programação não significa que é um engenheiro de software
- Além disso, você não precisa ser um engenheiro de software para aprender a criar um aplicativo ou site
- **Engenharia de software é mais do que programação!**



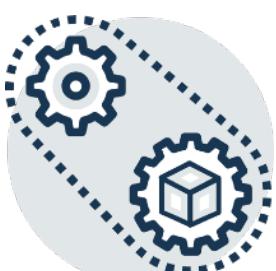
Caneta azul, azul caneta

- Como engenheiro de software, espera-se que você tenha **boas bases em computação e programação** (ao menos)
- É por isso que você cursa disciplinas relacionadas à programação
 - No entanto, os cursos de programação representam uma parte menor do currículo



Se tá rodando, tá ok

- Como engenheiro de software, você não tem a mentalidade de “apenas vamos fazer funcionar”
- Em vez disso, sua mentalidade é focada no **entendimento dos requisitos do usuário, na validação e verificação dos requisitos, no gerenciamento de projetos, nas restrições de tempo, na qualidade do produto, na análise dos stakeholders** e muito mais.



Nem tudo é programação

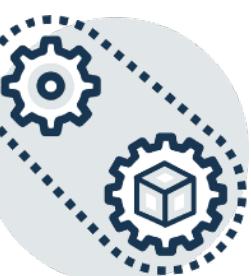
- Engenharia de Requisitos
- Processos de Negócio
- Arquitetura de Software
- Qualidade de Software
- Gerência de Projetos

•
...



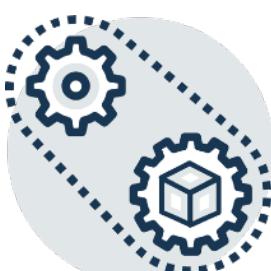
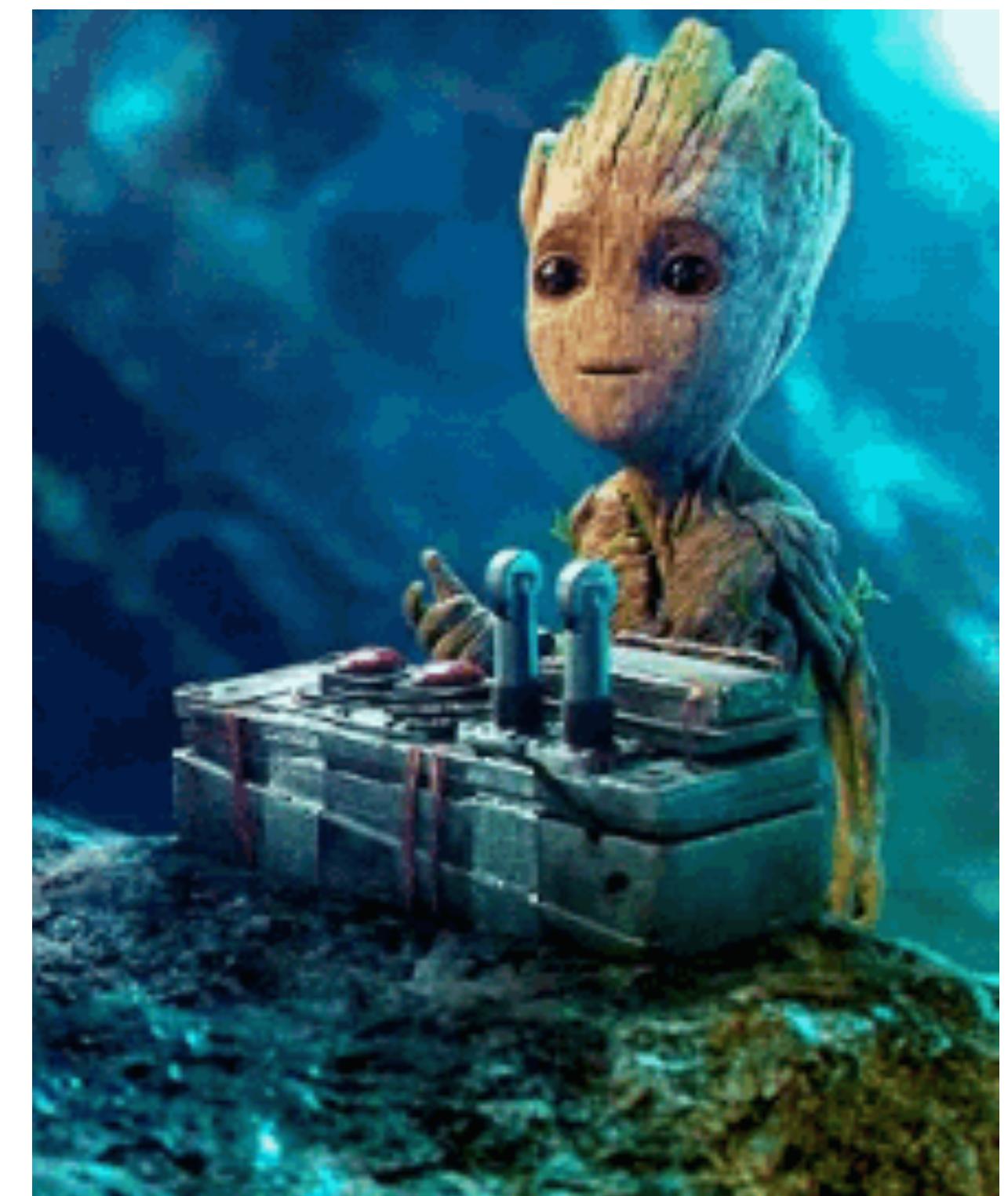
Para Mordor!

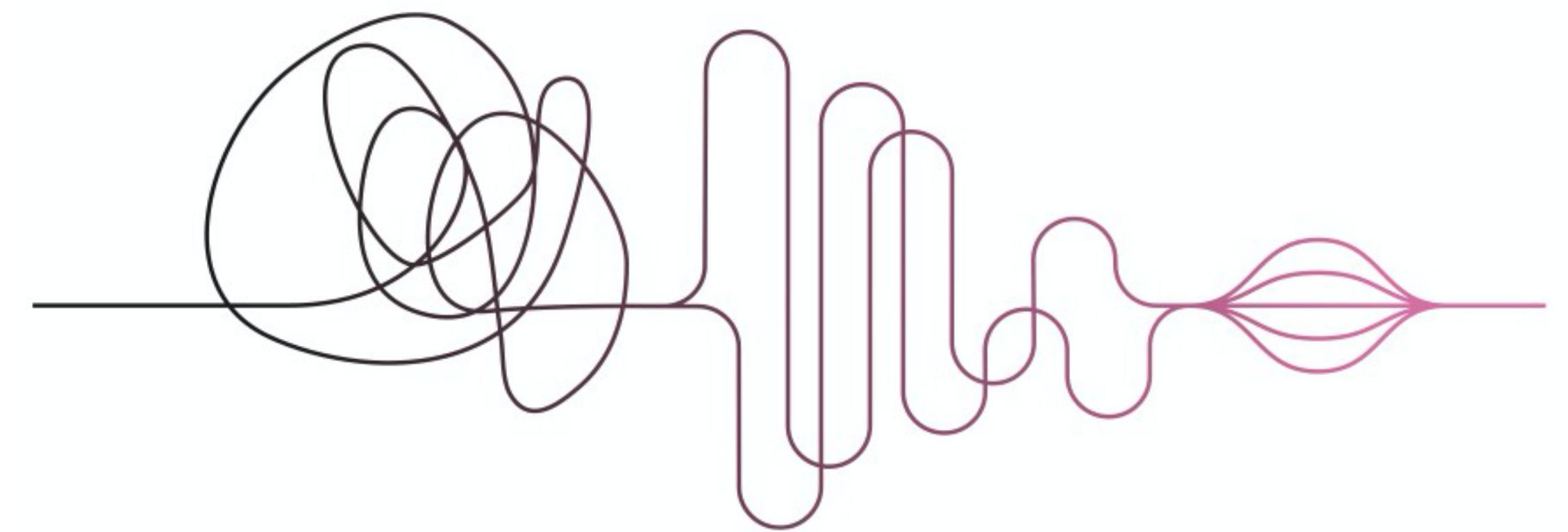
- Para ser um engenheiro de software **bem-sucedido**, é essencial ter um bom entendimento do **processo de desenvolvimento de software** e garantir que o software em desenvolvimento seja construído dentro das **restrições de tempo e orçamento** e da **expectativa de qualidade esperada pelos stakeholders**
- Além disso, é essencial ser capaz de **obter** requisitos do usuário, gerenciar as partes interessadas do projeto e satisfazer suas necessidades
- Por fim, como engenheiro de software, você está interessado em criar software de **alta qualidade** implementando **arquiteturas apropriadas** que facilitam o processo de **manutenção**



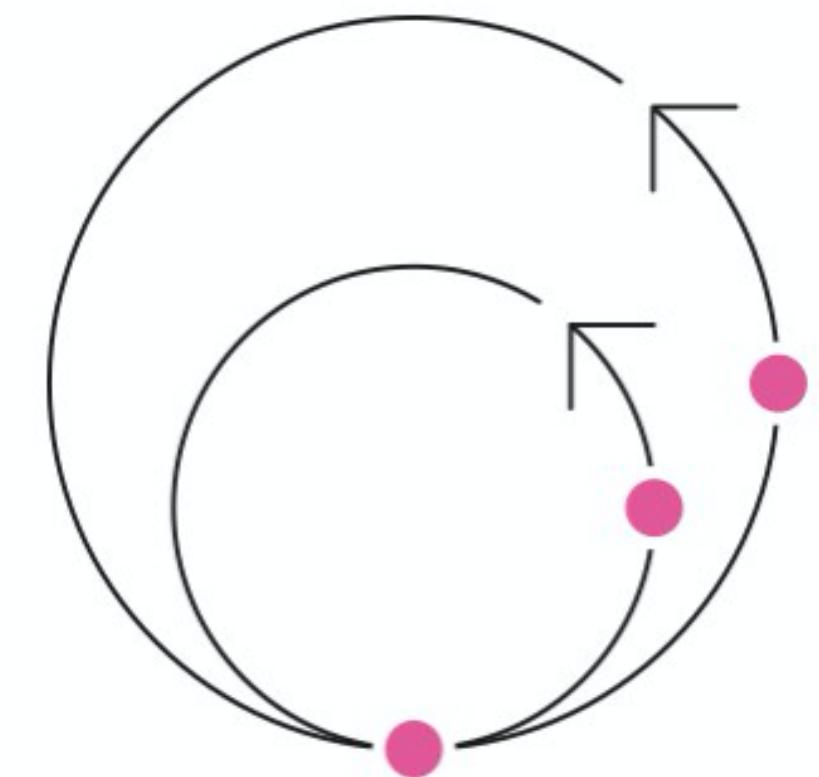
I'm groot!

- Programação é uma pequena parte do currículo de ES
- CC, por outro lado, é o campo da computação mais preocupado em implementar e analisar algoritmos
- Mas isso não significa que você não pode trabalhar como programador
 - Na verdade, muitas pessoas começam com essa tarefa
 - E tantas outras se mantém exclusivamente nela!

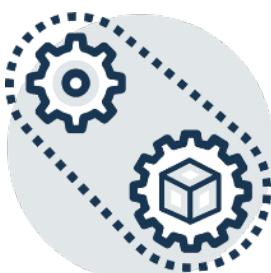




vs.



Design Sprint vs Agile Sprint

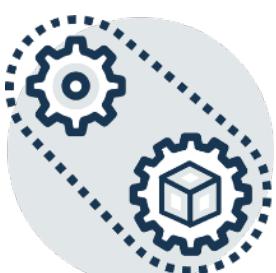


Fonte: <https://medium.com/pminsider/design-sprints-vs-agile-dev-sprints-eb5a11a997a8>

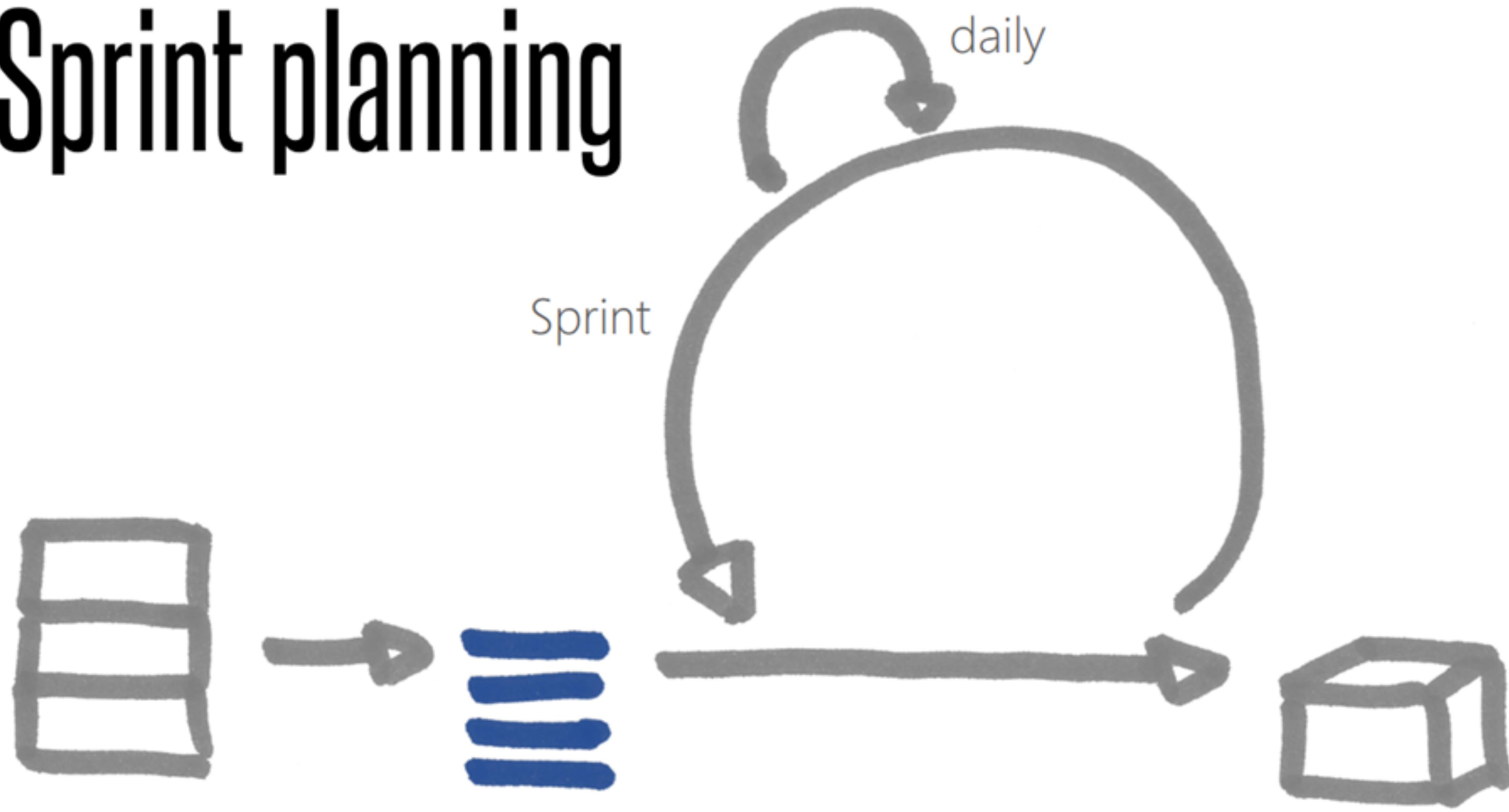


Introdução

- Nos anos 2000, o mundo dos negócios se acostumou a termos de desenvolvimento de produtos como **agile**, **scrum**, **lean** e **MVP**
- No lugar da cascata, surgiram abordagens de desenvolvimento iterativas, incluindo Scrum, Kanban, XP (programação extrema) e outras variações
- Essas disciplinas se afastaram da abordagem em cascata substituindo-a por ciclos enxutos e mensuráveis de construção, teste e entrega
 - Esses ciclos rápidos, normalmente de 1 a 2 semanas, foram apropriadamente chamados de "sprints"



Sprint planning



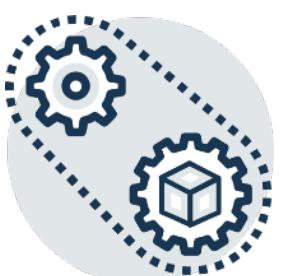
product backlog

Sprint backlog

product increment

Mas com a evolução

- Com os líderes C-level e o pessoal de Vendas e Marketing se acostumou a sprints e aprendeu a falar o idioma mais recente de seus grupos de tecnologia, em 2015 um novo 'sprint' começou a surgir - o Design Sprint
- Design Sprints foi criado com base no framework de design thinking da IDEO, foi originalmente incubado no Google por [Jake Knapp](#), onde ajudou a orquestrar o lançamento bem-sucedido de produtos como Gmail e Hangouts
 - Equipes multidisciplinares podem passar **5 dias** entendendo, identificando, prototipando e testando problemas de grandes empresas, antes de iniciar o desenvolvimento completo do produto



Mas calma...

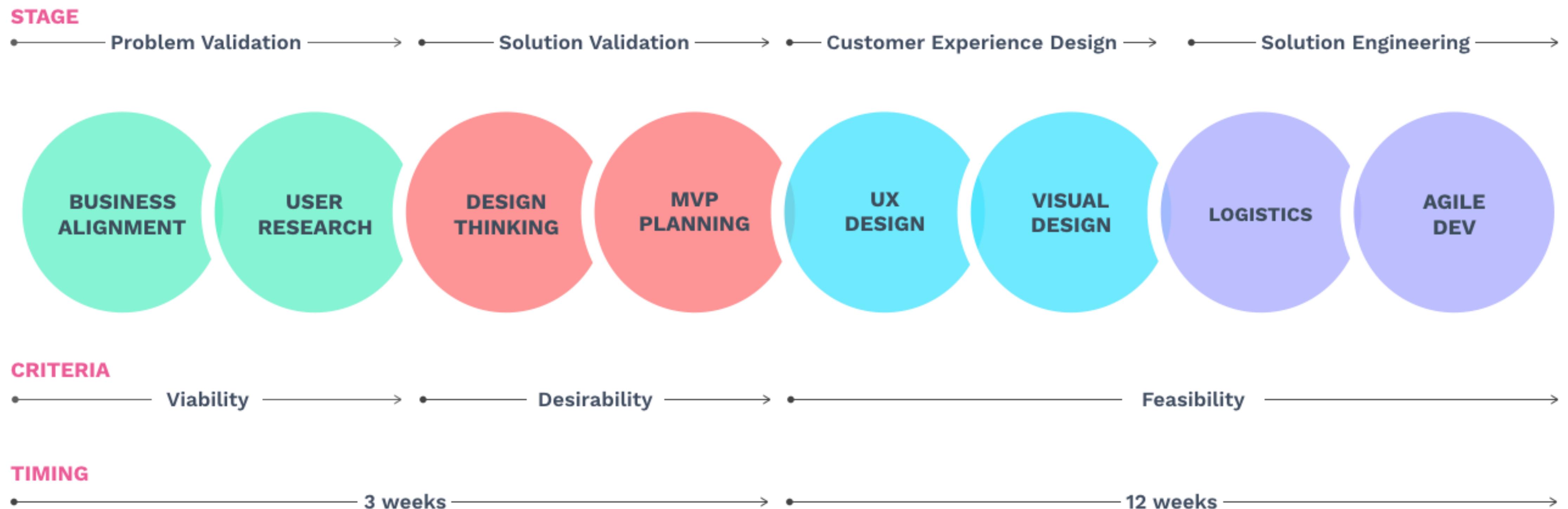
- Vamos recapitular tudo começando por Inovação habilitada por software



18



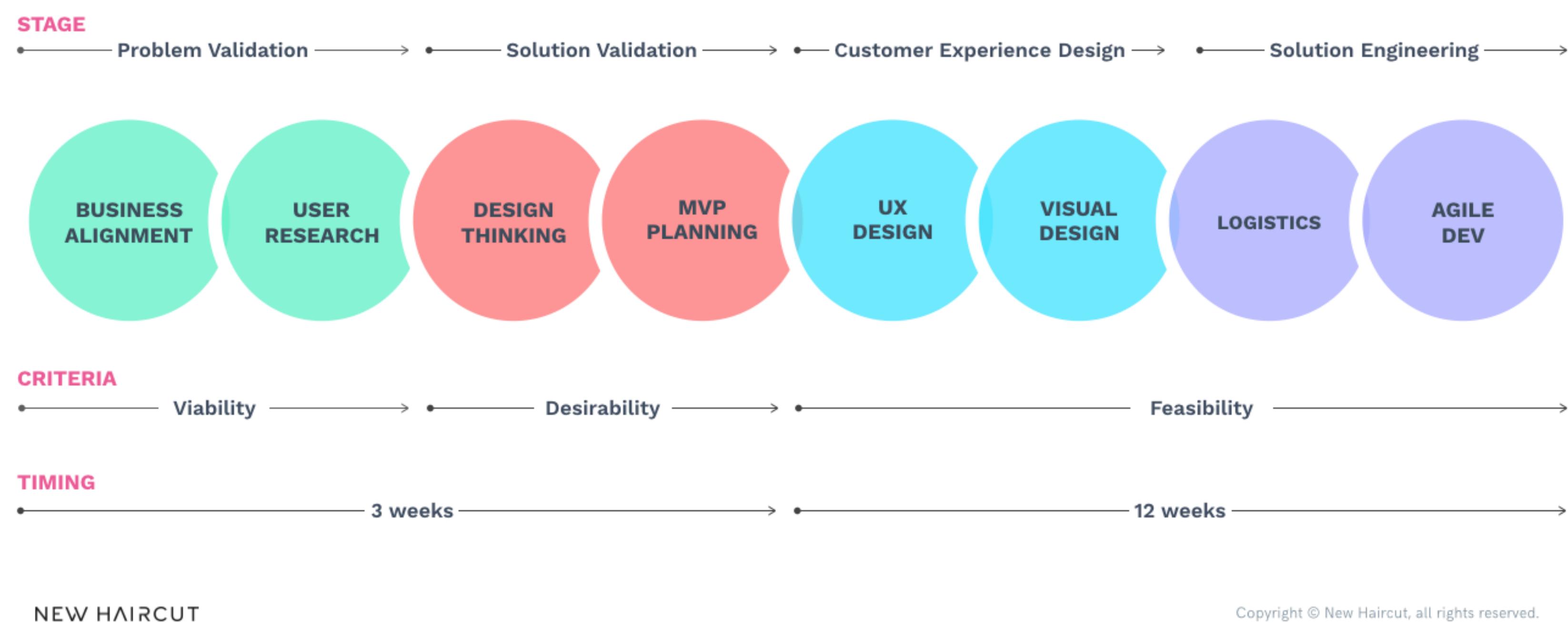
Processo de criação de soluções digitais para resolver grandes problemas



NEW HAIRCUT

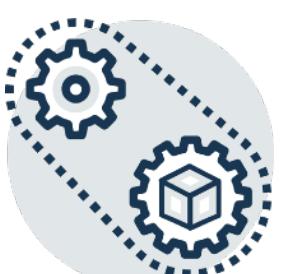
Copyright © New Haircut, all rights reserved.

By <http://newhaircut.com/>



Copyright © New Haircut, all rights reserved.

- **Business Alignment:** Deciding to work on problems that align with the strategy, vision, and resources of the company
- **User Research:** Getting to know more about the market, users, and competition that surround those problems
- **Design Thinking** (via Design Sprints): Building an understanding of the problem and validating prototyped solutions with potential customers
- **MVP Planning:** Creating a complete picture of your solution and deciding on the minimum valuable product (MVP) you'll bring to market inside of a Product Roadmap
- **UX Design:** Building out the information architecture, interactions, and flow of the human-centered experiences you'll offer within your solution
- **Visual Design:** Giving your solution a style, tone, and interface (when applicable)
- **Logistics:** Plotting out the system infrastructure and application architecture of your solution
- **Agile Development:** Building the crux of your software solution and launching to market.



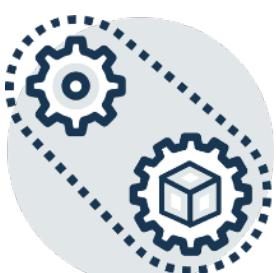
Sprint para novos produtos

- Na primeira vez, com essa abordagem - quando você está construindo um novo produto - os sprints de design acontecem **antes** dos sprints de desenvolvimento ágil
 - Isso faz sentido!
- Antes de começar a projetar e codificar, você deseja criar uma **perspectiva do problema e do usuário** e testar soluções com protótipos descartáveis
- Os protótipos não são apenas **mais baratos e mais rápidos**, mas a equipe se torna menos emocionalmente investida em um protótipo do que telas e códigos de trabalho muito bem projetados



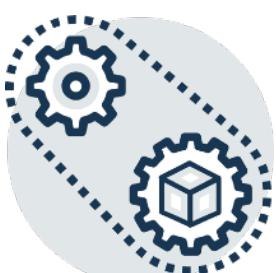
Como?

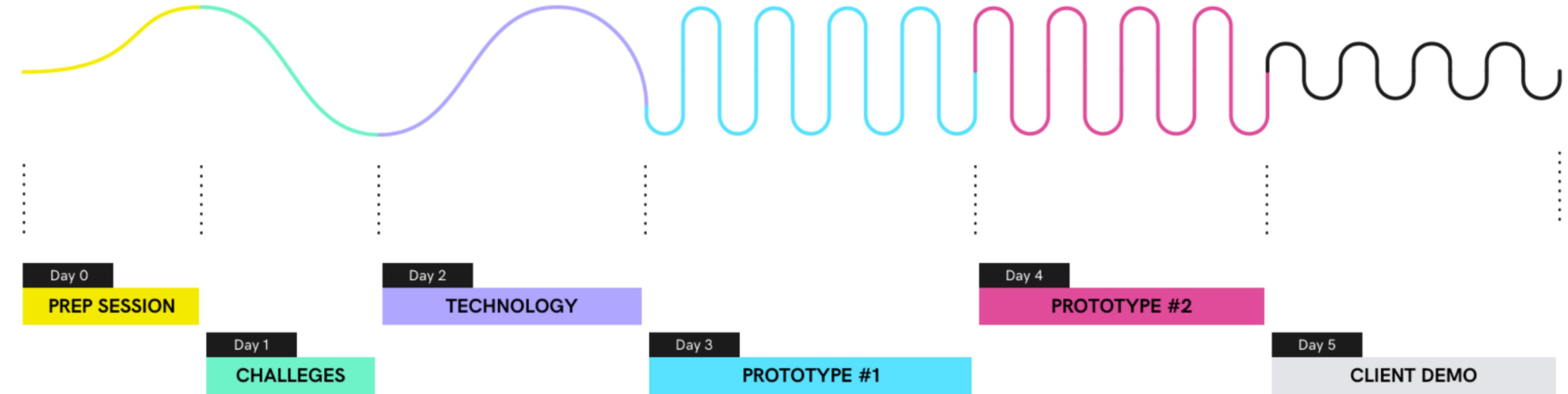
- Quando você sai de uma sprint de design, tem um protótipo que foi testado com ~5 usuários-alvo
- Como, então, fazemos a ponte do **protótipo para o produto funcional?**
- Um Roteiro do Produto expandirá o foco específico do protótipo de sprint de design para detalhar os recursos/telas/fluxos de usuários priorizados restantes
- Essas prioridades se tornam **entradas no backlog do seu produto**, que pode ser alimentado nos seus sprints de desenvolvimento ágeis



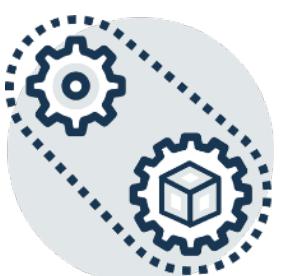
Sprints de Codificação

- Sempre devemos garantir a presença de um especialista em engenharia em nossos sprints de design
- Ainda assim, percebemos quando aquele engenheiro único levou as informações de sprint de design de volta à sua equipe de arquitetos, codificadores e testadores - ainda havia uma tonelada de perguntas que eles tinham antes que pudessem se aprofundar na resolução
 - Qual é a melhor tecnologia?
 - Existem soluções existentes que podemos incorporar?
 - Onde devemos nos concentrar primeiro?





- Em uma Sprint de Codificação, as equipes de tecnologia passam de 4 a 5 dias para:
 - Entender todas as informações reunidas anteriormente
 - Identificar os 2-3 maiores desafios tecnológicos que eles precisam superar
 - Passe para pesquisa e construa protótipos concorrentes para cada desafio
 - Teste os protótipos que ajudam a decidir sobre as implementações vencedoras



Sprint de Codificação

- Uma Sprints de Codificação deve usar as mesmas técnicas de foco intenso, duração e convergência divergente de um Design Sprint
- Posteriormente, as equipes de tecnologia são capazes de responder não apenas às perguntas "O que estamos construindo?" e "Como estamos construindo isso?"
- Mas produzir uma camada fundamental de **infraestrutura, arquitetura e código** a serem construídos dentro da sua sprint ágil de desenvolvimento



Melhorias esperadas

- Ao incentivar sprints de desenvolvimento ágeis com sprints de design e de código, experimentamos algumas melhorias importantes nos produtos que ajudamos a criar e nas práticas que usamos para lançá-las:
 - Recuperar meses e anos de esforços perdidos trabalhando em problemas com os quais as pessoas não se importam
 - Permitir que as organizações orientadas pela engenharia colaborem efetivamente com outras pessoas no processo de desenvolvimento de produtos
 - Capacite pessoas (papéis) fora do escopo direto de produção e tecnologia no processo de tomada de decisão
 - Forneça às equipes de engenharia a perspectiva empática e a voz do usuário que normalmente estão ausentes



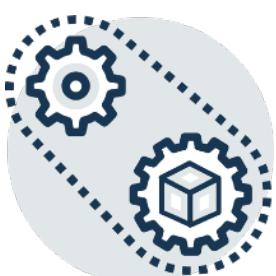
Sprints para Produtos Existentes

- Um erro que muitas equipes cometem ao adotar sprints de design é usá-las **em excesso**
- Começam a executar sprints de design para quase todos os recursos que desejam implantar...
 - Adicionando um formulário a uma página? "Vamos fazer um sprint".
 - Construindo a versão iOS do nosso aplicativo Android? "Sprint!"
- Um sprint de design destina-se melhor a um **problema que mapeia de volta a uma oportunidade ou desafio crítico de negócios**
 - Por exemplo, se sua empresa decide começar a oferecer seu produto para um segmento de clientes totalmente novo, essa é uma decisão essencial que deve ser canalizada por meio de um sprint de design



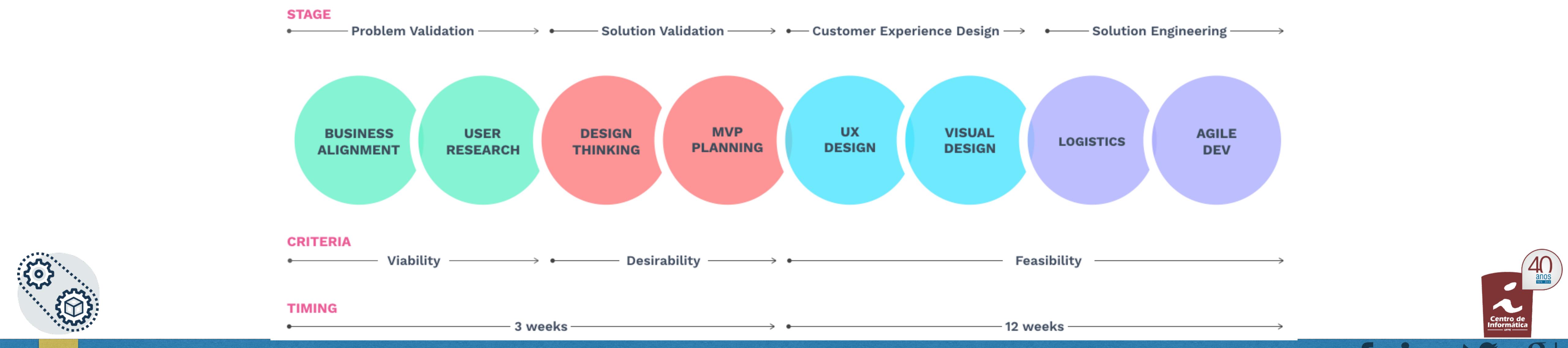
Como?

- Ter uma solução existente cria restrições de várias maneiras possíveis:
 - Nos força a entrar no modo de solução antes de entendermos completamente o problema
 - Ela influencia as soluções que criamos
- Se precisarmos criar algo em uma solução existente
 - Isso limita nossas soluções dentro dos limites dessa solução -
 - isso é particularmente desafiador quando essas soluções estão fora de nosso controle
- Por exemplo: plataformas, dados, acesso de terceiros...



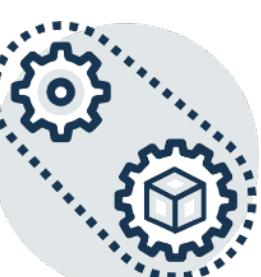
Como?

- Por outro lado, ter uma solução existente também nos fornece **parâmetros de referência** aos quais novos produtos não têm acesso
- Se você voltar ao diagrama, verá que a **validação do problema** é crítica para um sprint de design (validação da solução)
 - Isso geralmente vem na forma de pesquisa etnográfica, pesquisas, protótipos de papel...
 - No entanto, quando as soluções existentes anteriormente podem ser exploradas, você obtém uma fonte adicional de dados



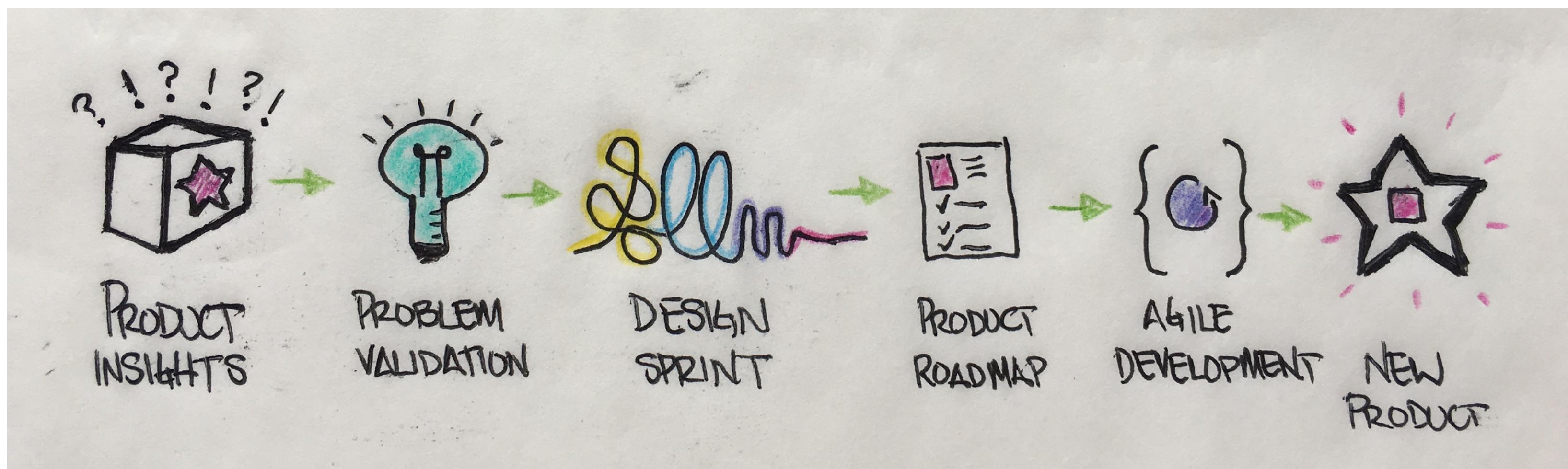
Como?

- O truque para alavancar os dados existentes da solução é **extrair as idéias**, sem o viés de implementação
 - Em outras palavras, os dados que você procura são mais sobre os padrões (bons ou ruins), com base no problema que a solução existente está tentando resolver
- Como essa solução foi implementada atualmente pode certamente ser notado, mas não deve influenciar fortemente um próximo sprint de design que visa descobrir novas possibilidades
 - Lembre-se de que a vantagem de um sprint de design é que você tem espaço para sonhar alto com soluções inteiramente novas, interessantes e viáveis
- Devido a esse viés herdado, muitas vezes significa que você deve pensar com muito cuidado sobre o uso de membros da equipe de implementações anteriores em soluções da próxima geração
 - O que tende a funcionar bem é alavancar os membros anteriores da equipe como especialistas externos



Benefícios

- Aproveitar informações críticas às quais novos produtos não têm acesso
- Novas soluções podem extrair (para replicar) ou criar experiências inteiramente novas das soluções existentes
- E como um produto existente encara uma sprint de design antes de avançar para o desenvolvimento ágil?



Leituras Futuras

- [Stop Brainstorming and Start Sprinting](#), by Jake Knapp
- [The Design Sprint - How it works?](#), by The Sprint Book
- [A Comprehensive Guide to Running Design Sprints](#), by Nick Babich
- [Exemplos de Design Sprints](#)
- [The Design Sprint 2.0: What is it and what does it look like?](#), by Jonathan Courtney
- [About Design Sprint 3.0](#), by Design Sprint Academy

