



TDC2013

the developer's conference



Lições do Futuro: O que eu queria saber há alguns anos atrás sobre como manter uma suíte de testes



 @ederign



 @flsususp



CRAFTING SOFTWARE

transforming
BUSINESS

Confidence

Câmbio

***Maior grupo financeiro de
câmbio do Brasil***



+120 agências
adm de cartões e banco



+1.000.000

clientes **por ano**

13 em cada 100
dolares que **sairam do Brasil**

...Confidence

busca empresa para desenvolver sua
nova **plataforma operacional**





**IF YOU WORK
REALLY HARD AND
ARE KIND, AMAZING
THINGS WILL HAPPEN.**



quinta-feira, 11 de julho de 2013

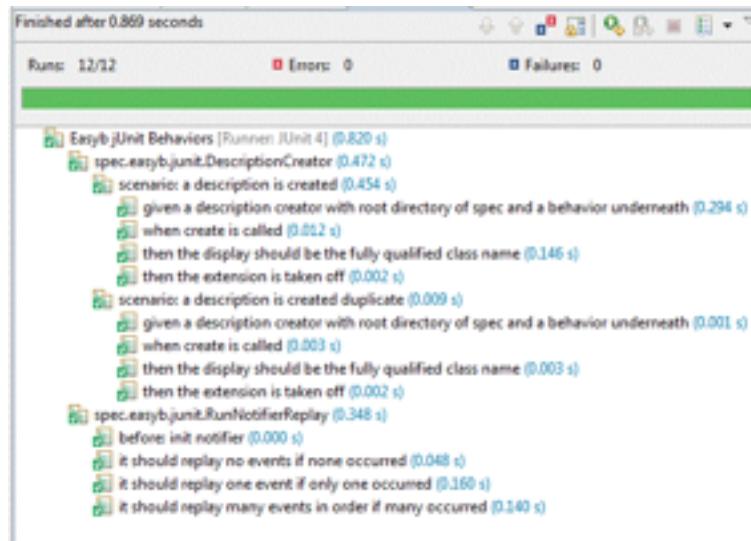
Qualidade não é negociável!!!

Você tem o direito de fazer certo!



Testes Unitários

Testes de Integração



BDD

Desenvolvimento de Framework Próprio



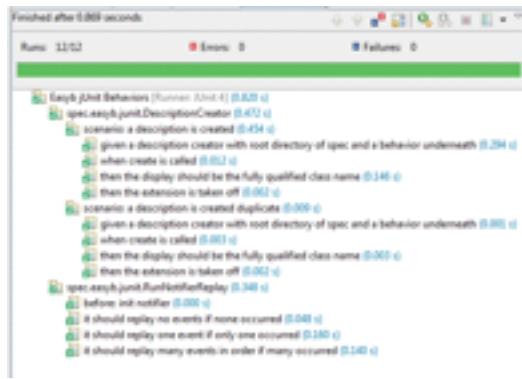
Testes Funcionais



Abandono Testes Integração



BDD + Testes Funcionais



+





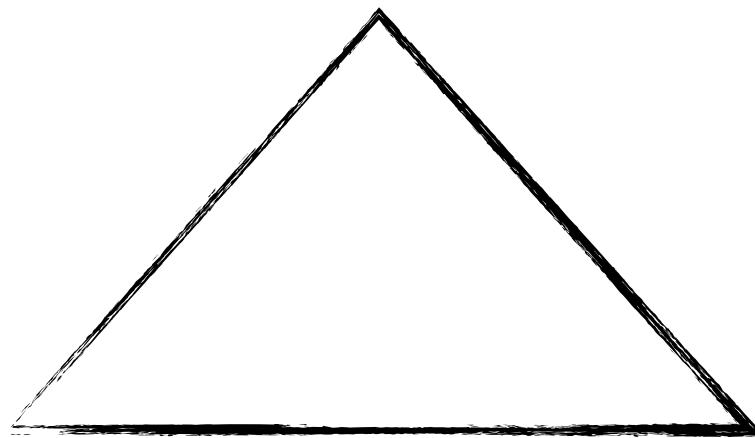
Alta taxa de Cobertura

Cliente Feliz

4 anos de projeto (~12 pessoas)



Parceria TI e Business



**Planejamento
focado na
mitigação de
riscos**

**Arquitetura
Sólida e
escalável**

Deploy ~1x/Semana

Plataforma Bancária



“Continuous Delivery”

#osDevPira



Números do Projeto

~1 Release por Semana

1.6k Testes de Aceitação

160k Operações/mês

400k Requests/dia

1k Usuários em dias normais

17 Nós no Cluster

Explosão na Quantidade de Testes Funcionais



Lentidão na execução da suíte de testes



Time Triste



Onde erramos?

**Como saber se minha suíte de testes não
está indo para o buraco?**



Bugs em Produção

Seus testes não passam?

Você não tem testes?



Bugs em Produção



Seus testes passam?

Você não está testando a funcionalidade certa!

Você não está testando o sistema correto

Teste quebra



Em produção funciona?

Você não está testando o sistema certo!

Funcionalidade é alterada

Seu teste passa sem precisar de alteração?

Você não está testando a funcionalidade correta!



Você corrige um defeito

Seu teste passa a quebrar?

*Você não está focando seus testes na funcionalidade
e sim no sistema*



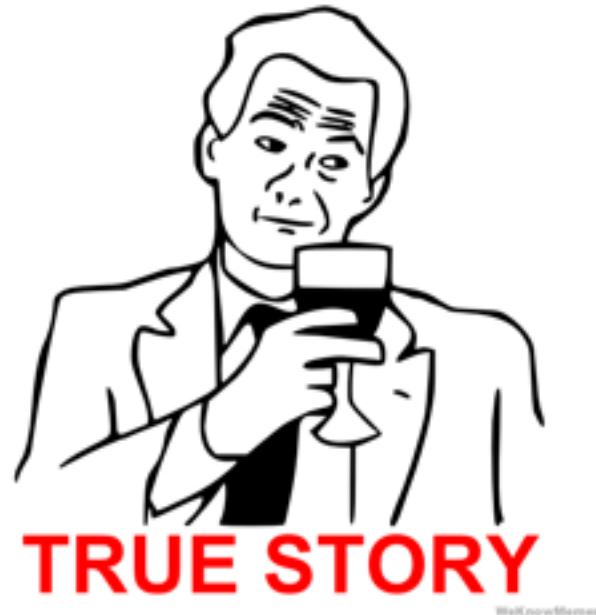
Você altera o sistema

Não tem feedback rápido dos testes?



*Você tem um problema arquitetural nos
seus testes!*

*Você não vai rodar a sua suíte!
(sob pressão)*



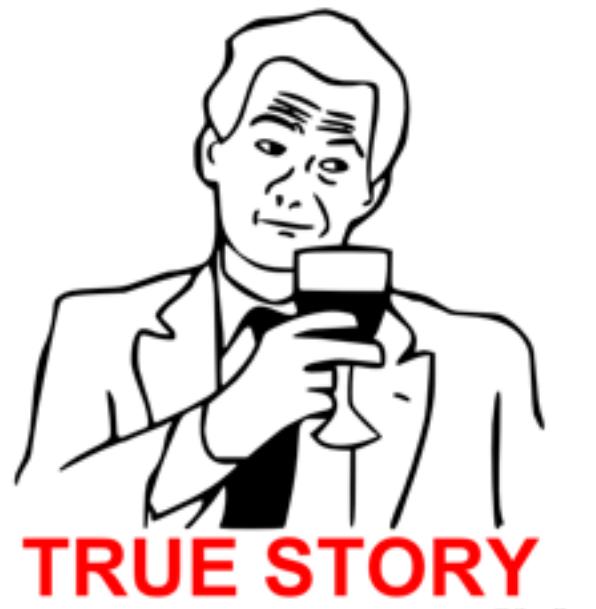
Você faz uma pequena alteração no sistema

*É difícil identificar um teste que possa ser rodado
agora?*

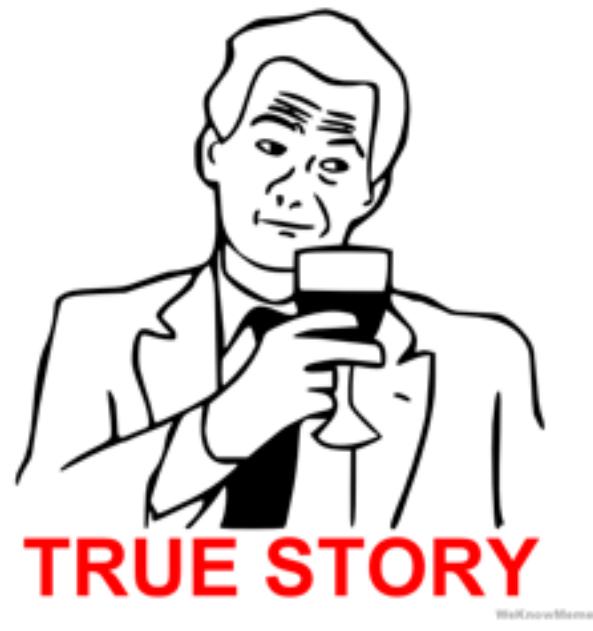
É difícil identificar qual teste foi afetado?

Você não vai testar a funcionalidade e nem rodar a suíte completa porque ela é lenta

Você vai implantar com a suíte quebrada



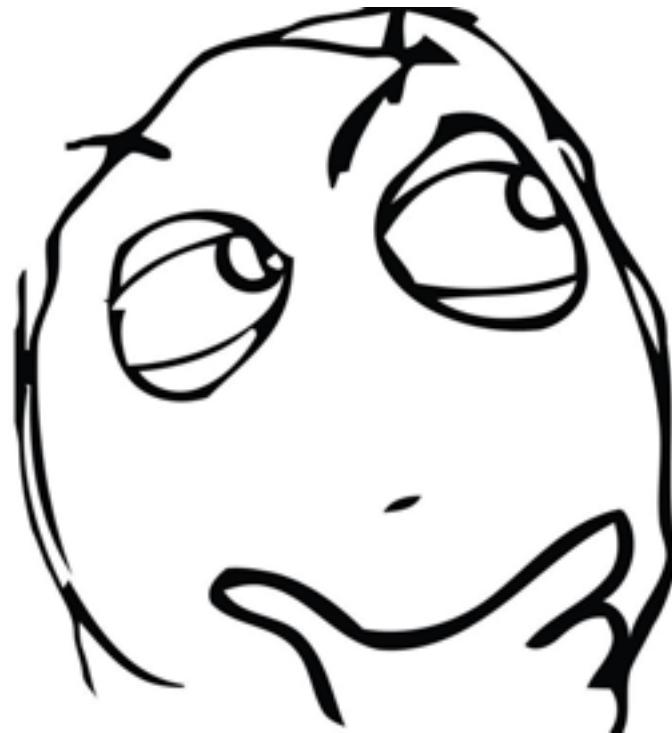
Pequena alteração, muitos testes quebram?



Você vai demorar para identificar o que quebrou

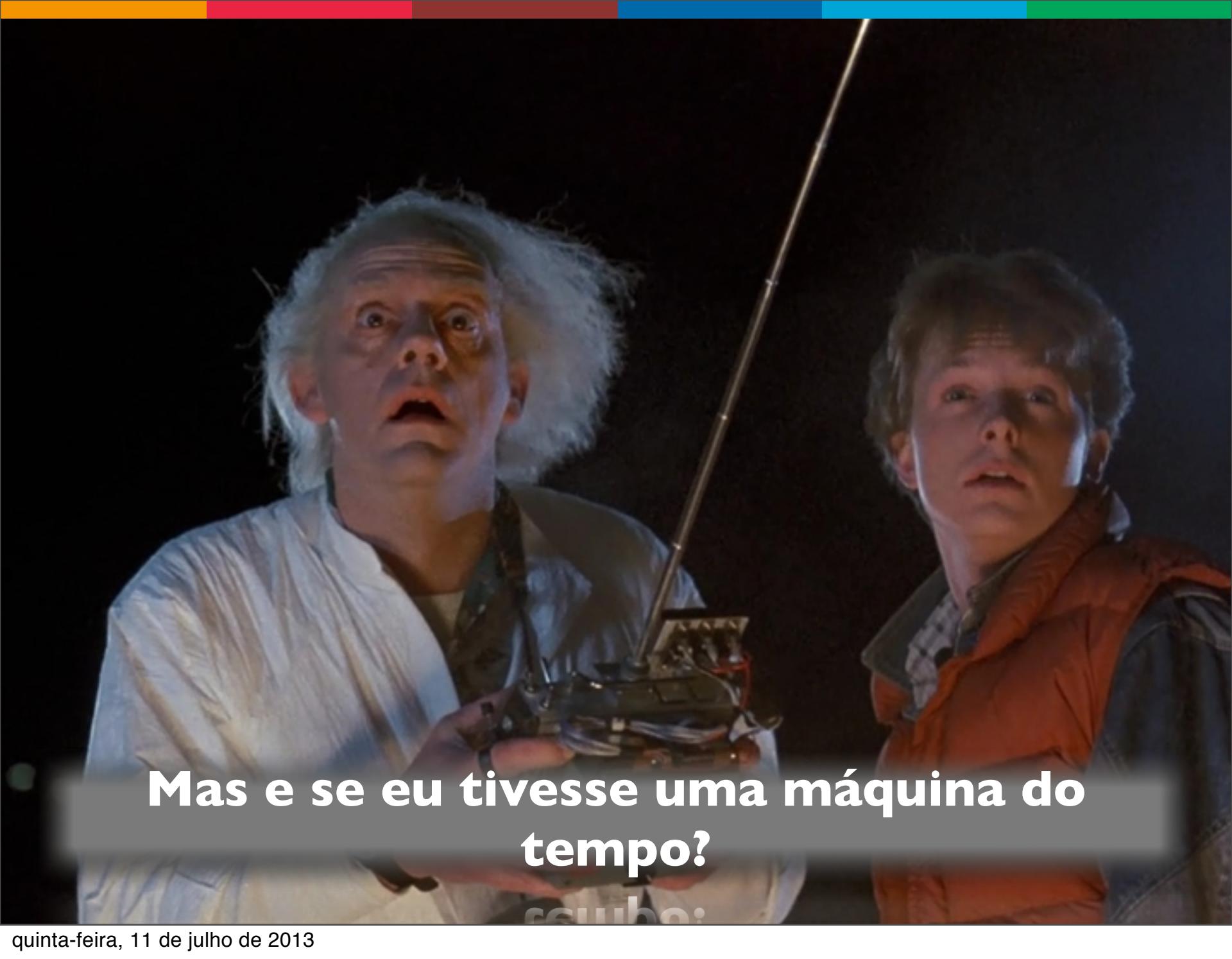
Você vai implantar em produção com a sua suíte quebrada

Mas quais são as características de uma boa suíte de testes?



***Uma boa suíte de testes é qualquer
suíte de testes que não tenha os
problemas apresentados!***





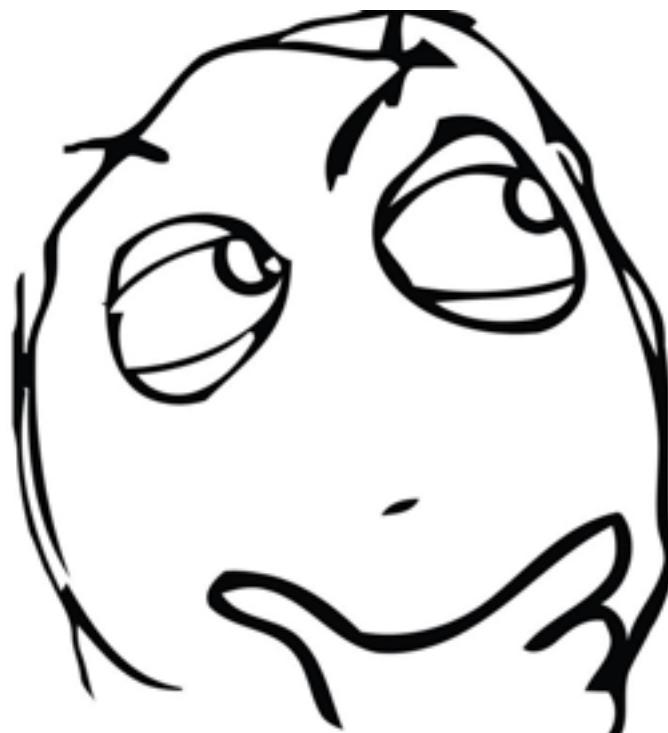
Mas e se eu tivesse uma máquina do tempo?

ccwibc:

Não existe bala de prata



Mas o que talvez (provavelmente) funcione?



3 F's - Corey Haines

Fast

Ciclo TDD - < 1 min

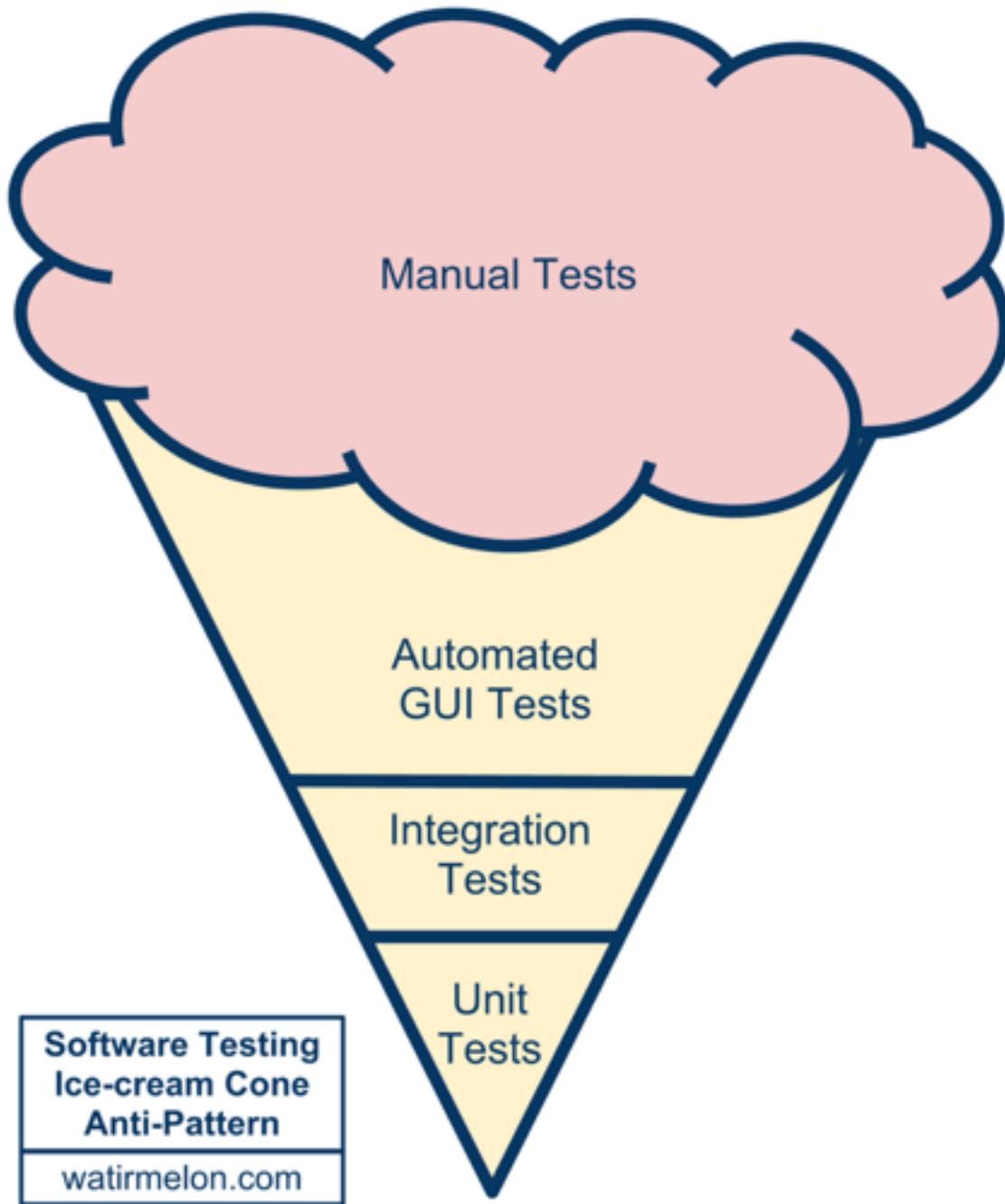
Ciclo Integração - rápido o suficiente
Suite Lenta == Builds Quebradas

Focused

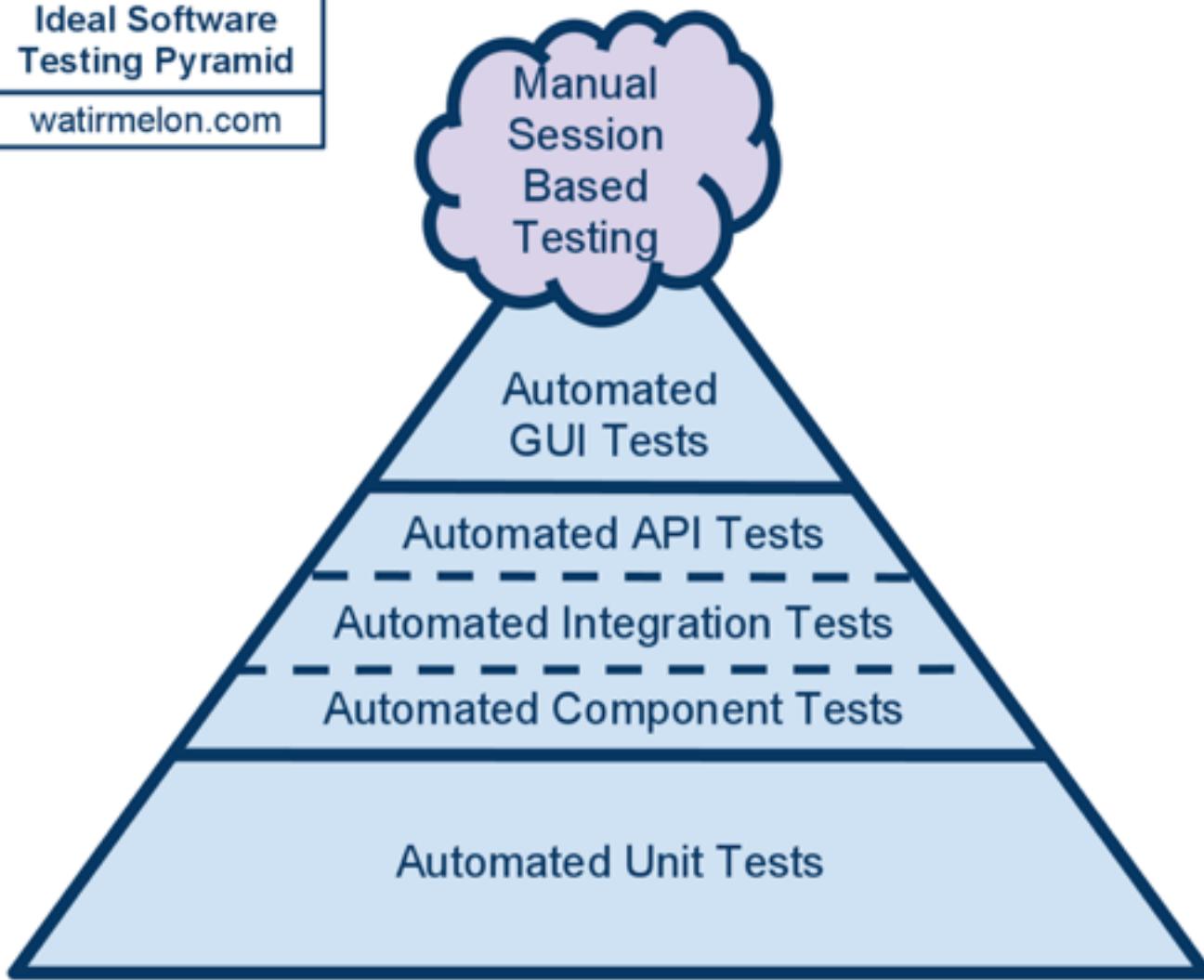
Um Teste = Uma Razão pra Mudança
Feedback Rápido

Full

Cobrir o maior % possível do sistema



Ideal Software
Testing Pyramid
watirmelon.com



Código de teste é tão importante quanto código de produção

Testes bem escritos e em grupos pequenos no mesmo contexto

Toda a equipe deve escrever a suíte de testes



Consistente e Confiável

A sua suíte de testes deve conquistar a confiança da sua equipe, do seu gerente e do cliente



“Isolada”

Um teste não pode influenciar o resultado de outro

Feedback Rápido



Duráveis

Mantém seu valor durante toda a vida do produto



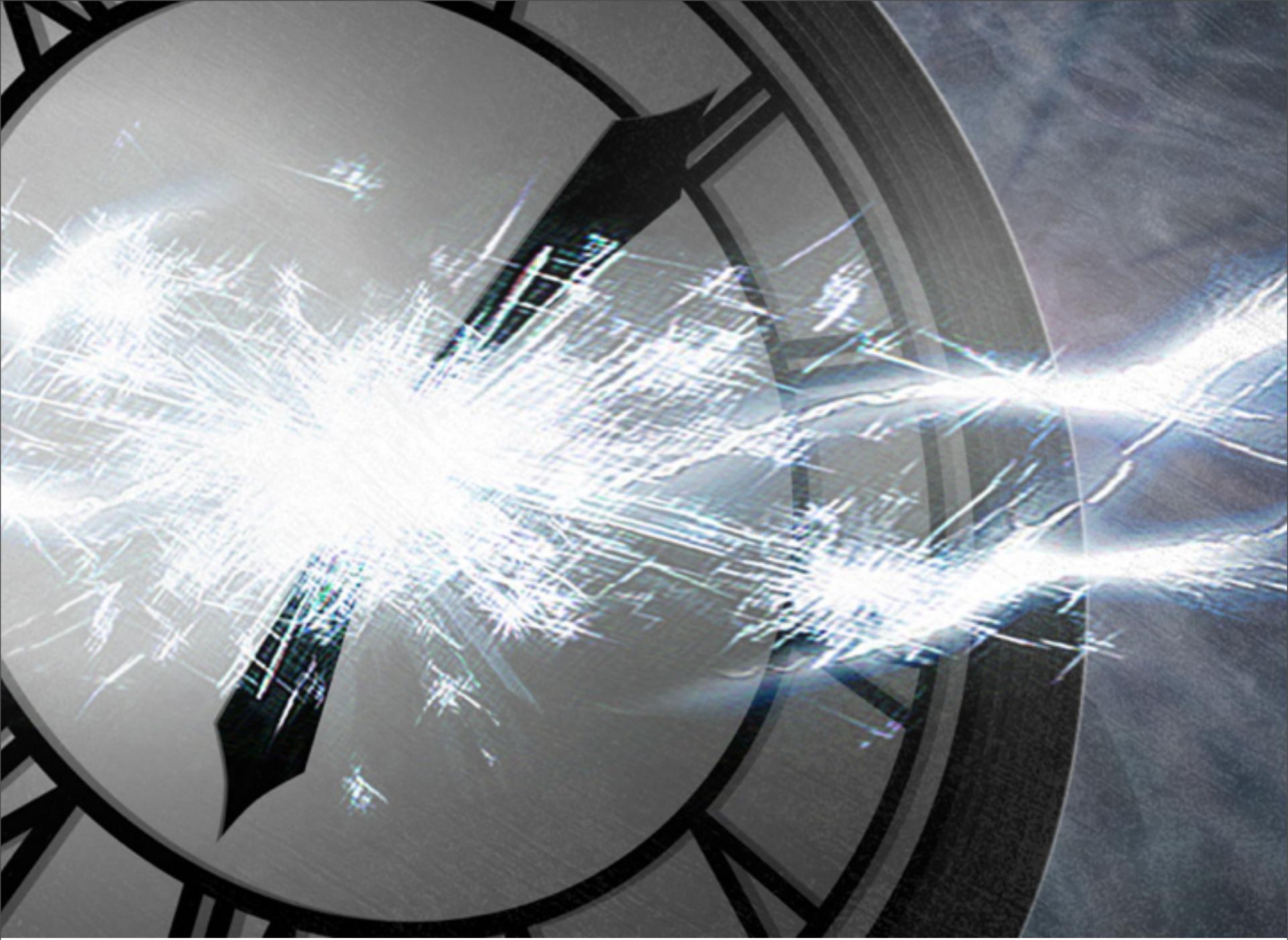
Documentação



*Testes servem como documentação
(são a melhor documentação que
você vai ter)*

*Documentação viva, especificação
executável.*

*Testes devem informar a intenção de
um desenvolvedor*



Opção I

Máquina gigante na Amazon, rodando Windows ou Linux com XVFB + Mycontainer

Servidor dedicado no dev

Pros: Setup parecido

Cons: \$, difícil de debugar (aws)



Opção II



myContainer Grid

Pros: Desenvolvido para paralelismo

Cons: Difícil de adaptar para paralelismo, difícil de adaptar meu projeto, projeto em construção

[https://github.com/feroult/
mycontainer-grid](https://github.com/feroult/mycontainer-grid)



Opção III

Selenium Grid

Pros: Fácil e estável
Cons: Difícil paralelizar na mesma máquina (ex. BD)



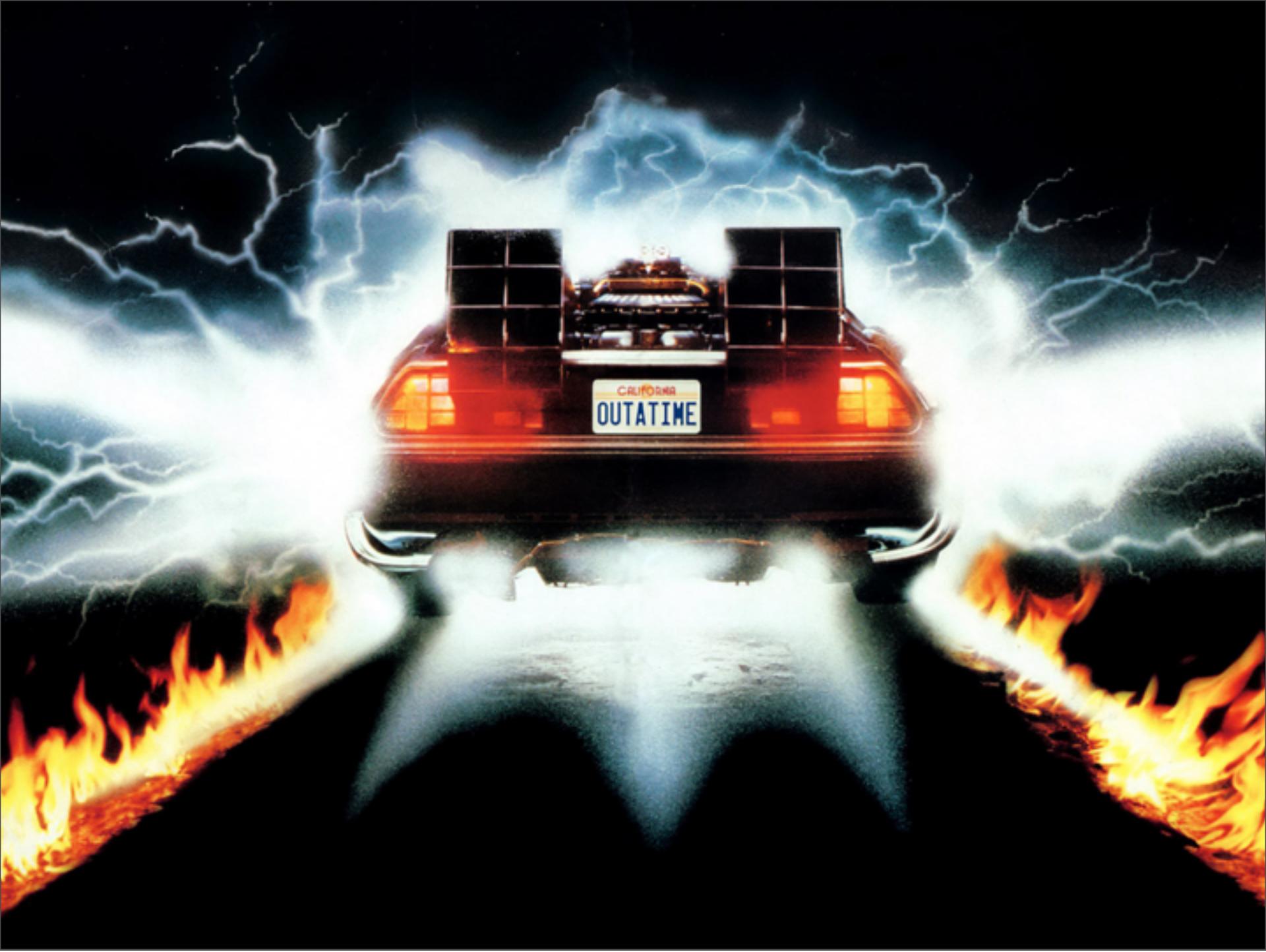
Opção IV

CloudBees + Sauce Labs

Pros: Cloud, ;)

Cons: Código de produção em servidor externo

@leguimas <- pode te ajudar



quinta-feira, 11 de julho de 2013



*Foque a cobertura nas
funcionalidades que mudam com
maior frequência*

*Faça testes para os bugs em
produção - você automaticamente
estará testando o mais crítico*



Não deixe o teste para o final

Não rode a suíte somente antes de gerar a versão

Faça integração contínua

Não se preocupe com a performance de execução de um teste, mas sim, com o paralelismo e concorrência.



*Mas o que é mais
importante?*

Possuir uma suíte de testes



Saber se sua suíte está indo para o buraco





TDC2013

the developer's conference



@ederign



@flsususp