

Engenharia de Software Moderna

Cap. 3 - Requisitos

Prof. Marco Tulio Valente

<https://engsoftmoderna.info>, @engsoftmoderna

Licença CC-BY; permite copiar, distribuir, adaptar etc; porém, **créditos devem ser dados ao autor dos slides**

"A parte mais difícil da construção de um software é a definição do que se deve construir" -- Fred Brooks

Requisitos

- O que um sistema deve fazer e sob que restrições
- "o que um sistema deve fazer": **requisitos funcionais**
 - suas funcionalidades
- "sob que restrições": **requisitos não-funcionais**
 - Desempenho, usabilidade, segurança, disponibilidade, etc

Exemplo: sistema de um banco

Requisitos Funcionais

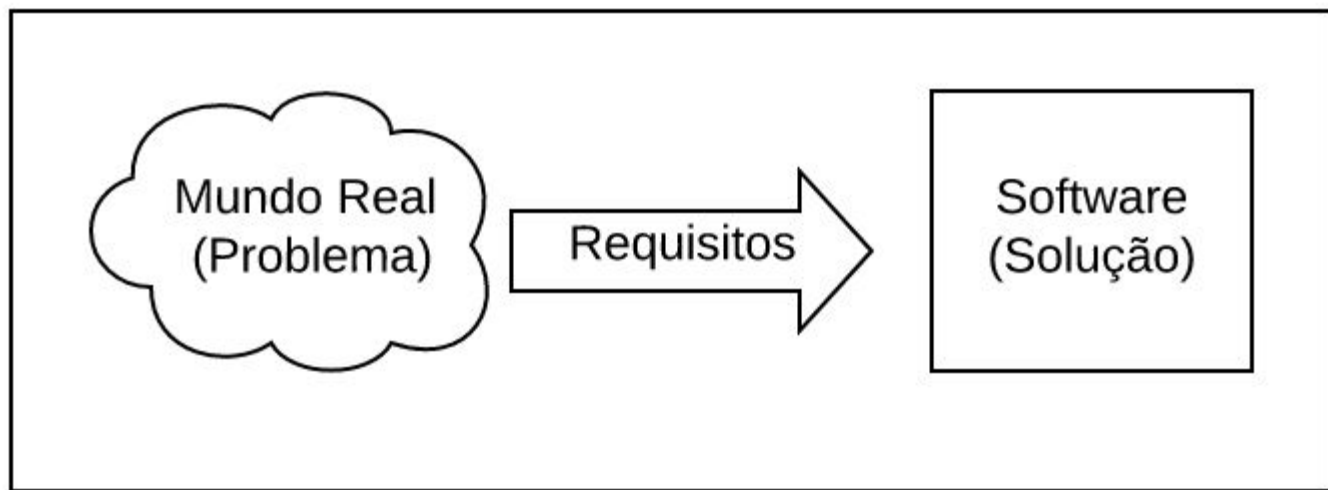
- Abrir e fechar conta
- Fornecer extrato e saldo
- Realizar transferência
- Realizar saque
- etc

Requisitos Não-Funcionais

- Desempenho: informar o saldo em menos de 5 segundos
- Disponibilidade: estar no ar 99.99% do tempo
- Tolerância a falhas: continuar operando se centro de dados cair
- Segurança: criptografar dados trocados com as agências
- Privacidade: não disponibilizar dados de clientes para terceiros

Requisitos Não-Funcionais

- Interoperabilidade: integrar-se com os sistemas do BACEN
- Capacidade: armazenar dados de 1 milhão de clientes
- Usabilidade: ter uma versão para deficientes visuais



Principais Desafios (survey com 228 empresas, 16 países)

- Requisitos incompletos ou não-documentados (48%)
- Falhas de comunicação entre membros do time e os clientes (41%)
- Requisitos em constante mudança (33%)
- Requisitos especificados de forma abstrata (33%)
- Restrições de tempo (32%)
- Problemas de comunicação entre membros do time (27%)
- Stakeholders com dificuldade de separar requisitos e soluções (25%)
- Falta de apoio dos clientes (20%)

Elicitação de Requisitos

- Elicitar (ou eliciar): "fazer sair, expulsar, expelir"
- Elicitação de Requisitos
 - "Fazer sair os requisitos do sistema"
 - Isto é, descobrir e entender os principais requisitos do sistema que se pretende construir.

O que vamos estudar?

- Histórias de usuários
- Casos de Uso
- Produto Mínimo Viável (MVP)
- Testes A/B

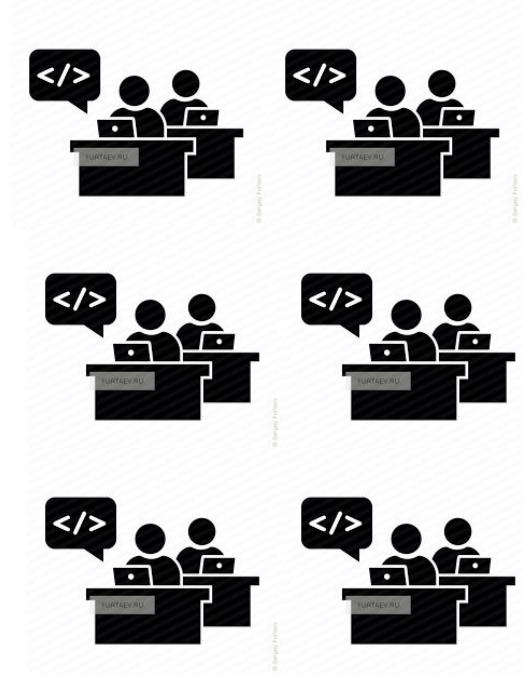
Histórias de Usuários

Antes ...



Analistas

(PRD: program requirement document)



Programadores
(Fábrica de Software)

Hoje ...



Product Owner senta junto dos desenvolvedores e
"explica" requisitos para eles

Histórias de Usuários = 3C's

- **C**artão + **C**onversas + **C**onfirmação
- Cartão: "lembrete" para conversar sobre o requisito durante o sprint
- Confirmação: cenários que serão usados pelo PO para aceitar a implementação da história (escritos no verso do cartão)

Exemplo: Loja Virtual

- Cartão: "Fechar uma compra"
- Conversas: PO explica os meios de pagamento; as formas de entrega, formas de parcelamento, etc
- Confirmação:
 - Testar compra à vista e compra parcelado
 - Testar com cartões de crédito A, B e C
 - Testar com modos de entrega X e Y

Kanban/Scrum Board



Quem escreve as histórias?

1. PO
2. Workshop de escrita de histórias
 - Realizado no início do projeto
 - Objetivo: criar uma lista inicial de histórias
 - Participam representantes dos principais usuários

Formato escrita de histórias

- Como um [certo tipo de usuário],
eu gostaria de [realizar algo com o sistema]

Exemplo: sistema de controle de bibliotecas

3 tipos de usuários

- Usuário típico
- Professor
- Funcionário da biblioteca

Histórias do usuário típico

- Como usuário típico, eu gostaria de realizar empréstimos de livros
- Como usuário típico, eu gostaria de devolver um livro que tomei emprestado
- Como usuário típico, eu gostaria de renovar empréstimos de livros
- Como usuário típico, eu gostaria de pesquisar por livros
- Como usuário típico, eu gostaria de reservar livros que estão emprestados
- Como usuário típico, eu gostaria de receber e-mails com novas aquisições

Histórias de professores

- Como professor, eu gostaria de realizar empréstimos de maior duração
- Como professor, eu gostaria de sugerir a compra de livros
- Como professor, eu gostaria de doar livros para a biblioteca
- Como professor, eu gostaria de devolver livros em outras bibliotecas

Histórias de funcionários da biblioteca

- Como funcionário, eu gostaria de cadastrar novos usuários
- Como funcionário, eu gostaria de cadastrar novos livros
- Como funcionário, eu gostaria de dar baixa em livros estragados
- Como funcionário, eu gostaria de obter estatísticas sobre o acervo
- Como funcionário, eu gostaria que o sistema envie e-mails de cobrança para alunos com empréstimos atrasados
- Como funcionário, eu gostaria que o sistema aplicasse multas quando da devolução de empréstimos atrasados

Características de boas histórias (INVEST)

- Independentes
- Abertas para Negociação
- Agregar Valor
- Estimáveis
- Sucintas
- Testáveis

Histórias ⇔ Requisitos Funcionais

E os requisitos não-funcionais?

Requisitos não Funcionais (RNF)

- Time deve definir RNF com o Product Owner
- Time deve incluir RNF nos critérios de conclusão de um sprint (done criteria)

Exemplo

- Suponha que desempenho seja um RNF importante
- Pode-se definir que história para ser considerada **pronta** deve:
 - Passar por uma revisão de código focada em desempenho
 - Passar por testes de desempenho, com carga real

Casos de Uso

Casos de Uso

- Documento mais detalhado de especificação de requisitos
- Uso **não** é tão comum com métodos ágeis
- Um ator realizando alguma operação com o sistema
- Incluem fluxo normal e extensões
- Extensões:
 - Exceções (ou erros)
 - Detalhamento

Transferir Valores entre Contas

Ator: Cliente do Banco

Fluxo normal:

1 - Autenticar Cliente

2 - Cliente informa agência e conta de destino da transferência

3 - Cliente informa valor que deseja transferir

4 - Cliente informa a data em que pretende realizar a operação

5 - Sistema efetua transferência

6 - Sistema pergunta se o cliente deseja realizar uma nova transferência

Extensões:

2a - Se conta e agência incorretas, solicitar nova conta e agência

3a - Se valor acima do saldo atual, solicitar novo valor

4a - Data informada deve ser a data atual ou no máximo um ano a frente

5a - Se data informada é a data atual, transferir imediatamente

5b - Se data informada é uma data futura, agendar transferência

Transferir Valores entre Contas

Ator: Cliente do Banco

Fluxo normal:

- 1 - Autenticar Cliente
- 2 - Cliente informa agência e conta de destino da transferência
- 3 - Cliente informa valor que deseja transferir
- 4 - Cliente informa a data em que pretende realizar a operação
- 5 - Sistema efetua transferência
- 6 - Sistema pergunta se o cliente deseja realizar uma nova transferência

Extensões:

- 2a - Se conta e agência incorretas, solicitar nova conta e agência
- 3a - Se valor acima do saldo atual, solicitar novo valor
- 4a - Data informada deve ser a data atual ou no máximo um ano a frente
- 5a - Se data informada é a data atual, transferir imediatamente
- 5b - Se data informada é uma data futura, agendar transferência

Transferir Valores entre Contas

Ator: Cliente do Banco

Fluxo normal:

- 1 - Autenticar Cliente
- 2 - Cliente informa agência e conta de destino da transferência
- 3 - Cliente informa valor que deseja transferir
- 4 - Cliente informa a data em que pretende realizar a operação
- 5 - Sistema efetua transferência
- 6 - Sistema pergunta se o cliente deseja realizar uma nova transferência

Extensões:

- 2a - Se conta e agência incorretas, solicitar nova conta e agência
- 3a - Se valor acima do saldo atual, solicitar novo valor
- 4a - Data informada deve ser a data atual ou no máximo um ano a frente
- 5a - Se data informada é a data atual, transferir imediatamente
- 5b - Se data informada é uma data futura, agendar transferência

Transferir Valores entre Contas

Ator: Cliente do Banco


Fluxo normal:

- 1 - Autenticar Cliente
- 2 - Cliente informa agência e conta de destino da transferência
- 3 - Cliente informa valor que deseja transferir
- 4 - Cliente informa a data em que pretende realizar a operação
- 5 - Sistema efetua transferência
- 6 - Sistema pergunta se o cliente deseja realizar uma nova transferência

Extensões:

- 2a - Se conta e agência incorretas, solicitar nova conta e agência
- 3a - Se valor acima do saldo atual, solicitar novo valor
- 4a - Data informada deve ser a data atual ou no máximo um ano a frente
- 5a - Se data informada é a data atual, transferir imediatamente
- 5b - Se data informada é uma data futura, agendar transferência

Fluxo "Feliz"



Transferir Valores entre Contas

Ator: Cliente do Banco

Fluxo normal:

- 1 - Autenticar Cliente
- 2 - Cliente informa agência e conta de destino da transferência
- 3 - Cliente informa valor que deseja transferir
- 4 - Cliente informa a data em que pretende realizar a operação
- 5 - Sistema efetua transferência
- 6 - Sistema pergunta se o cliente deseja realizar uma nova transferência

Extensões:

- 2a - Se conta e agência incorretas, solicitar nova conta e agência
- 3a - Se valor acima do saldo atual, solicitar novo valor
- 4a - Data informada deve ser a data atual ou no máximo um ano a frente
- 5a - Se data informada é a data atual, transferir imediatamente
- 5b - Se data informada é uma data futura, agendar transferência

Exceções e
Detalhamentos



Transferir Valores entre Contas

Ator: Cliente do Banco

Fluxo normal:

- 1 - Autenticar Cliente
- 2 - Cliente informa agência e conta de destino da transferência
- 3 - Cliente informa valor que deseja transferir
- 4 - Cliente informa a data em que pretende realizar a operação
- 5 - Sistema efetua transferência
- 6 - Sistema pergunta se o cliente deseja realizar uma nova transferência

Extensões:

- 2a - Se conta e agência incorretas, solicitar nova conta e agência
- 3a - Se valor acima do saldo atual, solicitar novo valor
- 4a - Data informada deve ser a data atual ou no máximo um ano a frente
- 5a - Se data informada é a data atual, transferir imediatamente
- 5b - Se data informada é uma data futura, agendar transferência

Erro (passo 2)

Transferir Valores entre Contas

Ator: Cliente do Banco

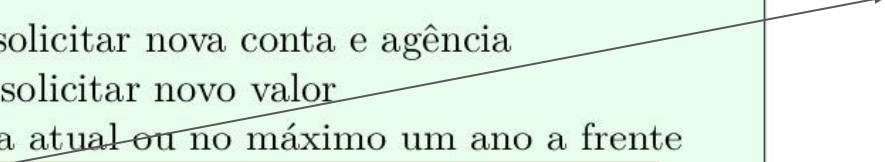
Fluxo normal:

- 1 - Autenticar Cliente
- 2 - Cliente informa agência e conta de destino da transferência
- 3 - Cliente informa valor que deseja transferir
- 4 - Cliente informa a data em que pretende realizar a operação
- 5 - Sistema efetua transferência
- 6 - Sistema pergunta se o cliente deseja realizar uma nova transferência

Extensões:

- 2a - Se conta e agência incorretas, solicitar nova conta e agência
- 3a - Se valor acima do saldo atual, solicitar novo valor
- 4a - Data informada deve ser a data atual ou no máximo um ano a frente
- 5a - Se data informada é a data atual, transferir imediatamente
- 5b - Se data informada é uma data futura, agendar transferência

Detalhamento
(passo 5)



Importante

- Casos de uso não são "algoritmos"
- Ainda estamos levantando requisitos:
 - Foco: entendimento e delimitação do problema
 - E não em possíveis soluções (i.e., algoritmos)

Exemplo: Venda em Caixa de Supermercado (PDV)

Fonte: Craig Larman. Applying UML and Patterns. Pearson, 2004

Fluxo Normal

1. Cliente chega no caixa com os produtos que deseja comprar
2. Caixa inicia uma nova venda
3. Caixa identifica um produto; por exemplo, usando leitor de código de barras
4. Sistema identifica produto, registra venda, apresenta a descrição do produto e seu preço, bem como o total da compra até o momento
5. Caixa repete passos 3-4 até não haver mais produtos para registrar venda
6. Sistema apresenta total da venda
7. Caixa informa total da venda para o cliente e pede o pagamento
8. Cliente faz o pagamento e o sistema processa o pagamento
9. Sistema registra a venda como concluída e envia informações para o sistema de contabilidade e para o sistema de controle de estoques
10. Sistema gera recibo da venda
11. Caixa entrega recibo para o cliente
12. Cliente encerra a compra, levando os produtos e o seu recibo

Extensões (ou fluxos alternativos): [vamos mostrar para apenas um dos passos do fluxo normal]

7a. Pagamento em dinheiro:

1. Caixa digita o montante de dinheiro que o cliente lhe forneceu
2. Sistema informa o valor do "troco" e libera a gaveta de notas
3. Caixa deposita o dinheiro na gaveta e retorna "troco" para o cliente Cliente
4. Sistema registra e conclui pagamento com dinheiro

7b. Pagamento com cartão de crédito:

1. Cliente insere cartão na máquina de cartão de crédito
2. Sistema informa para máquina de cartão o valor da compra
3. Cliente informa senha e confirma compra
4. Sistema envia solicitação de pagamento para operadora do cartão
 - 4a. Se erro de comunicação com o sistema da operadora do cartão
 1. Sistema sinaliza erro para o Caixa
 2. Caixa solicita ao Cliente um modo alternativo de pagamento

(continua no próximo slide)

- 5. Sistema recebe resultado da requisição de pagamento
 - 5a. Pagamento negado
 - 1. Sistema avisa o Caixa
 - 2. Caixa solicita ao Cliente um modo alternativo de pagamento
 - 5b. Timeout na espera pelo resultado da requisição de pagamento
 - 1. Sistema avisa o Caixa
 - 2. Caixa tenta de novo ou solicita modo alternativo de pagamento
- 6. Sistema registra e conclui pagamento com cartão de crédito

7c. Pagamento com cheque

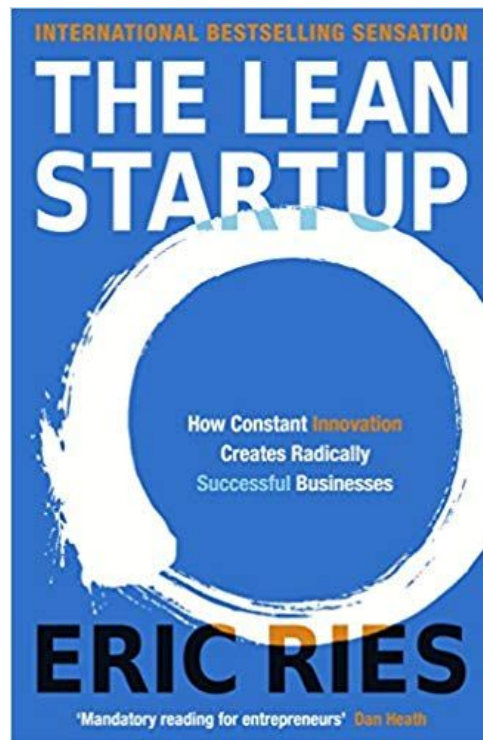
....

7d. Pagamento com cartão de débito

....

Produto Mínimo Viável

Origem



Dois tipos de sistemas

1. Baixo risco e usuários conhecidos
2. Alto risco e "sucesso" incerto

Baixo risco e usuários conhecidos

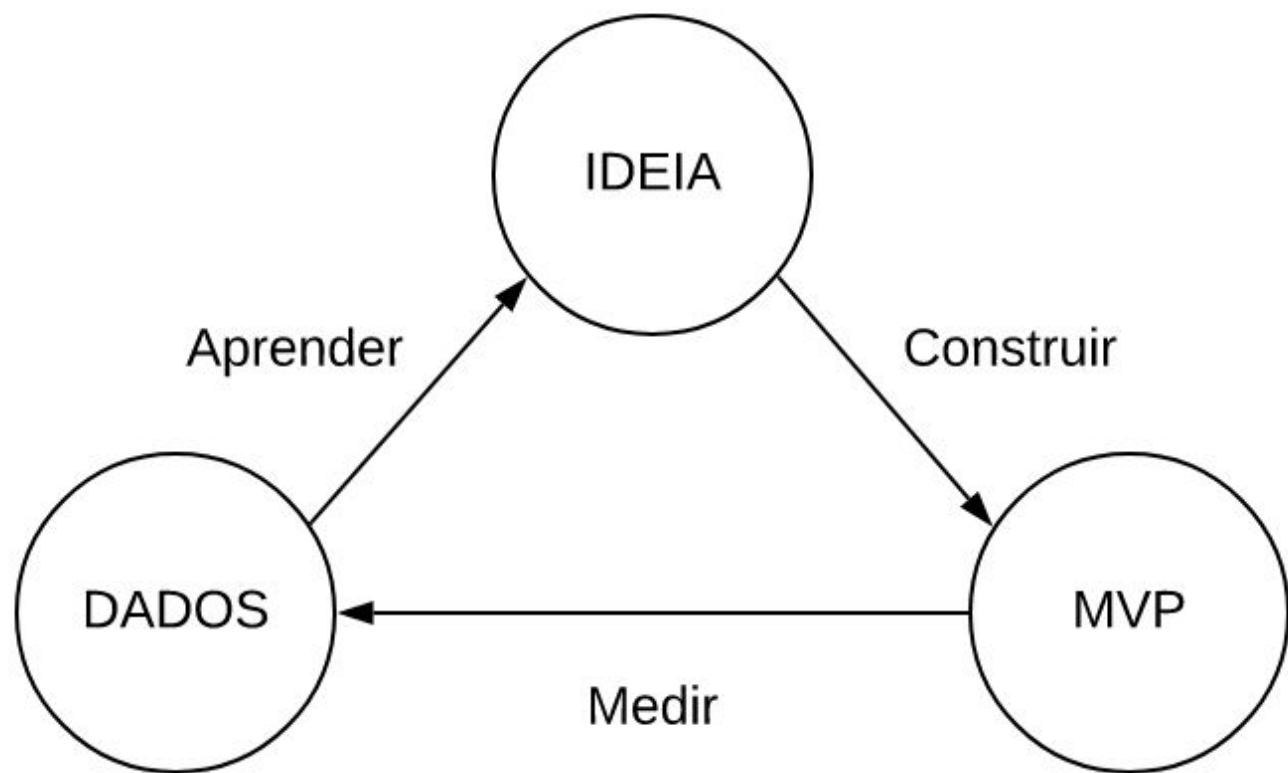
- Exemplo: sistema de controle de bibliotecas
- Sistema conhecido, fundamental em toda biblioteca, etc
- Necessidade e viabilidade desse sistema são óbvias
- Histórias de usuários funcionam bem!

Alto risco e "sucesso" incerto

- Exemplo: loja virtual para empréstimo de livros digitais com pagamento via blockchain
- Sistemas típicos de startups, mas não exclusivos
- Como risco é alto, implementação (ideia) tem que ser rapidamente validada com usuários reais

Produto Mínimo Viável (MVP)

- Produto = já pode ser usado
- Mínimo = menor conjunto de funcionalidades (menor custo)
- Viável = objetivo é obter dados e validar uma ideia



Ciclo Construir-Medir-Aprender

- Ao final desse ciclo, pode-se:
 - Realizar alguns ajustes e rodar ciclo de novo
 - Pivotar: mudar foco do produto (ex.: aluguel de músicas)
 - Desistir (dinheiro acabou!)
 - Deu certo! Vamos construir um produto mais robusto

MVP não precisa ser um software completo

Zappos

- Loja virtual de sapatos, depois adquirida pela Amazon
- As pessoas vão comprar sapatos pela Internet ? (em 1999)
- MVP:
 - Sistema Web simples
 - Com fotos de sapatos de lojas físicas da cidade
 - "Backend" era 100% manual
- Objetivo: apenas validar hipótese de negócio



women's

men's

dress

casual

athletic

dress

casual

athletic

kids

Pick a category
to shop from:

Category

Register now &
Save Money

registered customers

username

password

login



Measure Your Foot

The world's largest shoe store!

WHAT WE'RE HEARING!

Welcome to Zappos.com - the shoe store! We have a selection of over 100 brands to shop from! We offer FREE SHIPPING (U.S. orders only) and NO SALES TAX.

Live Customer Service!
Mon. - Fri. 10am-6pm PST
Sat. 10am-5pm PST
Click here!

Free Shoes!



Congratulations to the November 24 winner,
Adriana Rodriguez of Oakland, CA

We will be giving away a FREE PAIR OF SHOES (up to \$150 in store credit) every Wednesday until the year 2000!

To enter become a registered user or send an email to freeshoes@zappos.com with your name and email address. A winner will be notified by email every Wednesday!

featured brands.



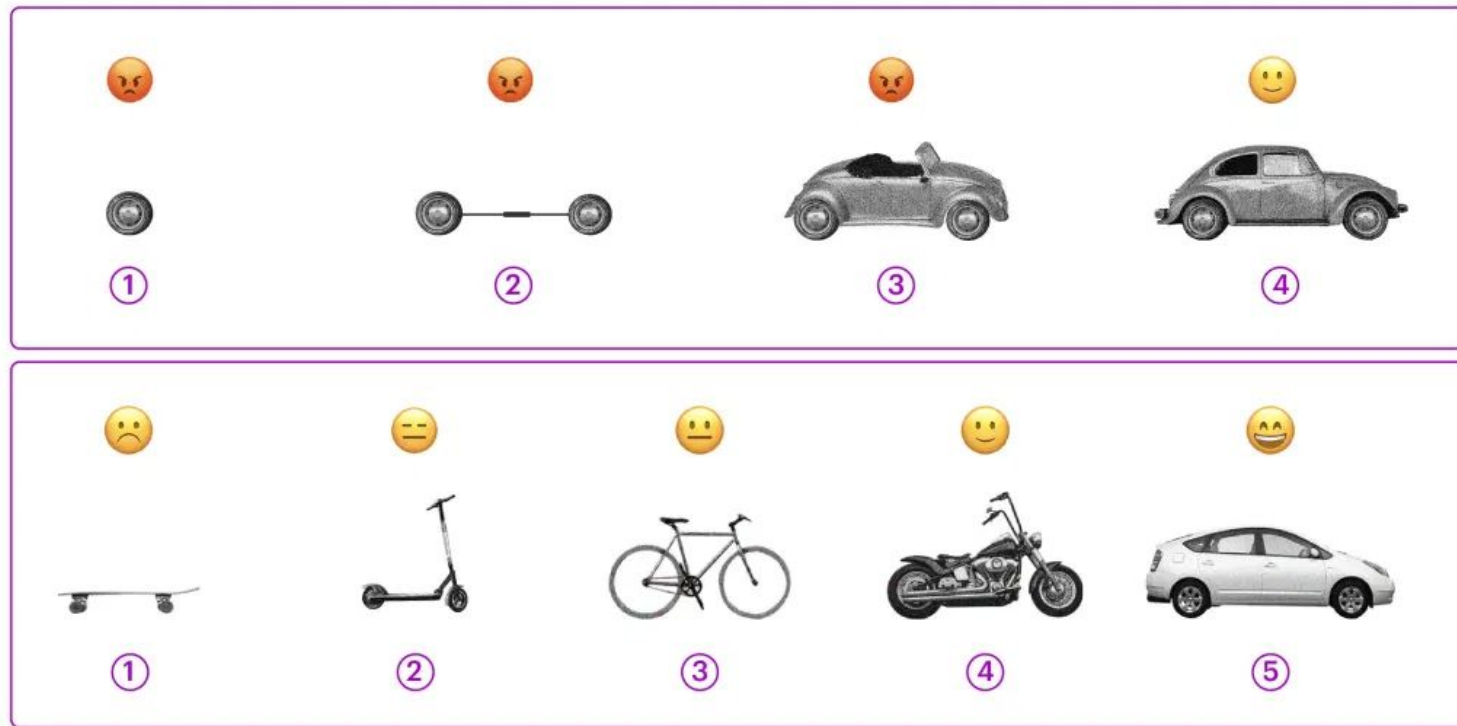
Zappos Special:

Free Shipping &
No Sales Tax

MVP do Dropbox (apenas um vídeo para atrair usuários para a versão beta)



Implementando um carro (preste atenção também na fisionomia do cliente)



Sem
MVP

Com
MVP

MVP & Engenharia de Software

- MVP não precisa usar todas as melhores práticas de ES
 - Testes de unidade, refatorações, etc
- Se a ideia der certo, pode-se depois reimplementar o sistema
- Porém, certos requisitos, principalmente NF, são importantes
 - Desempenho, usabilidade, estabilidade, etc

Testes A/B

Testes A/B

- Existe o cenário onde duas implementações de requisitos "competem" entre si
- Exemplo: loja virtual
- Sistema recomendação: quem compra P também compra X,Y,Z
 - Versão original
 - Nova versão, proposta por alguns devs
- Vale a pena migrar para nova versão?
- Testes A/B: deixar os dados decidirem

Versões de controle e tratamento

- Versão A: controle
- Versão B: tratamento
- Novos usuários:

```
version = Math.Random(); // número aleatório entre 0 e 1
if (version < 0.5)
    "execute a versão de controle"
else
    "execute a versão de tratamento"
```

No final do experimento

- Analisam-se os dados, isto é, alguma métrica
- Exemplo: taxa de conversão de visitas em compras
- Versão B introduz ganhos **estatisticamente** relevantes?
 - Sim, vamos migrar para ela
 - Não, continuamos com o algoritmo original

Calculadoras de tamanho de amostras

- Informa-se:
 - Taxa de conversão atual (1%)
 - Ganho pretendido (10%)
- Calculadora informa:
 - Tamanho da amostra necessário: 200K por versão

Comentários finais

- Testes A/B requerem amostras grandes
- São usados por todas as grandes empresas da Internet (veja comentários no livro).

Fim