

Unindo Especificação Ágil e Automação de Testes Funcionais com Concordia

THIAGO DELGADO PINTO

V O P E N L A B S

2 4 / 1 1 / 2 0 1 8



Licença Creative Commons 4



github.com/thiagodp



Professor do CEFET/RJ Nova Friburgo, desde 2009



Doutor em Computação, PUC-Rio, 2018
Mestre em Computação, PUC-Rio, 2013



Pós-graduado em Engenharia de Software, Universidade Senac Rio, 2009



Bacharel em Informática, UNESA Nova Friburgo, 2003



WAIS Tecnologia da Informação, 2002-2010



Thiago Delgado Pinto

PESQUISA RÁPIDA

Conhece / Usa no dia-a-dia

- Casos de Uso ?
- História de Usuário ?
- Teste unitário automatizado ?
- Teste de interface automatizado ?

introdução

software codifica conhecimento



código não é uma
boa mídia



transmitir conhecimento codificado?



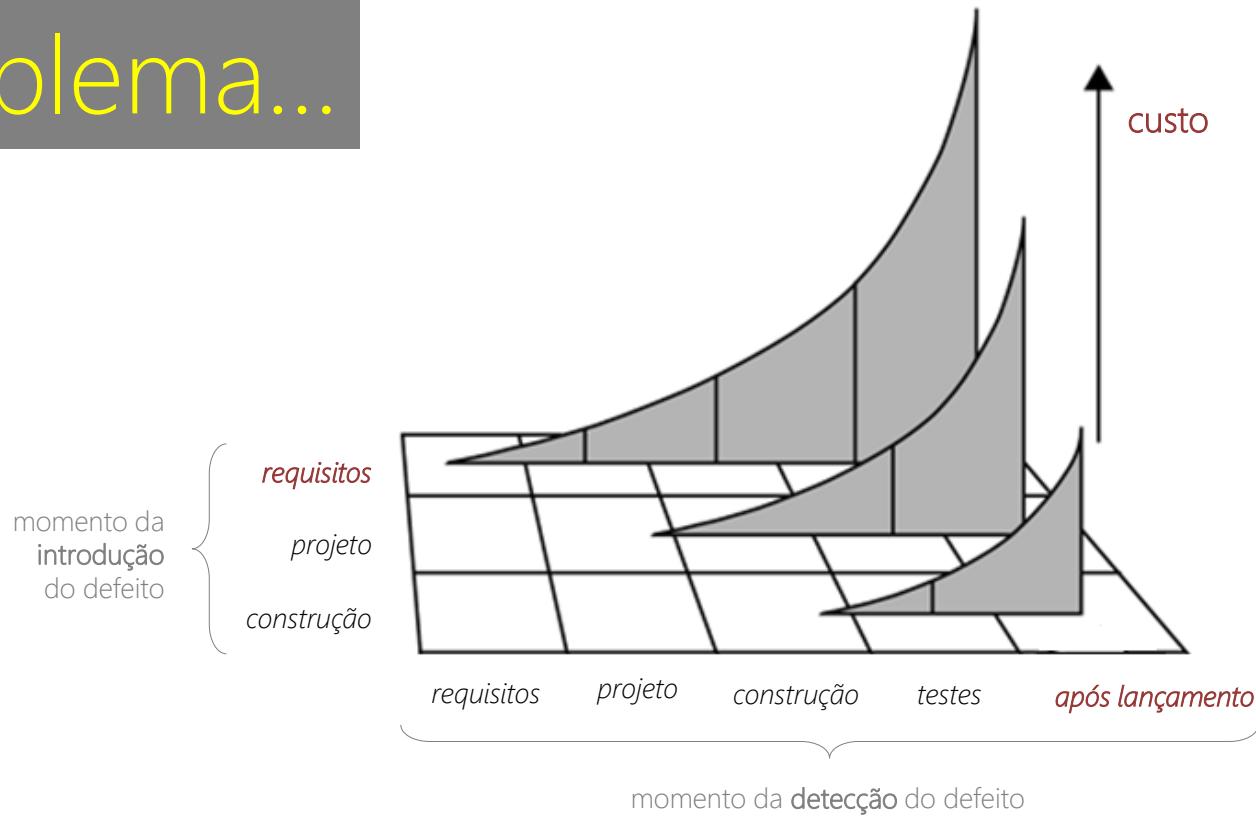


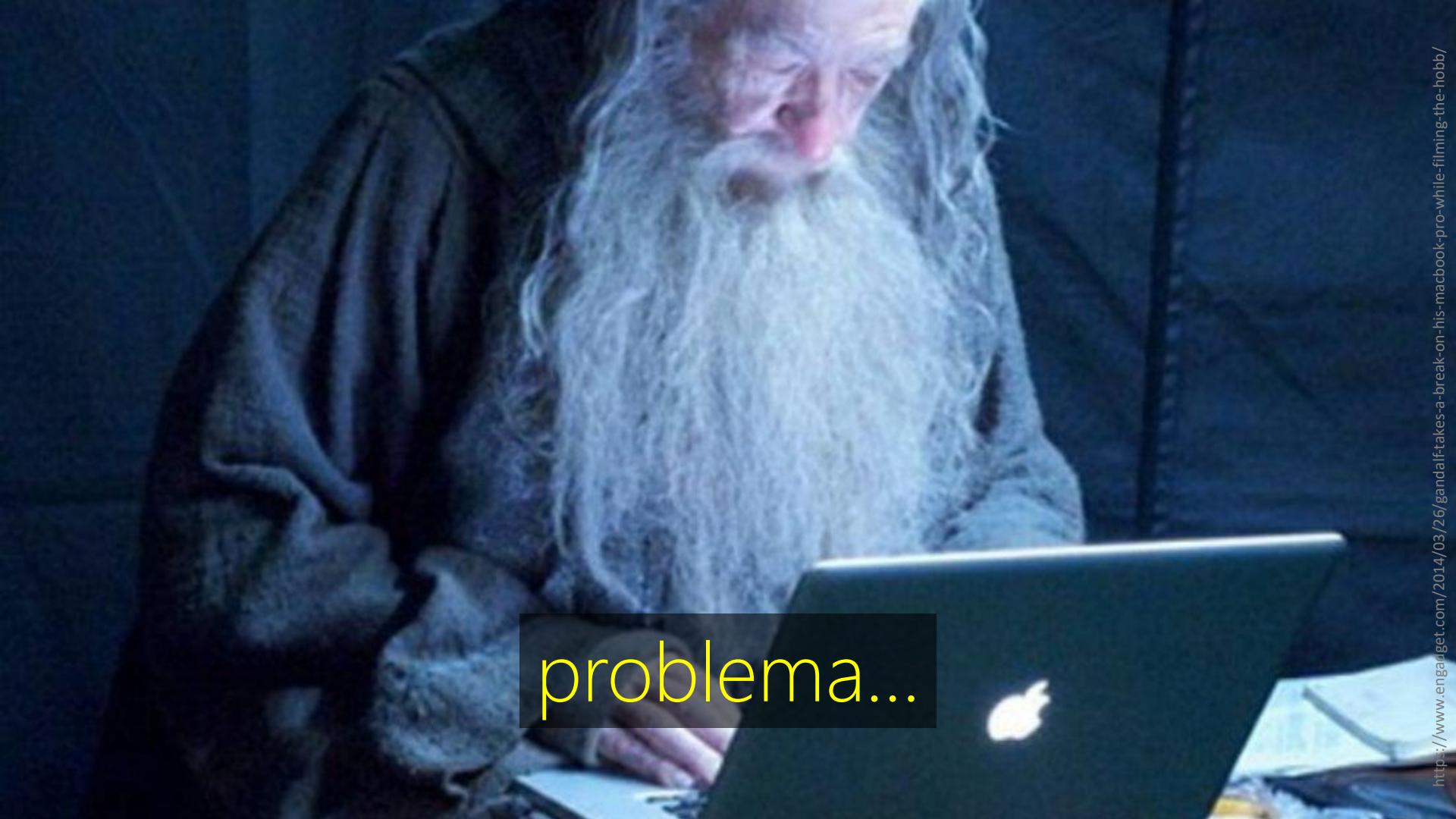
risco...



<http://www.energyrating.gov.au/lighting/types-of-light-bulbs>
<https://www.istockphoto.com/br/fotos/funny-keyboard-face?sort=mostpopular&mediatype=photography&phrase=funny%20keyboard%20faces>

problema...

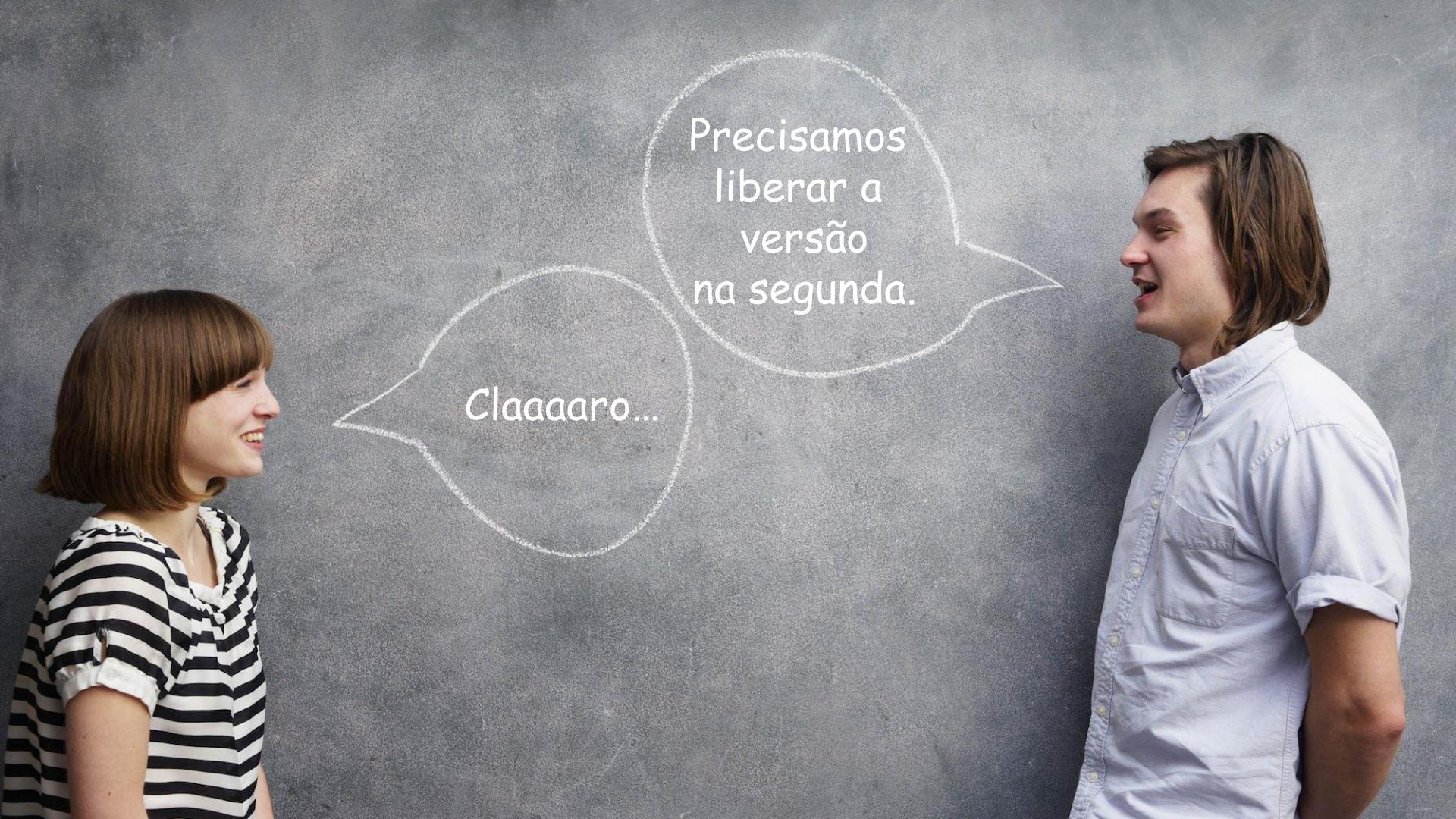


A photograph of Ian McKellen as Gandalf the Grey from The Hobbit movies. He is sitting at a table, looking down at a dark laptop computer. He has his signature long, white, bushy beard and is wearing a dark, textured robe. A black rectangular box with the word "problema..." written in yellow is overlaid on the lower-left portion of the image.

problema...

A woman with short brown hair and bangs, wearing a black and white horizontally striped shirt, looks upwards and to the right with a thoughtful expression. She is positioned in front of a dark grey chalkboard. A white chalk thought bubble originates from her head and extends towards the upper right corner of the board. Inside the thought bubble, the text "Ok, preciso ter ALGUMA especificação..." is written in a white, sans-serif font.

Ok, preciso ter
ALGUMA
especificação...



Precisamos
liberar a
versão
na segunda.

Claaaaaro...



A close-up photograph of a person's hand holding a silver-colored analog stopwatch. The stopwatch has a white face with black markings for hours (10, 11, 12, 1), minutes (5, 4, 3, 2), and seconds. A red second-hand needle is positioned between the 7 and 8 minute marks. The stopwatch is held over a clear, deflated balloon. On the balloon, the word "TIME" is printed in large, bold, black capital letters. The background is blurred, suggesting motion or a fast-paced environment.

E O MESMO
PROBLEMA
OCORRE PARA
TESTAR



Diagramas?

Outro?

Wiki?

Histórias de
Usuário?

Casos de Uso?

E aí ?

algumas
opções

Diagrams

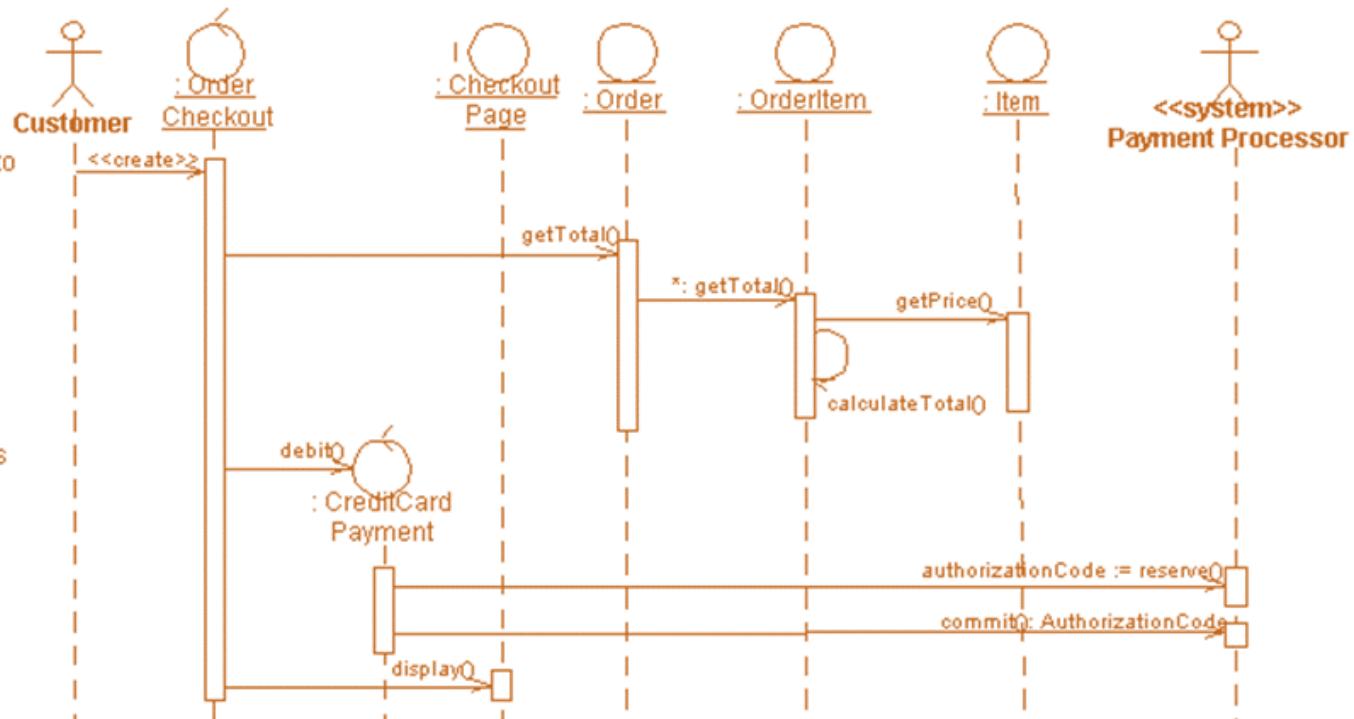
Checkout an online order.

1. The customer decides to checkout.
...

5. The system calculates the order total.
...

12. The system processes the credit card payment.
...

14. The system displays the checkout summary page.



Diagrams



Notação abstrata



Facilitam comunicação



Ferramenta é opcional



Permitem gerar testes
unitários simples



Notação especializada



Ferramenta requer treinamento



Manutenção cara



Vários diagramas p/ capturar o “todo”



Alterações simultâneas

Wiki-like

markdown.md

```
1 # Markdown Preview
2
3 Type on the _left_, see it
4   rendered on the _right_.
5
6 This is a
7   • [link](https://github.com).
8
9   - and this is
10    - a list
11
10 :tada: :fireworks:
```

markdown.md 10:19

markdown.md Preview

Markdown Preview

Type on the **left**, see it *rendered* on the **right**.

This is a [link](#).

- and this is
- a list



LF UTF-8 GitHub Markdown ⌂ master +11

Wiki-like



Notação fácil de aprender



Requer organização da equipe



Gera hipertexto (HTML)



Não indica como documentar
requisitos



Integração com
ferramentas de
desenvolvimento



Não gera testes

Casos de Uso

(descrição textual)

Delete Comment
Brief description: The actor deletes a comment
Actors: Group Leader
Preconditions: The actor is a registered student
Basic flow of events: <ol style="list-style-type: none">1. The actor logs into the system2. The system authenticates the actor and starts a session3. The actor chooses to delete a comment4. The actor is guided by the system to fill in required information to delete a comment5. The system acknowledges that comment deleted6. The actor leaves the system
Extensions: <ol style="list-style-type: none">1a. The system fails to authenticate the actor.<ul style="list-style-type: none">• The system informs the actor and doesn't allow the actor to proceed3a. The system fails to delete comment<ul style="list-style-type: none">• The system informs the actor
Post-conditions: The comment is deleted
Special requirements: Actor is connected to Internet and uses a Java enabled Mobile phone

Casos de Uso

(descrição textual)



Notação abstrata



Treinamento especializado



Facilitam comunicação



Notação formal



Podem documentar vários aspectos da aplicação



Alterações simultâneas



Permitem gerar testes funcionais



- funtester.org ou github.com/funtester
- gera testes funcionais completos para aplicações Web, Mobile e Java Swing
- opensource – escrita em Java
- desenvolvi durante meu mestrado

FunTester - PetShop [C:\dev\workspace\funtester\funtester\funtester-app\examples\petshop.fun]*

File Edit Software Tests Help

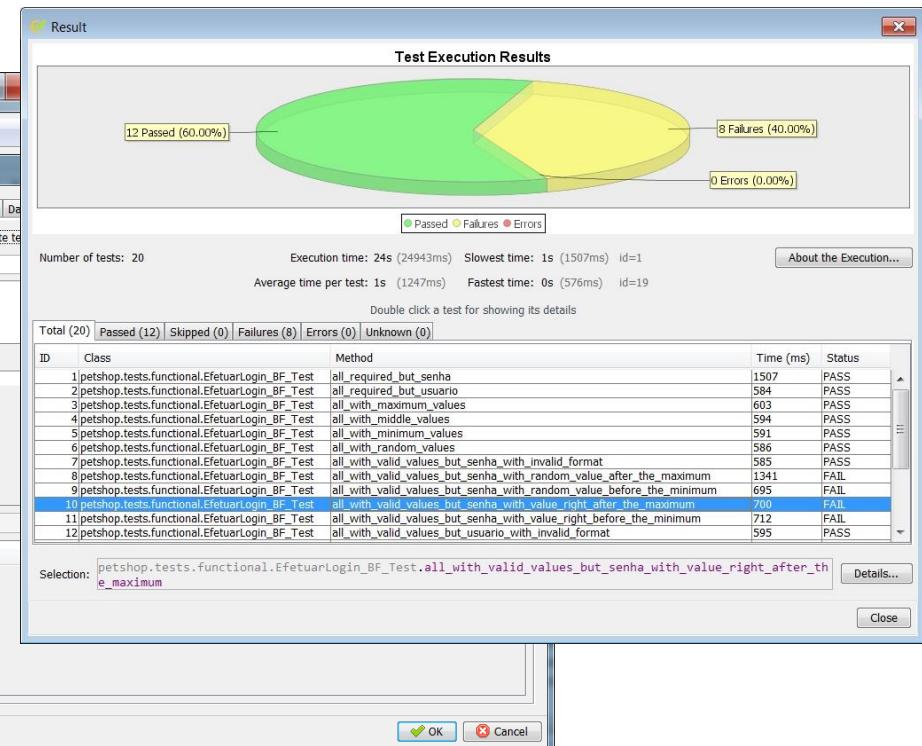
Use Cases Actors Databases & Queries Regular Expressions

New Edit Remove New

PetShop
petshop
logins
password for a user
petshopstest

Name: petshop
Driver: com.mysql.jdbc.Driver
Dialect:
Type: mysql
Host: 127.0.0.1
Port: 3306
Path: petshop
User: root
Password:
JDBC URL: jdbc:mysql://127.0.0.1:3306/petshop

Test Connection



Histórias de Usuário

Funcionalidade: Login

Como um usuário
Eu desejo me autenticar
Para acessar a aplicação

Cenário: Login com sucesso

Dado que eu vejo a tela de login
Quando eu informo minhas credenciais
Então eu consigo acessar a tela principal da aplicação

Histórias de Usuário



Notação fácil de aprender



Facilitam comunicação



Integração com
ferramentas de
desenvolvimento



Equipe precisa padronizar nível
de abstração

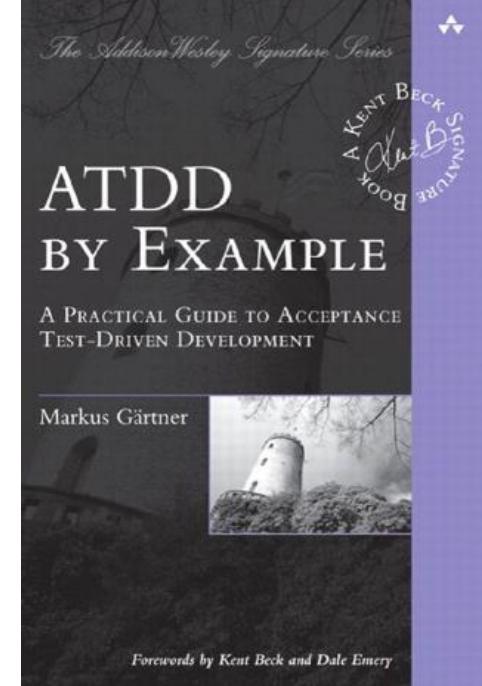
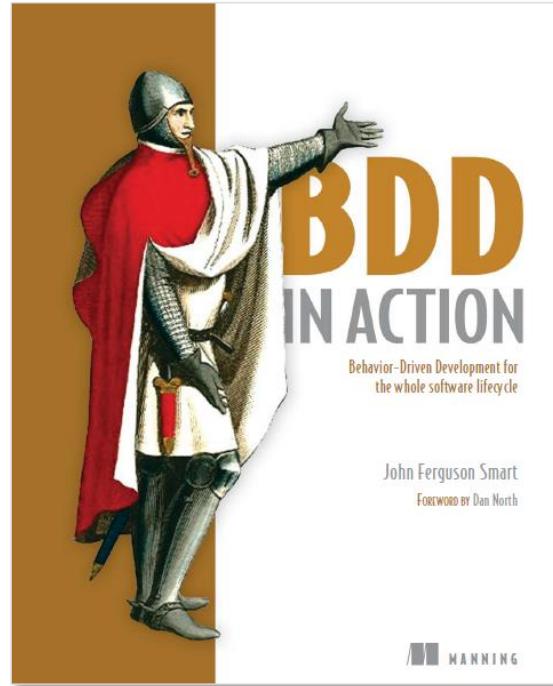
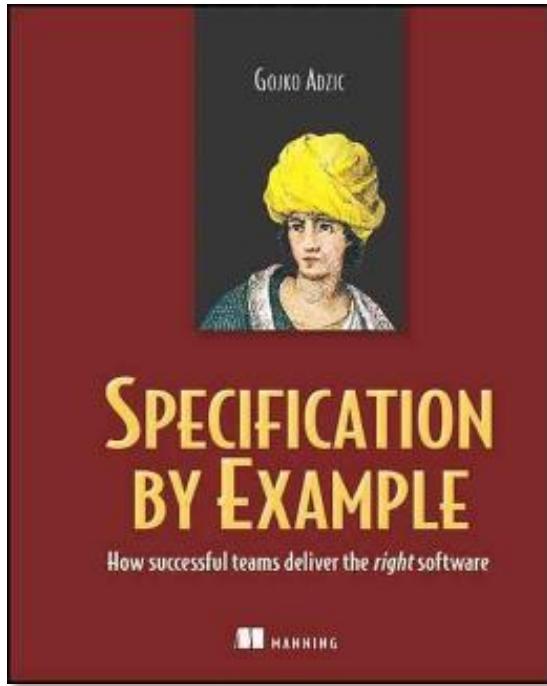


Rigor técnico depende de
esforço da equipe

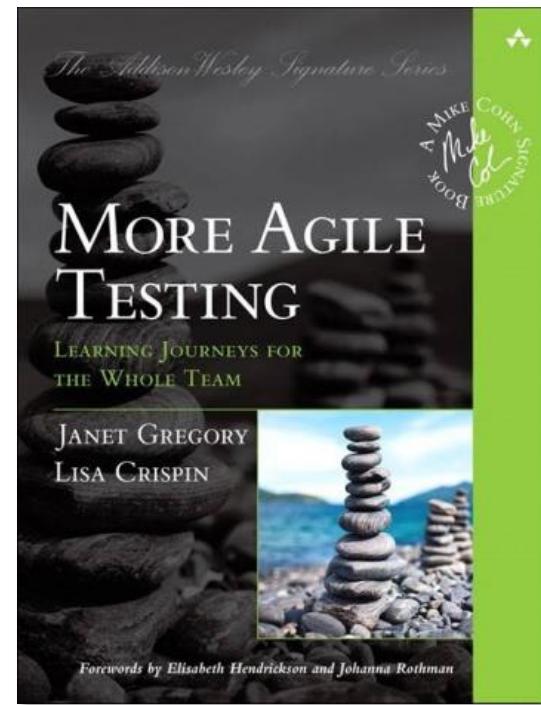
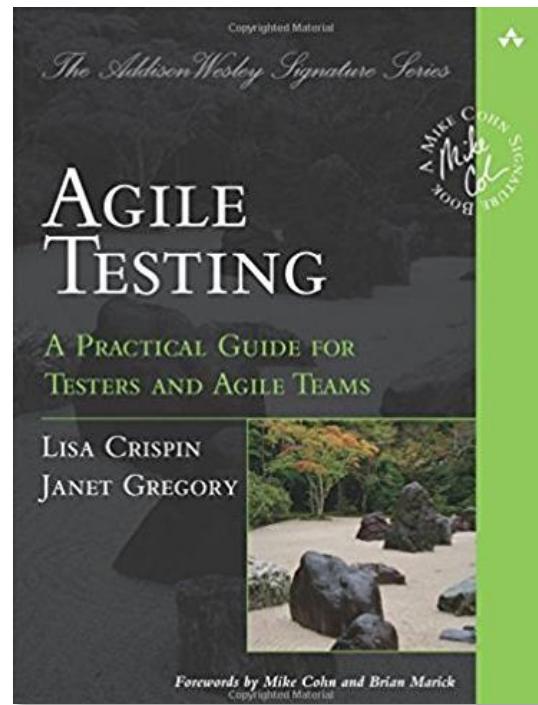
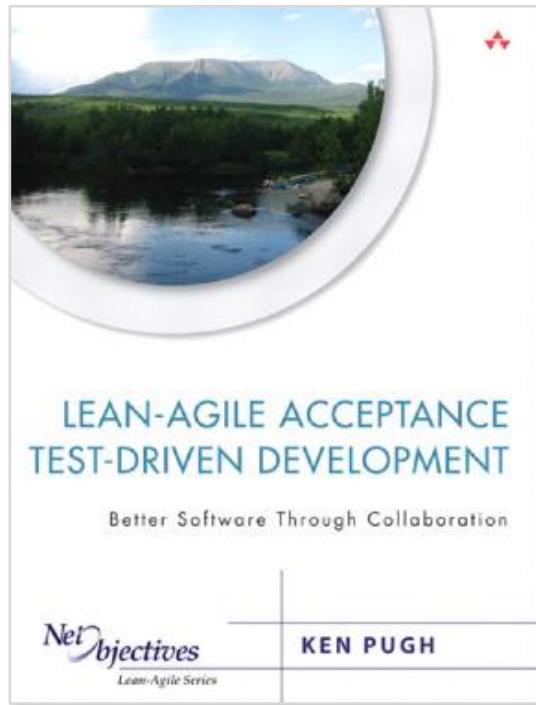


Não permite gerar testes

processos ágeis modernos



... e teste ágil

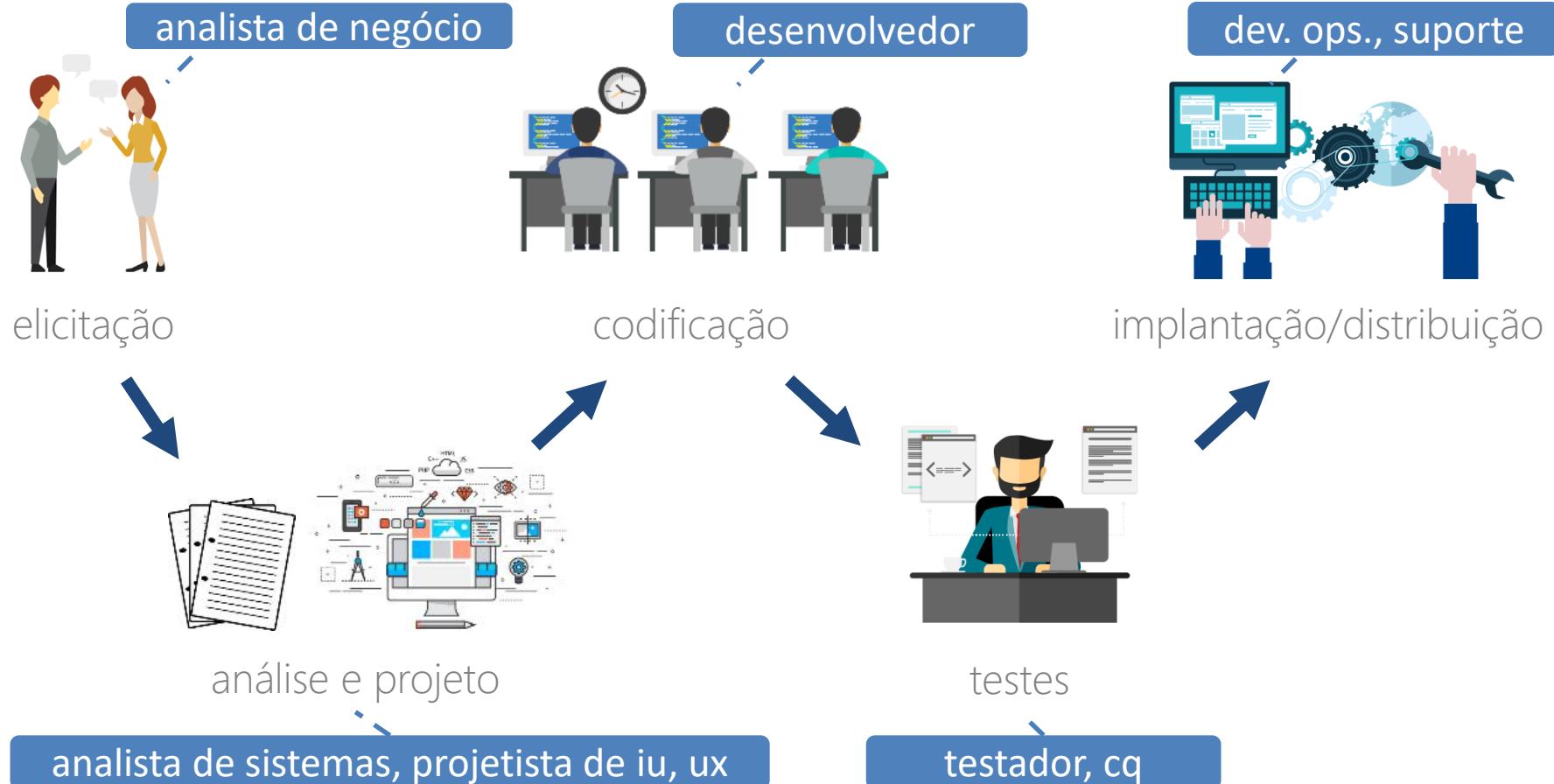


ideias do SbE/BDD/ATDD

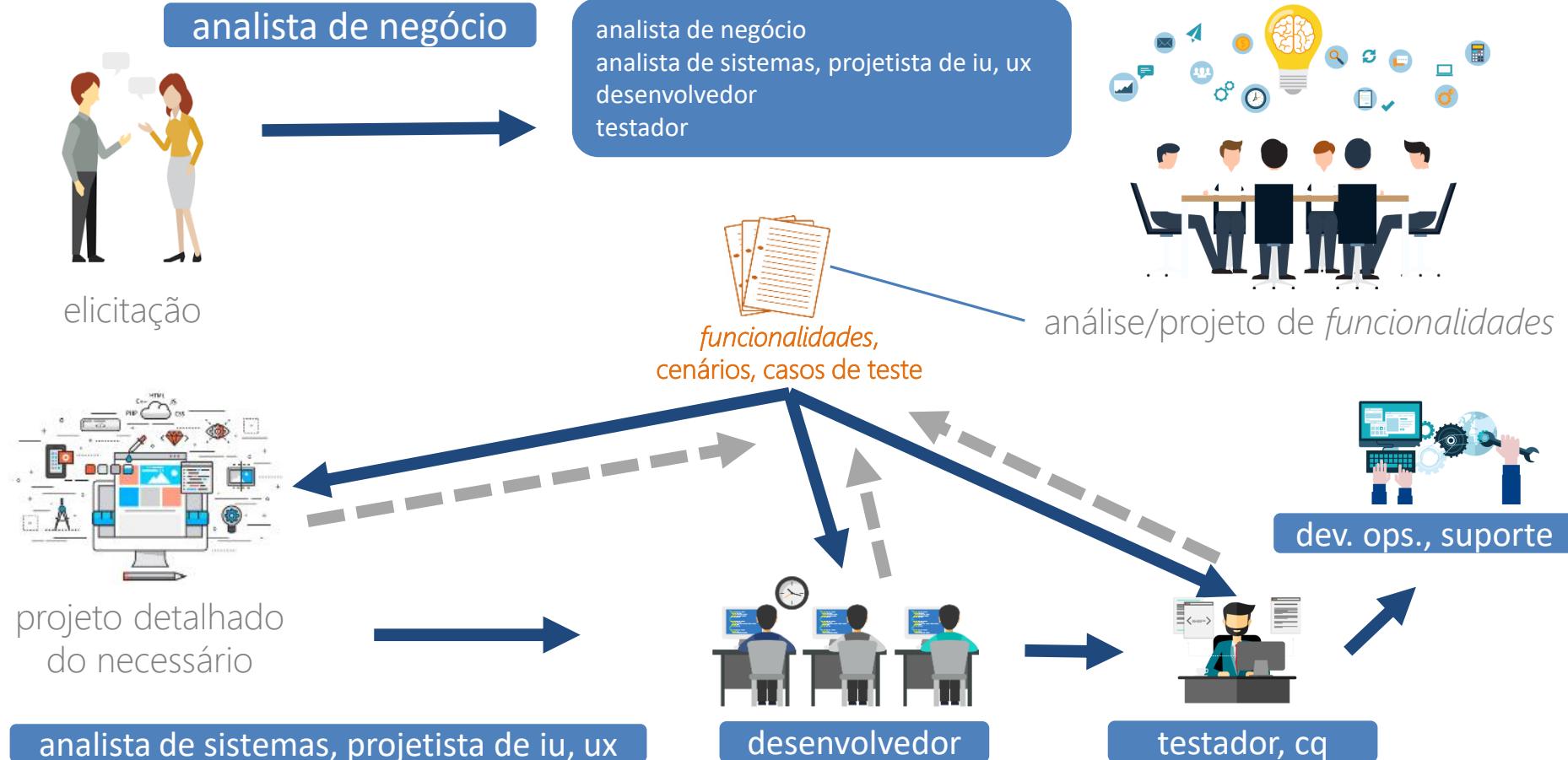
1. testes na forma de requisitos, requisitos na forma de testes
2. *workshops* (oficinas) para esclarecer requisitos
3. prevenção de defeitos, em vez de detecção
4. engenharia concorrente
5. validação na entrega

dinâmica

processo tradicional



processos ágeis modernos

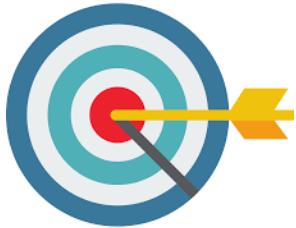


cooperação

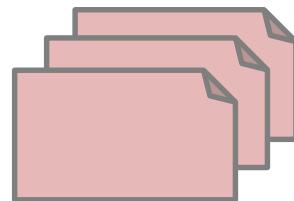
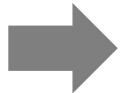


*analista de negócio
analista de sistemas, projetista de ui, ux
desenvolvedor
testador*

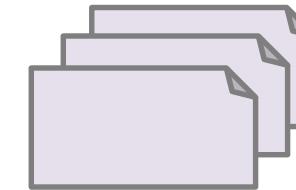
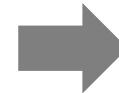
especificação



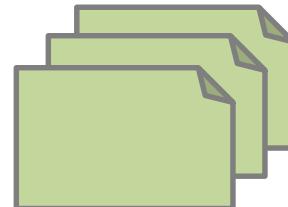
meta de negócio



funcionalidades



exemplos



teste de aceitação
automatizado

exemplo

"Aumentar a venda de passagens aéreas do próximo ano em 5%, estimulando viajantes a voarem com a Kaikai, ao invés de uma rival"



funcionalidades derivadas:

- Ganhar milhas ao voar
- Ganhar milhas ao fazer compras
- Ver milhas acumuladas online



especificando uma funcionalidade

Funcionalidade: Ganhar milhas ao voar

Como um viajante

Desejo poder acumular milhas toda vez que voar

Para poder viajar de graça ou ter descontos



História de Usuário
User Story

modelo

Funcionalidade: <título>

Como um <papel desempenhado>

Desejo <meta>

Para <benefício obtido>



opcional → descreva só quando agregar valor

derivando exemplos (cenários)

Ganhar milhas ao voar

Um novo viajante começa
com nível Bronze

Viajante passa para nível
Prata ao atingir 300 milhas

exemplos permitem

explorar requisitos

descobrir o que não sabemos

esclarecer ambiguidades

identificar mal-entendidos

compreender detalhes do que ocorre na prática

Ao solicitar detalhes do vôo PG-103, eu devo ver que é um vôo Porto Alegre/POA – Rio de Janeiro/Galeão, das 09:35.

escrevendo exemplos

expressamos exemplos de forma **estruturada**
padroniza a forma de escrever

com um modelo, chamado de *Given-When-Then*, ou GWT
em português, *Dado que – Quando - Então*, ou DQE

Cenário: Um novo viajante começa com nível Bronze
Dado que não sou um cliente da Kaikai
Quando eu me cadastro no programa de milhas
Então eu recebo o nível Bronze

escrevendo exemplos

Cenário: Viajante visualiza histórico de milhas

Dado que estou logado no site

Quando eu solicito meu histórico de milhas

Então eu vejo uma listagem como a seguir:

Data e hora	Origem	Vôo	Milhas
23/12/2017 19:43	Compra de passagem	PG-103	10
20/12/2017 13:15	Compra de passagem	GP-094	10
Total:			20

modelo

Cenário: <título>

Dado que <contexto>

Quando <evento ocorre>

Então <resultado deve ocorrer>

essa é uma Linguagem de Domínio Específico (DSL), assim como a *User Story*, mas voltada para descrever **cenários** de uma especificação

modelo – uso do "e"

Cenário: <título>

Dado que <contexto 1>

e que <contexto 2>

e que ...

Quando <evento 1 ocorre>

e <evento 2 ocorre>

e ...

Então <resultado 1 deve ocorrer>

e <resultado 2 deve ocorrer>

Cenário: Cadastro com dados básicos

Dado que não possuo cadastro

Quando aciono a opção para me cadastrar
e forneço nome, e-mail e senha

e aciono a confirmação de cadastro

Então sou direcionado para a área de login
e recebo um e-mail confirmando minha conta

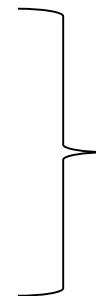
DSLs

Funcionalidade: <título>

Como um <papel desempenhado>

Desejo <meta>

Para <benefício obtido>



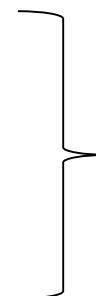
História de Usuário (opcional)

Cenário: <título>

Dado que <contexto>

Quando <evento ocorre>

Então <resultado deve ocorrer>



Descrição de Cenário

utilidade vs. nível de abstração

Cenário: Cadastro com dados básicos

Dado que não possuo cadastro

Quando me cadastro

Então passo a possuir cadastro

abstrato demais, não agrega valor

Cenário: Cadastro com dados básicos

Dado que não possuo cadastro

Quando aciono a opção para me cadastrar

e forneço nome, e-mail e senha

e aciono a confirmação de cadastro

Então sou direcionado para a área de login

e recebo um e-mail confirmando minha conta

equilibrado

Cenário: Cadastro com dados básicos

Dado que a tabela "usuários" não possui
um registro com "bob@email.com"

e estou em "/home"

Quando clico na div com id "cadastrar"

e digito nome, e-mail e senha

e clico no botão Salvar

Então sou direcionado "/login"

e recebo um e-mail confirmando minha conta

detalhes demais, podem servir para testador, mas
agregam pouco ao entendimento do negócio

utilidade vs. escopo

Funcionalidade: Cadastro de Cidade

Cenário: Cadastro com sucesso

Dado que estou na tela de cadastro de cidade
Quando forneço nome e DDD
e aciono a confirmação de cadastro
Então tenho a cidade cadastrada

Funcionalidade: Alteração de Cidade

Cenário: Alteração com sucesso

Dado que estou na tela de cadastro de cidade
Quando forneço nome e DDD
e aciono a confirmação de cadastro
Então tenho a cidade alterada

Funcionalidade: Cidade

Cenário: Cadastro com sucesso

Dado que estou na tela de cadastro de cidade
Quando forneço nome e DDD
e aciono a confirmação de cadastro
Então tenho a cidade cadastrada

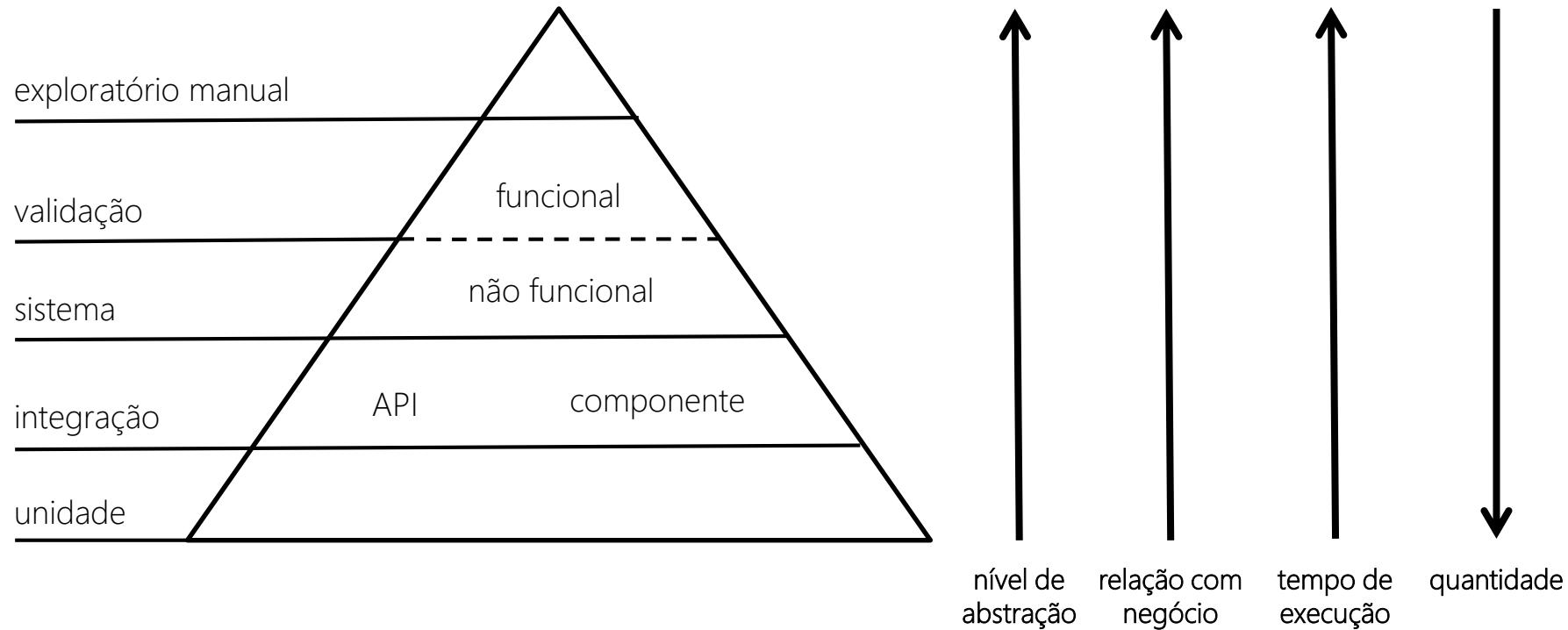
Cenário: Alteração com sucesso

Dado que estou na tela de cadastro de cidade
Quando forneço nome e DDD
e aciono a confirmação de cadastro
Então tenho a cidade alterada

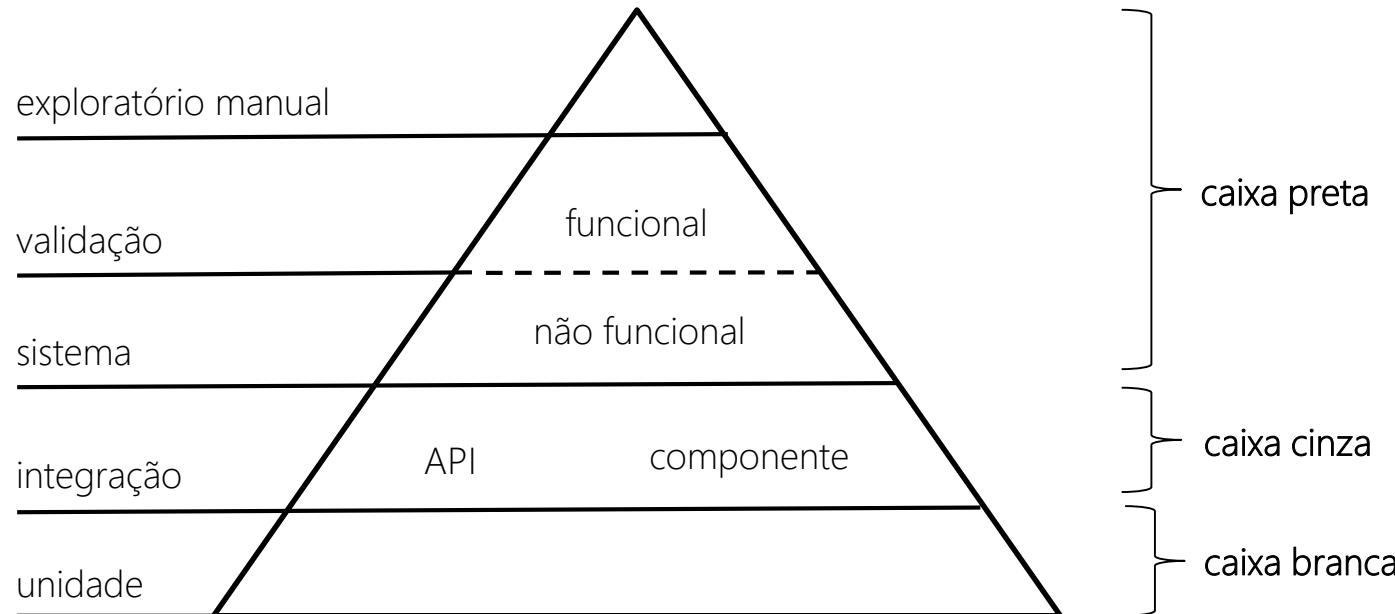
Ambos teoricamente corretos. Porém, versão com uma *feature* por arquivo é mais comum na comunidade ágil.

tipos
de teste

pirâmide



níveis de abstração



índices de detecção de defeitos

<i>atividade</i>	<i>mais baixo</i>	<i>modal</i>	<i>mais alto</i>
revisão informal de projeto	25%	35%	40%
inspeção formal de projeto	45%	55%	65%
revisão informal de código	20%	25%	35%
inspeção formal de código	45%	60%	70%
programação em pares	40%	50%	60%
modelagem ou prototipagem	35%	65%	80%
verificação pessoal em cópia impressa do código	20%	40%	60%
teste unitário	15%	30%	50%
teste de integração	25%	35%	40%
teste de regressão	15%	25%	30%
teste de sistema	25%	40%	55%



(JONES, 1986; JONES, 1996; SHULL *et al.*, 2002; MCCONNELL, 2004)

algumas
ferramentas e
frameworks JS

teste unitário e de integração

fornecem **estrutura de testes**

[Mocha](#), [Jasmine](#), [Jest](#), [Cucumber](#), [CodeceptJS](#)

fornecem **assertivas** (para oráculos)

[Chai](#), [Jasmine](#), [Jest](#), [Unexpected](#)

fornecem **exibição de resultados e monitoramento** (watch)

[Mocha](#), [Jasmine](#), [Jest](#), [Karma](#)

fornecem **snapshots** (serializa e compara na próxima execução)

[Jest](#), [Ava](#)

fornecem **mocks, spies e stubs** (imitam comportamento)

[Sinon](#), [Jasmine](#), [enzyme](#), [Jest](#), [testdouble](#)

fornecem **relatórios de cobertura de código**

[Istanbul](#), [Jest](#), [Blanket](#)

fornecem ou simulam um **navegador**

[Protractor](#), [Nightwatch](#), [Phantom](#), [Casper](#), [Cypress](#), [CodeceptJS](#)

[supertest](#), [apickli](#), [api-easy](#), [frisby](#), [chakram](#)

exemplo de SuperTest:

```
describe('GET para /usuario', function() {  
  it('responde com json', function(done) {  
    api.get('/usuario')  
      .expect( 'Content-Type', /json/ )  
      .expect( 200, done );  
  });  
});
```

desempenho

[benchmark.js](#)

exemplo:

```
var suite = new Benchmark.Suite;
var raiz1 = require( 'raiz' ).raiz1, raiz2 = require( 'raiz' ).raiz2;

suite
  .add('raiz quadrada com algoritmo 1', function() {
    raiz1( 1000000 );
  })
  .add('raiz quadrada com algoritmo 2', function() {
    raiz2( 1000000 );
  })
  .on('complete', function() {
    console.log( Mais rápido é ' + this.filter('fastest').map('name'));
  })
  .run({ 'async': true });
```

carregamento

Artillery

exemplo:

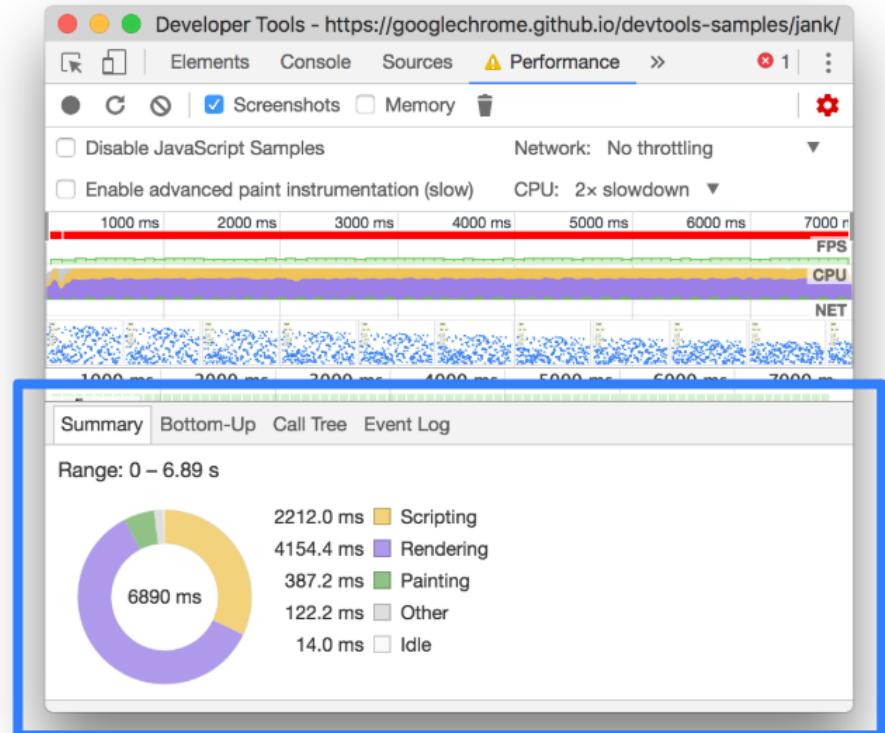
```
artillery quick --count 10 -n 20 https://artillery.io/
```

irá criar 10 "usuários virtuais", cada um disparando 20 requisições HTTP GET para o endereço informado

desempenho detalhado de aplicações web

[Google Chrome](#) (Dev Tools)

[Mozilla Firefox Developers Edition](#)



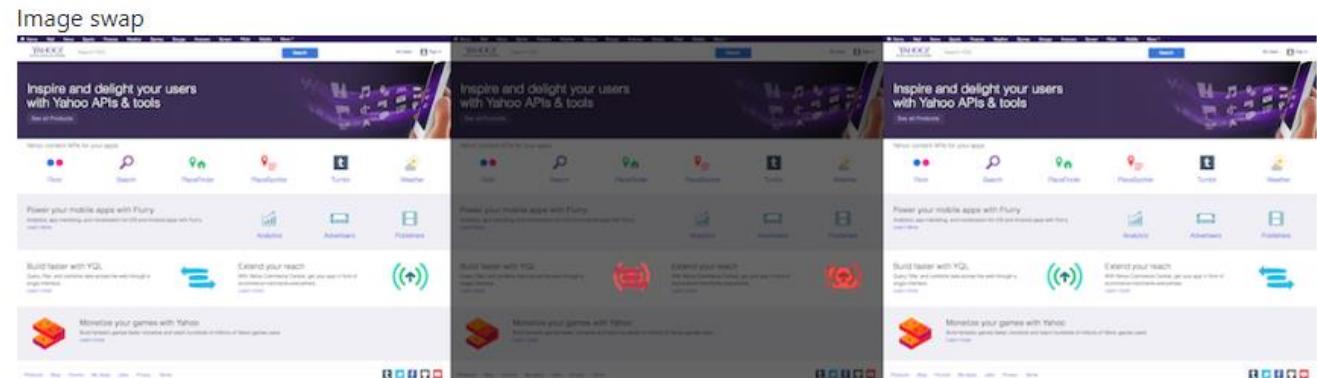
testes não funcionais

teste visual (comparação de conteúdos)

Kobold

Appraise

BlinkDiff



Exemplo do Kobold:

```
$ kobold test/ui/regression
```

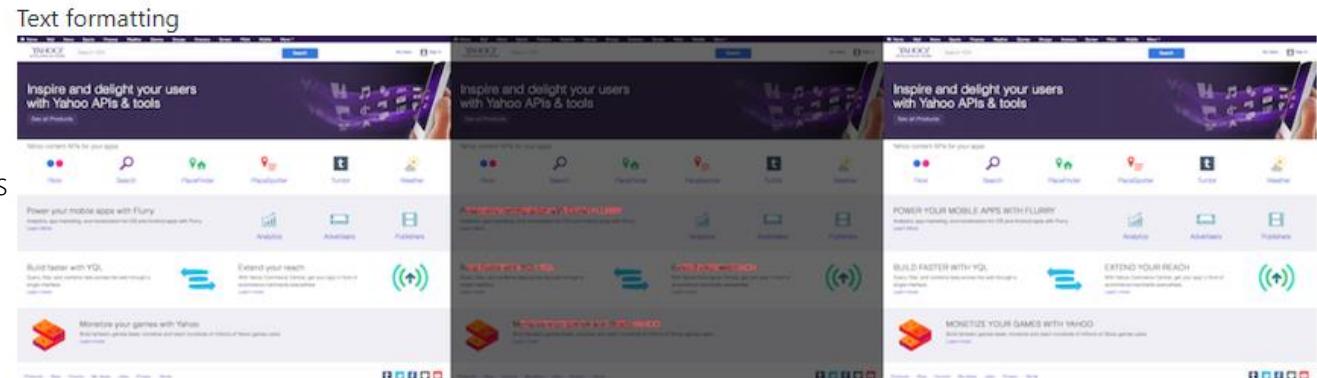
Irá comparar imagens em subdiretórios

/approved

/build

e irá gerar em caso de diferenças

/highlight



testes funcionais

1/3

[Nightwatch](#), [Puppeteer](#), [WebDriver.io](#), [Casper](#), [TestCafe](#), [Cypress](#), [CucumberJS](#), [Tartare](#), [CodeceptJS](#)

Recurso	Casper	Nightwatch	WebDriver.io	Protractor	CodeceptJS
Screenshots	Sim	Sim	Sim	Sim	Sim
Profiling de memória e desempenho	Não	Sim (Chrome Driver)	Sim (Chrome Driver)	Sim (Chrome Driver)	Sim (Chrome Driver)
Análise de Cobertura	Sim	Sim	Sim	Sim	Sim
Suporta PageObject	Não	Não	Sim	Sim	Sim
Sup. testes Síncronos	Não	Não	Sim	Sim	Sim
Relatórios	CMD, xUnit	HTML, Allure, xUnit	HTML, Allure, xUnit, Perfecto	HTML, xUnit, Allure	HTML, CLI, xUnit
Nuvem e automação	Não	web e mobile	web e mobile	web e mobile	web e mobile

exemplo WebDriver.IO

```
var webdriverio = require('webdriverio');
var options = { desiredCapabilities: { browserName: 'chrome' } };
var client = webdriverio.remote(options);
client
  .init()
  .url('https://localhost/app/')
  .setValue('#login', 'admin')
  .setValue('#senha', '123456')
  .click('#entrar')
  .getText().then(function(text) {
    assert.equal( text, 'Olá' );
  })
  .end();
```

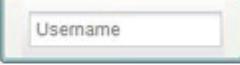
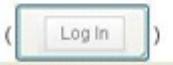
exemplo CodeceptJS

```
Feature('Login');

Scenario('Login de administrador com sucesso', (I) => {
    I.amOnPage('http://localhost/app');
    I.fillField('#login', 'admin');
    I.fillField('#senha', '123456');
    I.click('#entrar');
    I.see('Olá');
});
```

outras ferramentas

Sikuli (automação de tarefas)

```
31 #Logearse como usuario registrado
32 if exists(  ) :
33     print("Usuario ya logeado")
34 else:
35     type( , "javipello")
36     wait(5)
37     type( , "*****")
38     wait(5)
39     click(  )
40     print("Usuario se acaba de logear")
41 wait(8)
42
```

Exemplo de Sikuli em uma aplicação desktop (retirado do site)

outras ferramentas

Selenium IDE

ferramenta de gravação e reprodução

Katalon Automation Recorder

ferramenta de gravação e reprodução

The screenshot shows the Selenium IDE interface. At the top, there's a toolbar with icons for file operations and a search bar. Below that is a navigation bar with tabs: 'Tests' (selected), 'Selenium IDE*', and 'Log'. The main area has a table titled 'Selenium IDE*' showing a sequence of test steps:

	Command	Target	Value
1.	open	/	
2.	click at	css=a[title="Selenium Projects"]	52,15
3.	click at	link=Selenium IDE	67,7
4.	assert text	css=td > p	Selenium IDE is a n integrated dev elopment envir...

Below the table, there are input fields for 'Command' (set to 'assert text'), 'Target' (set to 'css=td > p'), 'Value' (set to 'Selenium IDE is an integrated development environment'), and a 'Comment' field. At the bottom left, it says 'Runs: 1 Failures: 0'. The 'Log' tab is open at the bottom, showing the execution results:

Running 'Selenium IDE'
1. Trying to execute open on /... Success
2. Trying to execute clickAt on css=a[title="Selenium Projects"] with value 52,15... Success
3. Trying to execute clickAt on link=Selenium IDE with value 67,7... Success
4. Trying to execute assertText on css=td > p with value Selenium IDE is an integrated development environment for Selenium scripts. It is implemented as a Firefox extension, and allows you to record, edit, and debug tests. Selenium IDE includes the entire Selenium Core, allowing you to easily and quickly record and play back tests in the actual environment that they will run in.... Success
'Selenium IDE' completed successfully

Exemplo do Selenium IDE (retirado do site)

gravação e reprodução - observações

ferramentas são práticas

porém, podem ter problemas de manutenção

- gravação de ações indesejadas

- mudança de interface (ex.: identificadores) pode requerer regravação
 - problemático para funcionalidades grandes

- não permite modularização ou é preciso fazê-la manualmente

efetividade para teste pode ser baixa

- só repete o que foi gravado

- não testa caminhos novos ou dados novos

katalon-concordia

<https://github.com/thiagodp/katalon-concordia>



Chrome e Firefox

transforma testes gravados com **Katalon Recorder** em **Concordia**

bastante útil para capturar identificação de elementos da tela

katalon-concordia

The screenshot shows the Katalon Automation Recorder interface version 3.5.6. The main window displays a test suite structure on the left and a command table on the right.

Test Suites:

- Test Suite 1
 - Test Case 1
- Untitled Test Suite
 - Untitled Test Case

Command Table:

Command	Target	Value
open	https://www.google.com/	
click	id=lst-ib	
type	id=lst-ib	sasa
click	id=rhscol	
click	id=lst-ib	
sendKeys	id=lst-ib	\$(KEY_ENTER)

Bottom Navigation:

- Log (selected)
- Screenshots
- Variables
- Data Driven
- Extension Scripts
- Reference

Analytics:

- Analytics icon
- Log icon
- Clear icon
- More options icon

OK, MAS COMO FAÇO PARA
CRIAR BONS TESTES ?

QUE REVELEM DEFEITOS



testes funcionais técnicas básicas

aleatório

teste aleatório

seleciona entradas aleatórias, de acordo com o domínio

exemplo: no domínio entre 1 e 100, escolher valores como 17, 63 e 82

entradas podem diferir daquelas usadas por testadores

exercitando novos caminhos
e com isso revelar defeitos

uso recomendado junto a outras práticas

adivinhação

adivinhação

geralmente baseada em **experiência prévia** do testador com
domínio

ou quando há alguma **indicação** de **problemas** em potencial

exemplos

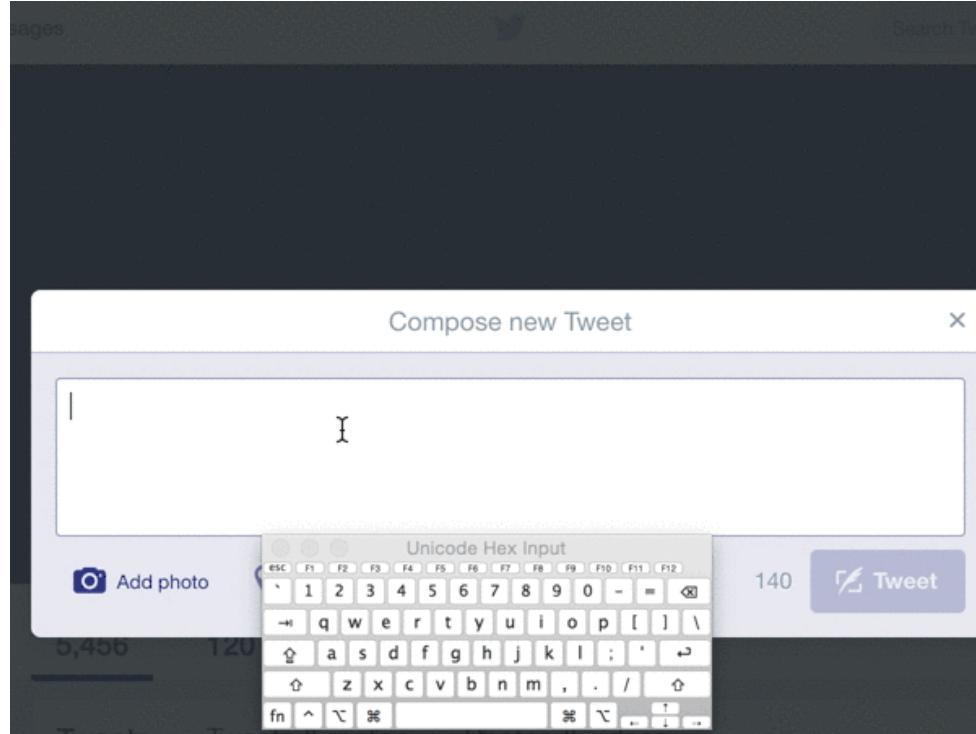
divisão por zero

caracteres problemáticos

extrapolação de limites de string ou array

se possível, deve ser documentada para uso futuro

falando nisso...



veja <https://github.com/minimaxir/big-list-of-naughty-strings>

particionamento
em classes de
equivalência

cria **partições** no domínio de entrada ou de saída
as chamadas *classes de equivalência*

a **equivalência** se refere ao fato de a técnica assumir que todos os membros de certa classe são **processados** pelo sistema da **mesma forma**
sendo, portanto, equivalentes para o sistema

testar um **membro** da classe **equivale** a testar **qualquer outro** dela

exemplo: "usuário deve ter 18 anos ou mais"



se um **membro** da classe é capaz de **revelar** certo **defeito**,
qualquer outro dela é capaz de relevan o mesmo defeito

idem se não for capaz de revelar

a não ser que haja subclasses
discussão em Beizer (1990)*

*Beizer, B. **Software Testing Techniques**, second edition, Van Nostrand Reinhold, New York, 1990.

pce – vantagens

elimina a necessidade de **teste exaustivo**, que não é viável

guia o testador a **selecionar** um **subconjunto** de entradas de teste com **alta probabilidade** de encontrar um **defeito**

permite ao testador **cobrir** um **largo domínio** de entrada/saída com um **pequeno subconjunto** da classe de equivalência escolhida

pce – escolha das classes

Myers (2004) propõe escolher condições de entrada "interessantes"

vêm de uma descrição na **especificação do software** a ser testado

testador e analista interagem para **desenvolver**...

- conjunto **verificável** de requisitos
- especificação (completa e correta) de **entradas e saídas**

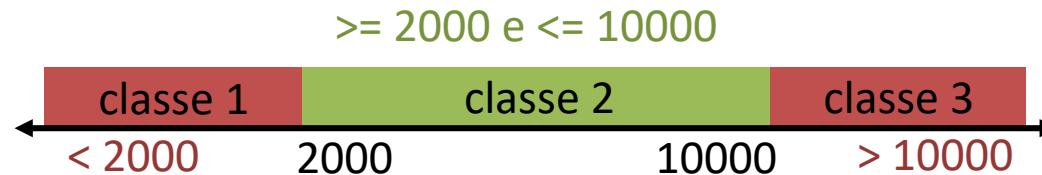
testador identifica **condições** e cria **classes**

como no exemplo anterior → "usuário deve ter 18 anos ou mais" → < 18 e ≥ 18

daí desenvolve **Casos de Teste** para **cobrir classes** identificadas

faixa de valores → uma classe para válidos e duas para inválidos

exemplo: "salário deve estar entre 2.000 e 10.000 reais"



classe 1 – inválidos: qualquer < 2000

classe 2 – válidos: qualquer $\geq 2000 \text{ e } \leq 10000$

classe 3 – inválidos: qualquer > 10000

conjunto de valores → uma classe para válidos e uma para inválidos

exemplo: "nível de acesso deve ser *Administrador*, *Gerente*, *Vendedor* ou *Estoquista*"

classe 1 - *válidos*: "*Administrador*", "*Gerente*", "*Vendedor*", "*Estoquista*"

classe 2 - *inválidos*: qualquer outro, como "*Visitante*", "*XjNq6*", ou ""

condição especial → uma classe para válidos e uma para inválidos

exemplo: "código de barras deve começar com 789"

classe 1 – **válido** – exemplos: "7891234567890", "789", ...

classe 2 – **inválidos** – exemplos: "6781234567890", "z78", "", ...

exemplo: "código de barras deve começar com 789 e ter 13 dígitos numéricos"

tem 2 condições – porém válidos deve respeitar ambas

classe 1 – **válido** - exemplos: "7891234567890", "7890000000000", ...

classe 2 – **inválidos com 13 dígitos numéricos** – exemplos: "6781234567890", "0000000000000", ...

classe 3 – **inválidos que começam com 789** – exemplos: "789", "789@zw\$", ...

classe 4 – **inválidos que não começam com 789 nem têm 13 dígitos numéricos** – exemplos: "", "@zw\$", ...

subcasos → quando certos valores funcionam de modo diferente
não seguem o mesmo padrão dos demais

exemplo: "CPF deve ter 11 dígitos numéricos e ser válido segundo o cálculo ..."
porém, há casos conhecidos que fogem a regra, como ter todos os números repetidos (ex.: "00000000000", "11111111111", ..., "99999999999")

subcaso deve ser tratado com nova(a) classe(s)

ex.:

classe 1 – válidos – exemplos: "35005215093", "95411615020", ...

classe 2 – inválidos que passam no cálculo – exemplos: "00000000000", ...

classe 3 – inválidos que não passam no cálculo – exemplos: "35005215095", ...

classe 4 – inválidos que não passam no formato – exemplos: "U4_%#", ...

análise de valor
limite

análise de valor limite

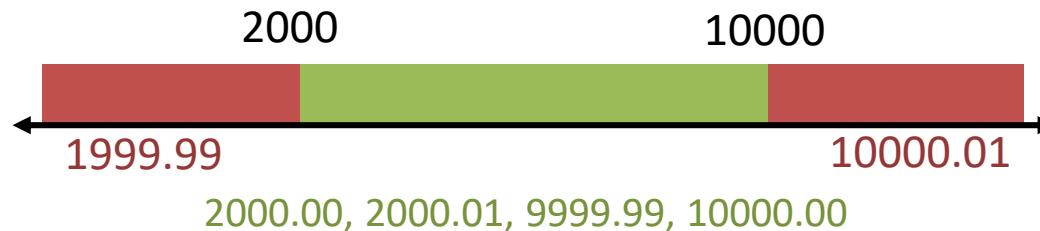
sugere que o testador selecione **valores próximos** aos limites
por muitos **defeitos** ocorrerem nos limites, ou **acima**, ou **abaixo**
complementa a técnica de particionamento em **classes de equiv.**
pode atuar no limite das classes

exemplo: "usuário deve ter 18 anos ou mais"
17 (abaixo), 18 (limite) e 19 (acima) podem ser incluídos no teste



faixa de valores → válidos para os limítrofes e inválidos para abaixo e acima

exemplo: "salário deve estar entre 2.000 e 10.000 reais"



válidos: 2000.00, 2000.01, 9999.99 e 10000.00

inválidos: 1999.99 e 10000.01

conjunto de valores → foco no primeiro e último

exemplo: "nível de acesso deve ser Administrador, Gerente, Vendedor ou Estoquista"

válidos – exemplos: "Administrador", "Estoquista"

inválidos – exemplos: "Visitante", "XjNq6", ou ""

combinando
técnicas

"salário deve estar entre 2.000 e 10.000 reais" → faixa de valores

pce + aleatório:

- caso 1 – pce classe 1 – inválido → < 2000.00 → aleatório → 653.11
- caso 2 – pce classe 2 – válido → >= 2000.00 e <= 10000.00 → aleatório → 7885.90
- caso 3 – pce classe 3 – inválido → > 10000.00 → aleatório → 224.020.15

avl:

- caso 1 – inválido – 1999.99
- caso 2 – válido – 2000.00
- caso 3 – válido – 2000.01
- caso 4 – válido – 9999.99
- caso 5 – válido – 10000.00
- caso 6 – inválido – 10000.01

pce + avl + aleatório – exemplo

2/2

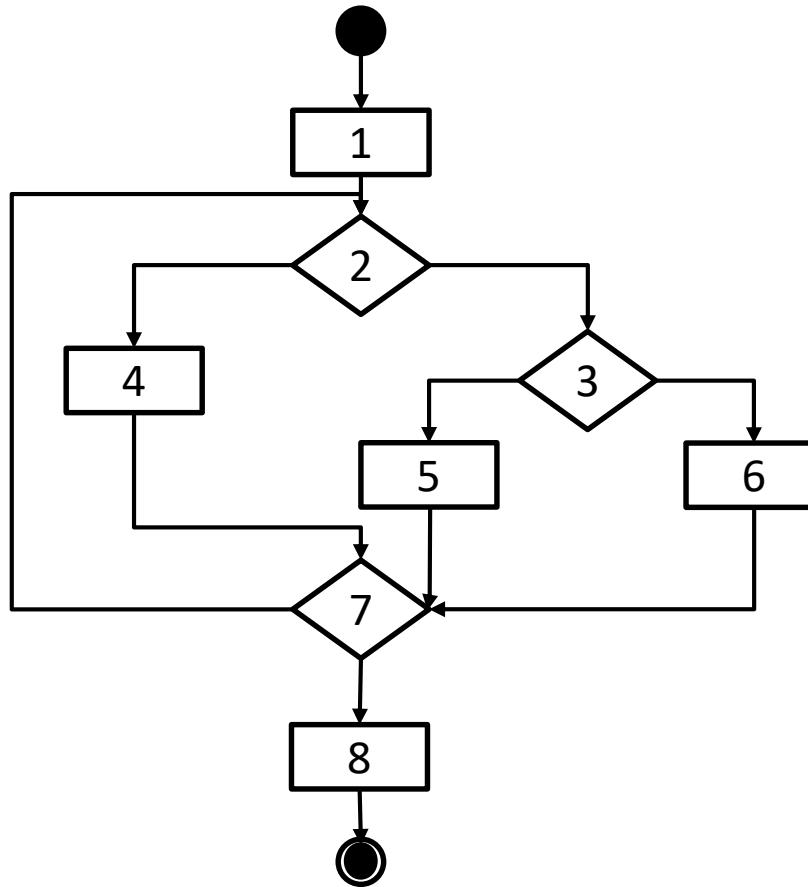
"salário deve estar entre 2.000 e 10.000 reais" → faixa de valores

caso	origem	expectativa	valor
1	pce classe 1 (< 2000) + aleatório	inválido	653.11
2	pce classe 2 (≥ 2000.00 e ≤ 10000.00) + aleatório	válido	7885.90
3	pce classe 1 (> 10000) + aleatório	inválido	224.020.15
4	avl caso 1	inválido	1999.99
5	avl caso 2	válido	2000.00
6	avl caso 3	válido	2000.01
7	avl caso 4	válido	9999.99
8	avl caso 5	válido	10000.00
9	avl caso 6	inválido	10000.01

teste combinatório

- introdução rápida -

software e explosão combinatória



exemplo

parâmetros

valores

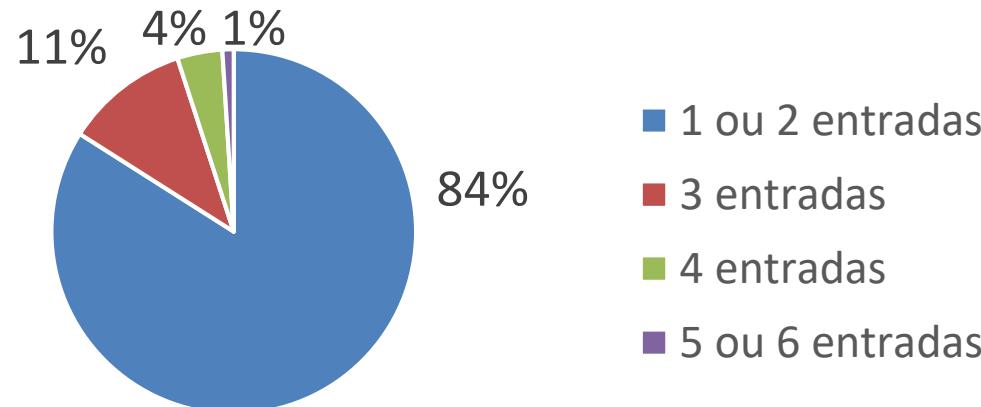
Type	Single, Spanned, Stripped, Mirror, RAID-5
Format Method	Quick, Slow
File System	FAT32, NTFS, Ext4, ReiserFS
Cluster Size	512, 1024, 2048, 4096, 8192, 16384
Compression	On, Off

todas as combinações: $5 \times 2 \times 4 \times 6 \times 2 = 480$ testes

*ignorando que podem haver restrições entre valores (*i.e.*, combinações inválidas)

constatação

Quantas combinações de entradas (parâmetros) são necessárias para detectar defeitos em produção?



WALLACE, D. R. & KUHN, D. R. **Failure modes in medical device software: an analysis of 15 years of recall data.** International Journal of Reliability, Quality and Safety Engineering, 8(4), 2001.

KUHN, D. R. & REILLY, M. J. **An investigation of the applicability of design of experiments to software testing.** In Proceedings of the 27th NASA/IEEE Software Engineering Workshop, NASA Goddard Space Flight Center, 2002.

KUHN, D. R.; WALLACE, D. R; GALLO JR, A. M.. **Software Fault Interactions and Implications for Software Testing.** IEEE Transactions on Software Engineering. pp. 418-421. Volume 30 Issue 6, June 2004.

BELL, K Z. **Optimizing Effectiveness and Efficiency of Software Testing: A Hybrid Approach.** PhD Thesis. North Carolina State University, 2006.

combinação *n-wise*

Exemplo:

parâmetros e valores

A	X, Y
B	W, Z
C	I, J, K

$2 \times 2 \times 3 = 12$ testes

100% cobertura

cada valor
aparece pelo menos
uma vez

1-wise

#	A	B	C
1	X	W	I
2	Y	Z	J
3	X	W	K

3 testes

25% cobertura

cada par de valores
aparece pelo menos
uma vez

2-wise

#	A	B	C
1	X	W	I
2	X	W	K
3	X	Z	J
4	Y	Z	I
5	Y	Z	J
6	Y	Z	K
7	Y	W	J

7 testes

~58% cobertura

cada trio de valores
aparece pelo menos
uma vez*

3-wise

#	A	B	C
1	X	W	I
2	X	W	J
3	X	W	K
4	X	Z	I
5	X	Z	J
6	X	Z	K
7	Y	W	I
8	Y	W	J
9	Y	W	K
10	Y	Z	I
11	Y	Z	J
12	Y	Z	K

12 testes

100% cobertura

*Como há 3 parâmetros, a 3-wise é igual a combinação total.

voltando ao exemplo

Type	Single, Spanned, Stripped, Mirror, RAID-5
Format Method	Quick, Slow
File System	FAT32, NTFS, Ext4, ReiserFS
Cluster Size	512, 1024, 2048, 4096, 8192, 16384
Compression	On, Off

$$5 \times 2 \times 4 \times 6 \times 2 = 480 \text{ testes}$$

	1-wise	2-wise	3-wise	4-wise	5-wise (tudo)
testes	6	30	121	249	480

Cob.

~1%

~6%

~25%

~52%

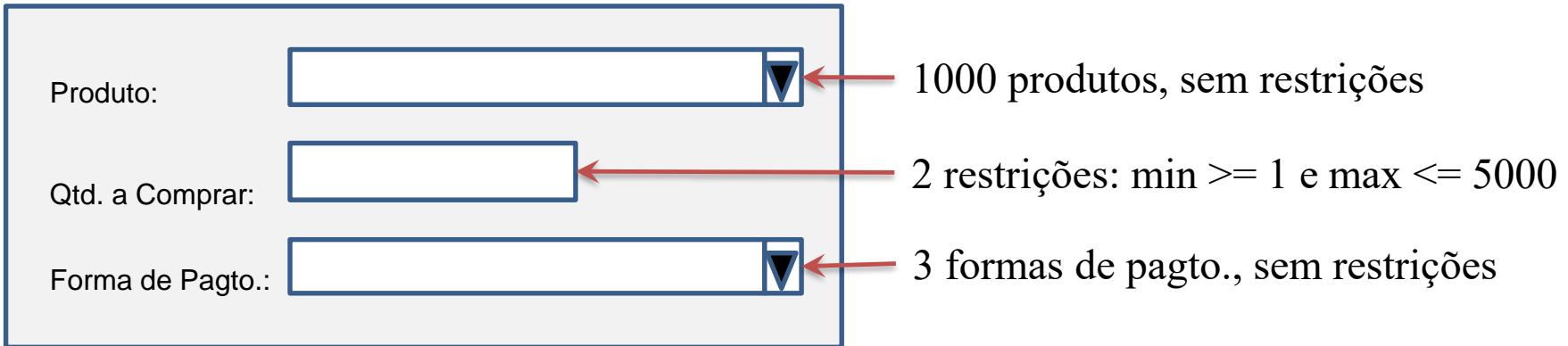
100%



~85% defeitos

redução baseada em modelo – Concordia

Pedido de compra de produto → somente casos válidos (positivos), por simplificação



$$\text{exaustivo} = 1000 \times 2 \times 3 = 6000 \text{ testes}$$

$$1\text{-wise} = 1000 \text{ testes}$$

Suposições (casos válidos):

- 1) Basta 1 exemplar (aleatório) de uma entrada sem restrições → não influencia lógica da aplicação;
- 2) Cada restrição deve ser coberta pelos valores limítrofes → melhoria da cobertura relevante.

$$\text{proposto (básico)} = 1 + 4 + 1 = 6 \text{ testes}$$

(4 é referente a: =1, >1, =5000, <5000)

Concordia

curiosidade

Concordia é a deusa romana que personifica a "concordância", o "acordo"¹.

a ideia que a linguagem seja usada como forma de **aproximar clientes e desenvolvedores**



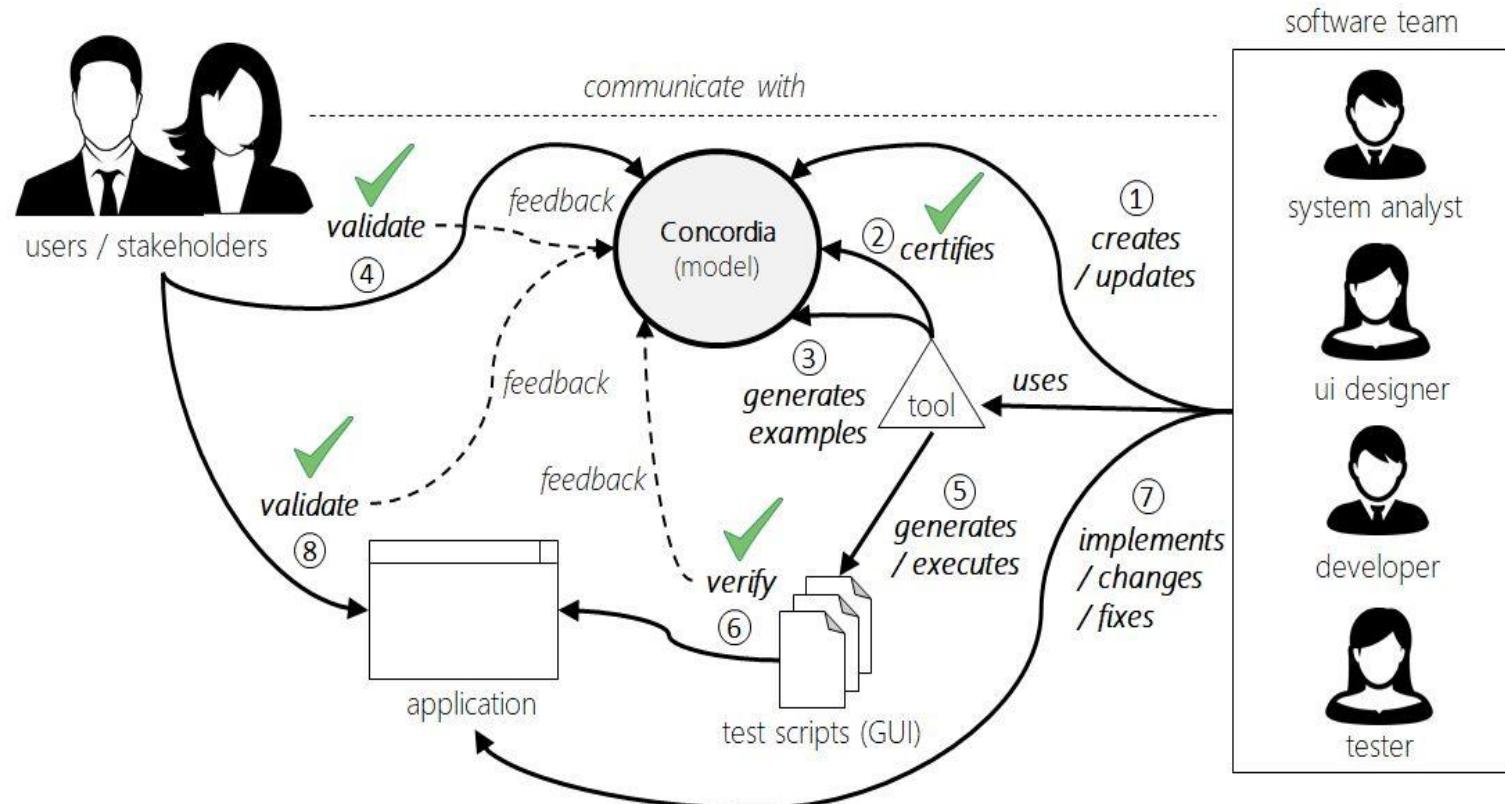
1. <https://www.britannica.com/topic/Concordia-Roman-goddess>

2. [https://it.wikipedia.org/wiki/Tempio_della_Concordia_\(Agrigento\)](https://it.wikipedia.org/wiki/Tempio_della_Concordia_(Agrigento))

Templo de Concordia - Sicília, Itália - 440-430 a.C.²

visão geral

dinâmica



características da linguagem

inspirada em [Gherkin](#)

legível para pessoas envolvidas no negócio (*business-readable*)

separa declarações de *negócio* das "*tecnológicas*"

baseada em dicionário → traduzível, expansível

permite especificar requisitos de qualquer tipo

permite especificar casos de teste e regras de negócio

características da ferramenta

funciona como um **compilador**

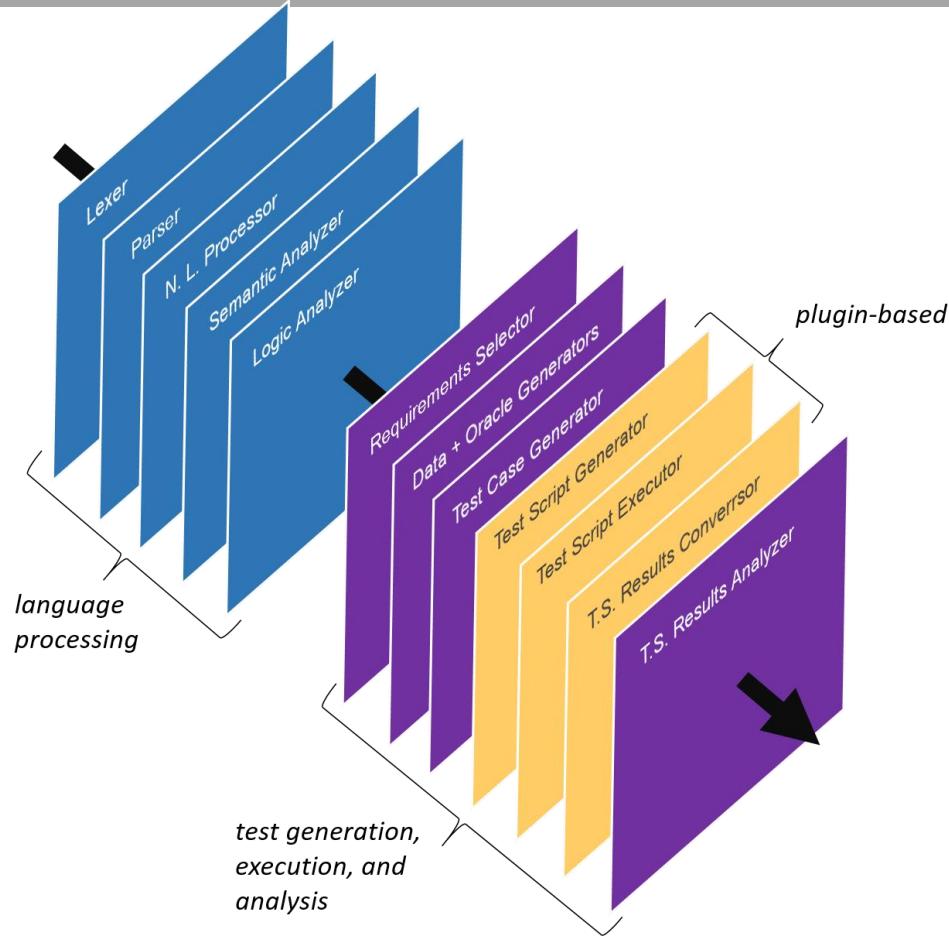
usa **processamento de ling. nat.** e aprendizagem supervisionada
reconhecimento de intenção

gera **casos de teste**

gera e executa **scripts de teste funcional**

analisa resultados da execução

estrutura



desenvolvimento



build passing
license AGPL-3.0
slack chat

SEMANTIC VERSIONING

Breaking
change

Bug
fix

1.2.3

New
feature



```
PASS __tests__\lexer\DatabaseLexerTest.spec.ts
PASS __tests__\util\ReferenceReplacerTest.spec.ts
PASS __tests__\db\QueryParserTest.spec.ts
PASS __tests__\lexer\ImportLexerTest.spec.ts
PASS __tests__\testdata\raw\DoubleGeneratorTest.spec.ts
PASS __tests__\parser\TestCaseParserTest.spec.ts
PASS __tests__\lexer\RegexBlockLexerTest.spec.ts
PASS __tests__\lexer\StepWhenLexerTest.spec.ts
PASS __tests__\parser\TagCollectorTest.spec.ts
PASS __tests__\req\LineCheckerTest.spec.ts
PASS __tests__\testdata\random\RandomStringTest.spec.ts
PASS __tests__\testdata\random\RandomDateTest.spec.ts
PASS __tests__\testdata\random\RandomDoubleTest.spec.ts
PASS __tests__\parser\ScenarioParserTest.spec.ts
PASS __tests__\testdata\random\RandomLongTest.spec.ts
PASS __tests__\testdata\random\RandomTimeTest.spec.ts
PASS __tests__\lexer\ConstantBlockLexerTest.spec.ts
PASS __tests__\lexer\TextLexerTest.spec.ts
PASS __tests__\lexer\RegexLexerTest.spec.ts
PASS __tests__\nlp\RuleBuilderTest.spec.ts
PASS __tests__\req\ExpressionsTest.spec.ts
```

Test Suites: 81 passed, 81 total

Tests: 695 passed, 695 total

Snapshots: 0 total

Time: 16.929s, estimated 29s

Ran all test suites.

como contribuir

divulgando a ferramenta

mais pessoas conhecendo, mais feedback, melhor ela ficará!

usando a ferramenta

informe o que você achou, quais dúvidas que teve ou tem
mostre casos em que ela não funcionou como esperado
mostre como ela pode melhorar para ajudar você ou sua empresa

melhorando a documentação

estendendo, traduzindo, corrigindo
criando um *logotipo*!

como contribuir

registrando **defeitos** ou **sugestões**

criando **novos testes**

pode ser simplesmente criando pequenas variações de coisas que existem

criando **novos plug-ins**

para sua linguagem e framework de teste preferidos

colocando a mão na massa

corrigindo bugs no código

propondo alterações ou melhorias



instalação

```
npm install -g concordialang
```

instalando um plug-in

```
concordia --plugin-install codeceptjs
```

execução

iniciando o servidor de testes (só 1 vez)

```
concordia --plugin-serve codeceptjs
```

rodando

```
concordia --plugin codeceptjs
```

exemplo – busca.feature

#language: pt

Funcionalidade: Busca no Google

Cenário: Busca retorna resultado esperado

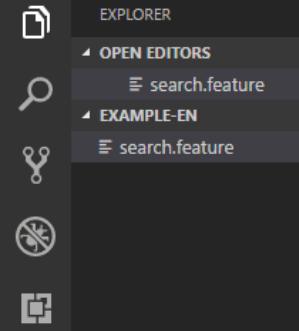
Variante: Busca ao teclar Enter

Dado que estou em "https://google.com.br"

Quando eu informo "concordialang.org" em <q>

e eu pressiono "Enter"

Então eu vejo "npm"



1 Feature: Google Search
2
3 Scenario: Search returns expected result
4
5 Variant: Search content on pressing Enter
6 Given that I am on "https://google.com"
7 When I type "concordialang.org" in <q>
8 | And I press "Enter"
9 | Then I see "npm"

PROBLEMS

TERMINAL

1: cmd

C:\code\tmp\example-en>

exemplo – execução

```
concordia --plugin codeceptjs
```

irá gerar

`busca.testcase`

`test/busca.js`

e irá executar o teste `test/busca.js`

arquivos

FUNCIONALIDADE



.feature

CASOS DE TESTE



. testcase

SCRIPTS DE TESTE



.???

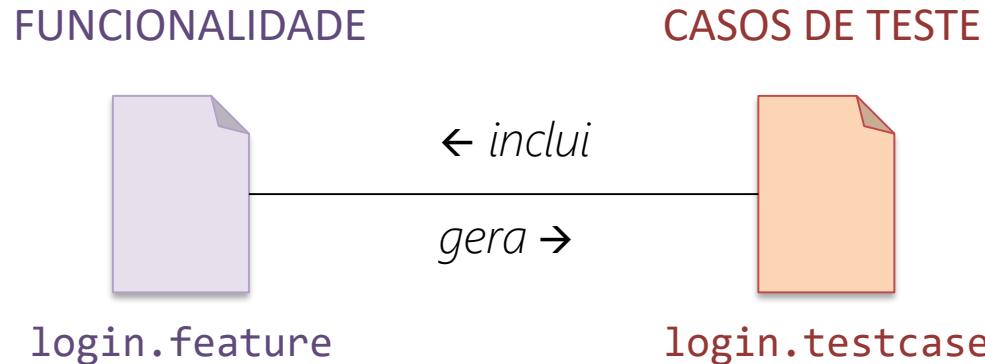
exemplo

FUNCIONALIDADE



`login.feature`

exemplo



exemplo

FUNCIONALIDADE



`login.feature`

CASOS DE TESTE



`login.testcase`

SCRIPTS DE TESTE



`login.js`

$\leftarrow inclui$

$gera \rightarrow$

$gera \rightarrow$

exemplo

FUNCIONALIDADE



login.feature

CASOS DE TESTE



login.testcase

SCRIPTS DE TESTE



login.js

← *incluir*

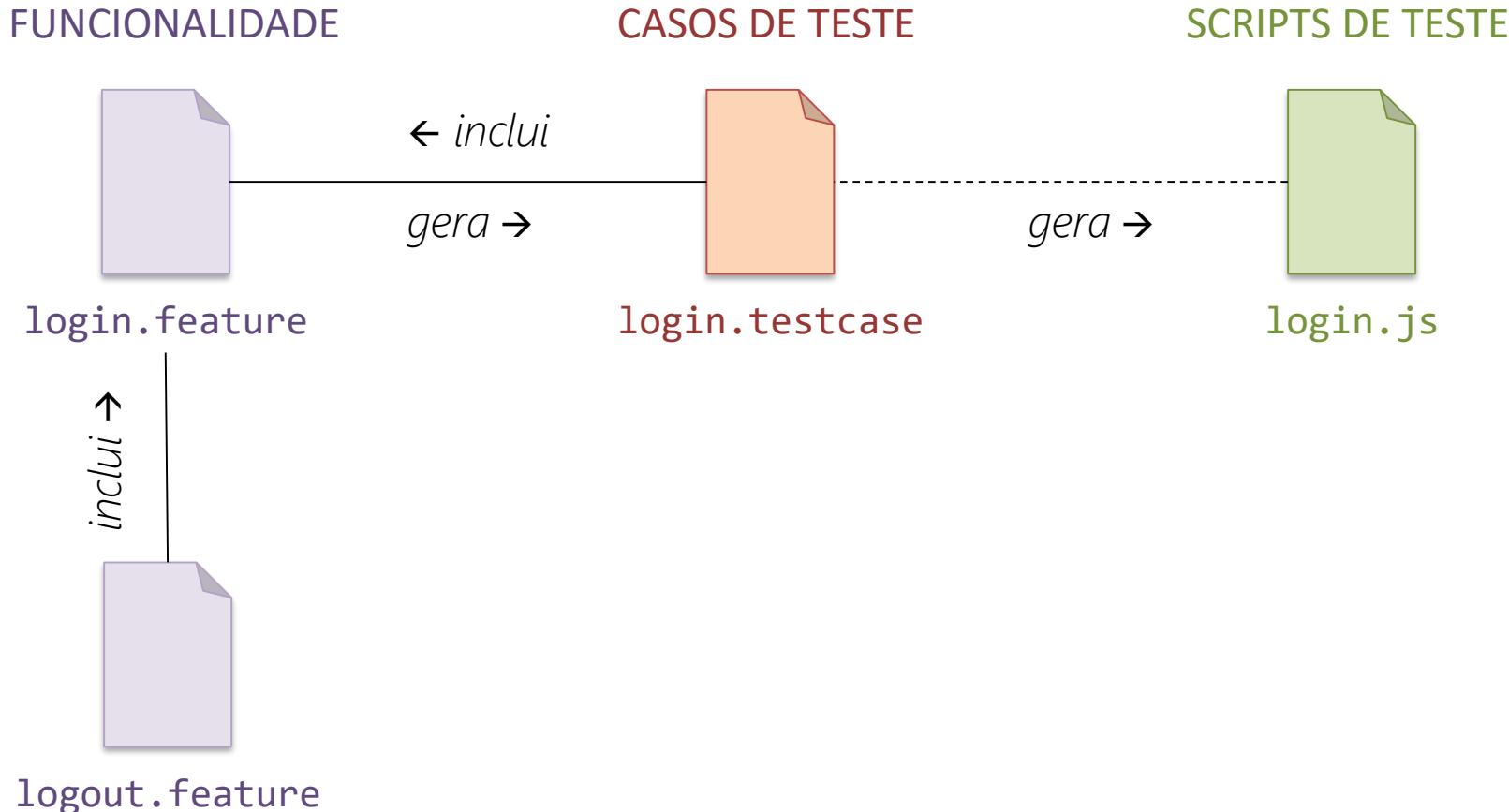
gera →

gera →

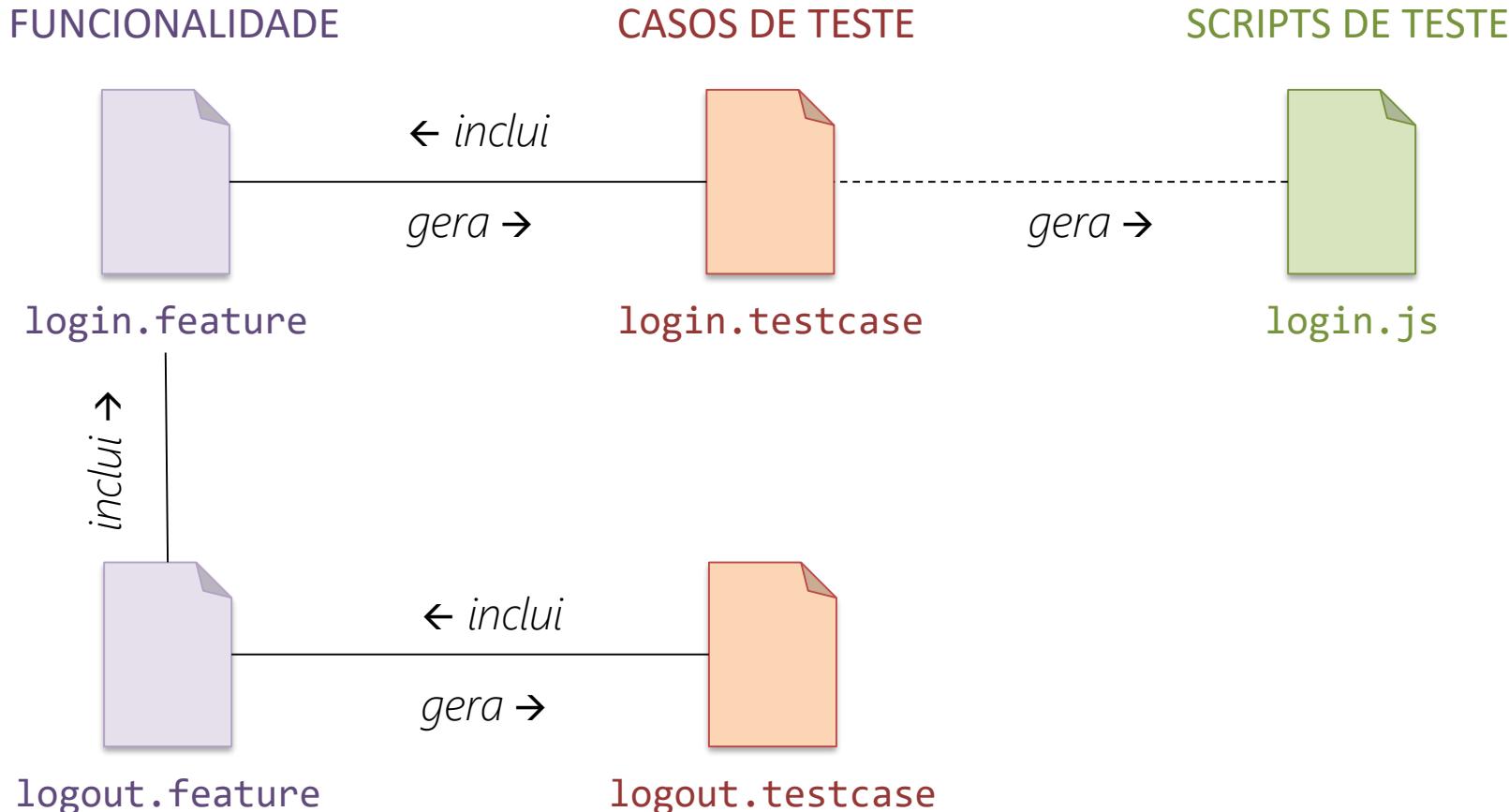


logout.feature

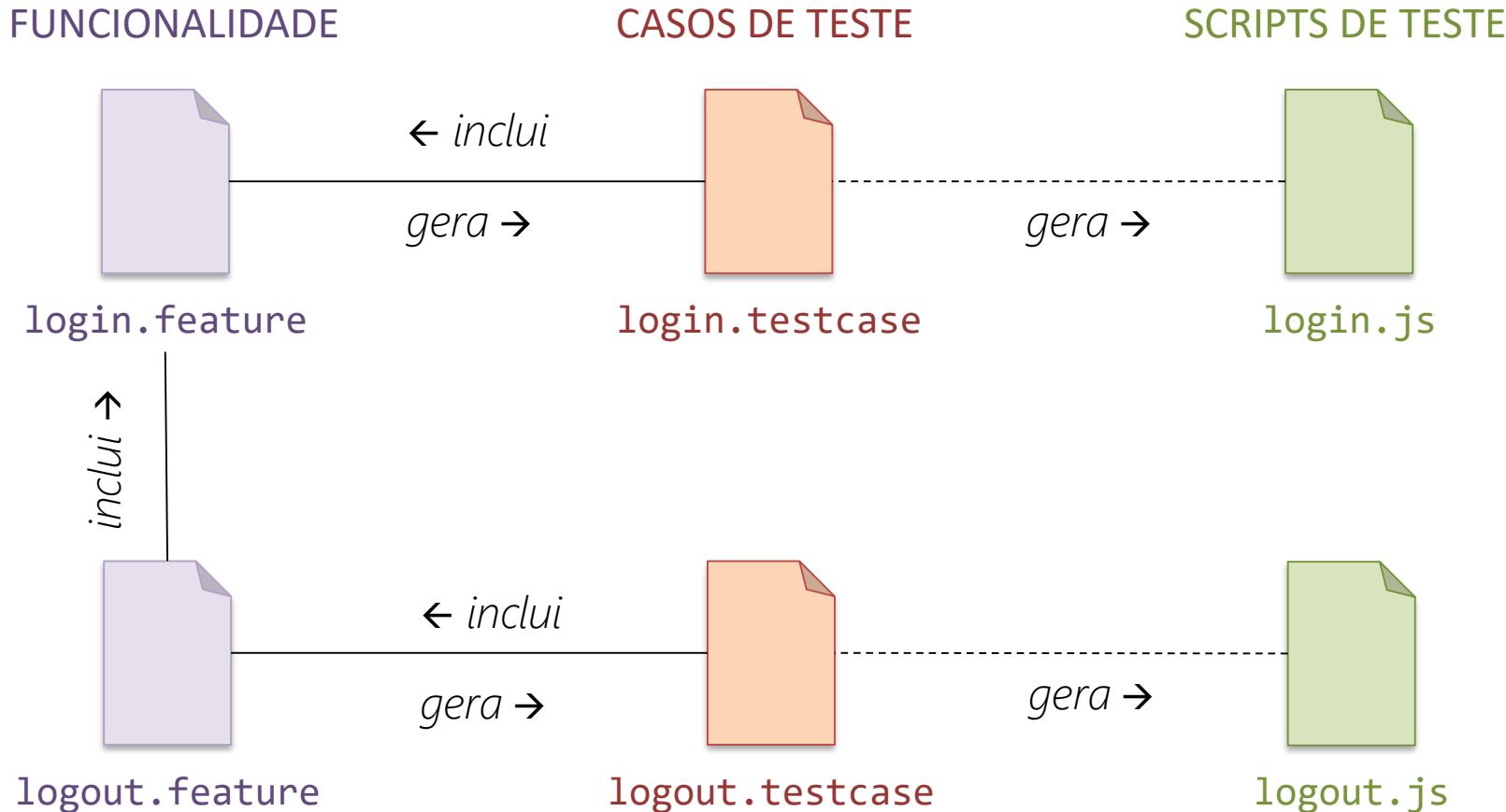
exemplo



exemplo



exemplo



linguagem

DSLs compatíveis com Gherkin

Funcionalidade: <título>

Como um <papel desempenhado>

Desejo <meta>

Para <benefício obtido>



História de Usuário (opcional)

Cenário: <título>

Dado que <contexto>

Quando <evento ocorre>

Então <resultado deve ocorrer>



Descrição de Cenário

exemplo

Funcionalidade: Ganhar milhas ao voar

Como um viajante

Desejo poder acumular milhas toda vez que voar

Para poder viajar de graça ou ter descontos

Cenário: Um novo viajante começa com nível Bronze

Dado que não sou um cliente da companhia Kaikai

Quando eu me cadastro no programa de milhas

Então eu recebo o nível Bronze

começando com uma Funcionalidade

#language: pt

indica a língua usada na especificação

Funcionalidade: Logout

não necessita de História de Usuário, se não acrescenta valor !

declarando um Cenário

#language: pt

Funcionalidade: Logout

Cenário: Vai para a página principal ao sair

Dado que estou logado no site

Quando eu aciono a opção de sair

Então sou redirecionado para a página principal do site

Cenário em linguagem não tecnológica

declarando uma Variante

```
#language: pt
```

Funcionalidade: Logout

Cenário: Vai para a página principal ao sair

Dado que estou logado no site

Quando eu aciono a opção de sair

Então sou redirecionado para a página principal do site

Variante: Sai quando aciona opção Sair

Quando eu clico em <Sair>

Então eu vejo a url "/"

Variante expressa interação com sistema

analisando de perto a Variante

#language: pt

Funcionalidade: Logout

Cenário: Vai para a página principal ao sair

Dado que estou logado no site

Quando eu aciono a opção de sair

Então sou redirecionado para a página principal do site

também
chamado de
Literal de IU

Variante: Sai quando aciona opção Sair

Quando eu clico em <Sair>

Então eu vejo a url "/"

<Sair> é um identificador de elemento da IU

analisando de perto a Variante

#language: pt

Funcionalidade: Logout

Cenário: Vai para a página principal ao sair

Dado que estou logado no site

Quando eu aciono a opção de sair

Então sou redirecionado para a página principal do site

Variante: Sai quando aciona opção Sair

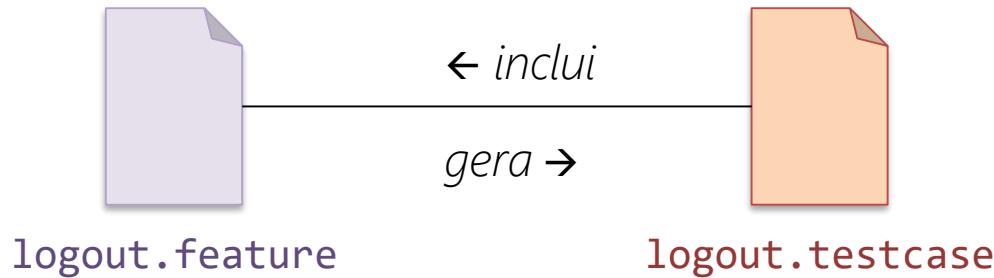
Quando eu clico em <Sair>

Então eu vejo a url "/"

"/" é um valor

gerando um caso de teste

```
$ concordia --just-testcase
```



caso de teste

```
#language: pt  
importe "login.feature"
```

importa definições de Funcionalidades

@generated tag que indica que o Caso de Teste foi gerado

@scenario(1) tag que referencia Cenário pelo índice

@variant(1) tag que referencia Variante pelo índice

Caso de Teste: Sai quando aciona opção Sair - 1

Quando eu clico em <Sair>

Então eu vejo a url "/"

Caso de Teste não diferiu da Variante,
pois não houve nada para gerar

voltando à Variante

#language: pt

Funcionalidade: Logout

Cenário: Vai para a página principal ao sair

Dado que estou logado no site

Quando eu aciono a opção de sair

Então sou redirecionado para a página principal do site

Variante: Sai quando aciona opção Sair

Quando eu clico em <Sair>

Então eu vejo a url "/"

Verificação não
está ocorrendo na
Variante.
Vamos incluir!

voltando à Variante

```
#language: pt  
importe "login.feature"
```

importa definições de Login

Funcionalidade: Logout

Cenário: Vai para a página principal ao sair

Dado que estou logado no site

Quando eu aciono a opção de sair

Então sou redirecionado para a página principal do site

Variante: Sai quando aciona opção Sair

Dado que tenho ~usuário logado~

Quando eu clico em <Sair>

Então eu vejo a url "/"

nesse caso,
produzido
por "Login"

~usuário logado~ é um estado do sistema

em Login

#language: pt

Funcionalidade: Login

Cenário: Loga com sucesso

Dado que estou na página de login

Quando eu informo minhas credenciais

Então consigo acessar o sistema

Variante: Entra com credenciais de Administrador

Dado que estou em "/login"

Quando eu preencho <#usuario> com "admin"

e preencho <#senha> com "123456"

e clico em <#entrar>

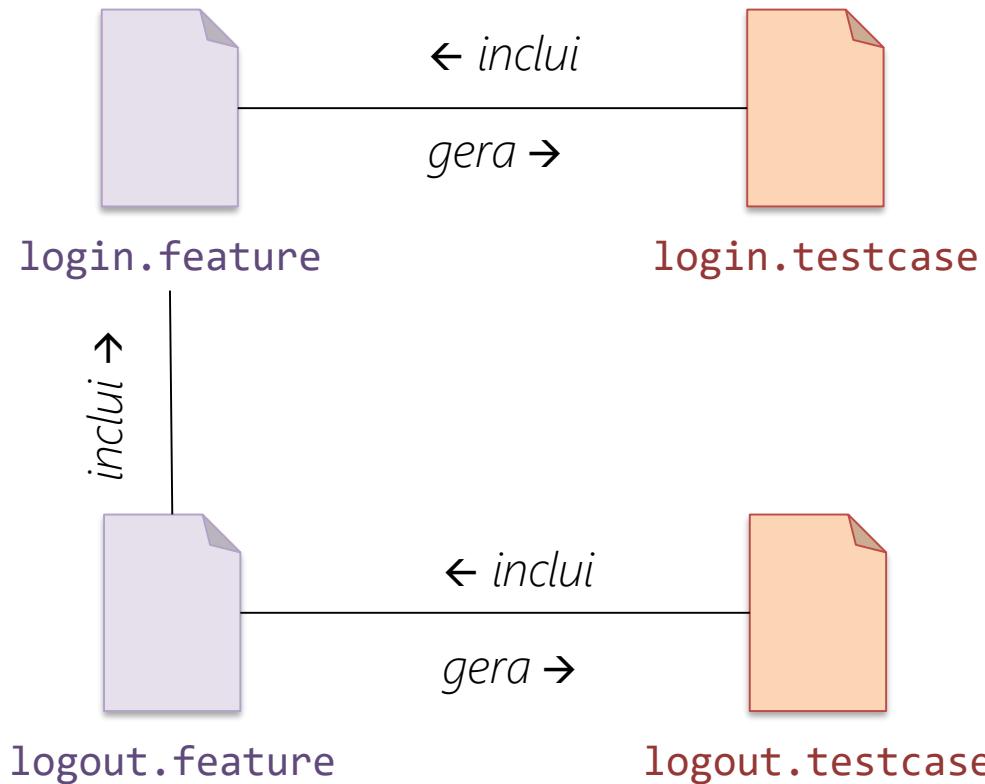
Então eu vejo "Olá"

e tenho ~usuário logado~

~usuário logado~ é produzido

regerando casos de teste

```
$ concordia --just-testcase
```



voltando ao caso de teste de Logout

```
#language: pt  
importe "login.feature"
```

```
@generated  
@scenario( 1 )  
@variant( 1 )
```

Caso de Teste: Sai quando aciona opção Sair - 1

Dado que estou em "/login"

Quando eu preencho <#usuario> com "admin"
e preencho <#senha> com "123456"

e clico em <#entrar>

Então eu vejo "Olá"

Quando eu clico em <Sair>

Então eu vejo a url "/"

Dado que tenho ~usuário logado~

estados do sistema

aparecendo em um **Dado que** denota uma pré-condição
necessidade que tenha executado antes

aparecendo em um **Quando** denota uma chamada
necessidade que seja executado agora

aparecendo em um **Então** denota uma pós-condição
estado é produzido

aparecendo em um **Dado que** denota uma pré-condição
necessidade que tenha executado antes

Dado que tenho ~usuário logado~

...

aparecendo em um **Quando** denota uma **chamada**
necessidade que seja executado agora

...

Quando tenho ~cidade cadastrada~

...

aparecendo em um **Então** denota uma pós-condição
estado é produzido

...

Então tenho ~usuário logado~

observações sobre estados

a ferramenta o **procura nos arquivos importados** quando um **estado** é referenciado, em **Dado que** ou **Quando**

se houver **mais de um produtor** do estado, mais de um Caso de Teste *pode* ser gerado

cada um Caso de Teste pode cobrir uma **combinação** diferente depende do **algoritmo** escolhido (opção da linha de comando)

definindo restrições para Login

...

Variante: Acessa com credenciais

Dado que estou em "/login"

Quando eu preencho {Usuario}, {Senha}
e clico em <#entrar>

Então eu vejo "Olá"

e tenho ~usuário logado~

Elemento de IU: Usuario

- id é "#usuario"
- valor vem da consulta "SELECT usuario FROM [Usuarios]"

Caso contrário eu vejo "Usuário ou senha inválidos"

Elemento de IU: Senha

- id é "#senha"
 - valor vem da consulta "SELECT senha FROM [Usuarios] WHERE usuario = {Usuario}"
- Caso contrário eu vejo "Usuário ou senha inválidos"

Tabela

Referência para Elementos de IU

Tabela: Usuarios

usuario	senha
admin	123456
gerente	ger123

Elemento de IU

Restrição

Expectativa quando restrição ignorada

Elemento de IU

Restrição

Expectativa quando restrição ignorada

representa um *widget* presente na tela

sintaxe comum:

Elemento de IU: <nome>

- <propriedade> <operador> <conteúdo>

[Caso contrário <sentença-1>]

[e <sentença-2>]

...

[e <sentença-N>]

- . . .

Elemento de IU

por *default*

tem propriedade **tipo** com valor **textbox**

tem propriedade **tipo de dado** com valor **string**

tem propriedade **id** igual ao *nome*, convertido para *camel case*

por exemplo, Elemento de IU: Nome Completo é o mesmo que

Elemento de IU: Nome Completo

- id é "nomeCompleto"
- tipo é textbox
- tipo de dado é string

Elemento de IU - propriedades

- **id** é a identificação do elemento de IU na tela
- **tipo** é o tipo do elemento de IU
- **tipo de dado** é o tipo de dado aceito
- **editável** indica se pode ser editado pelo usuário
- **obrigatório** indica se é obrigatório
- **formato** define uma *expressão regular* como formato aceito
- **valor** define um valor para o elemento
- **valor mínimo** define um valor mínimo
- **valor máximo** define um valor máximo
- **comprimento mínimo** define um comprimento mínimo
- **comprimento máximo** define um comprimento máximo



sentença Caso contrário

estabelece o **comportamento esperado** quando um valor de entrada não **satisfaz a restrição** declarada

aceita para as propriedades

obrigatório

formato

valor

valor mínimo

valor máximo

comprimento mínimo

comprimento máximo

equivale a uma sentença **Então**

tabela

define uma tabela de dados, consultável via SQL

tipo dos dados são inferidos

observação: datas devem estar no formato ano/mês/dia

sintaxe

Tabela: <nome>

	<coluna-1>		<coluna-2>		...		<coluna-K>	
	<valor-1>		<valor-2>		...		<valor-K>	
...								
	<valor-1N>		<valor-2N>		...		<valor-KN>	

banco de dados

permite usar um arquivo ou banco de dados externo

atualmente Firebase, MySQL, PostgreSQL, SQLite, MS Access, SQL Server, JSON, CSV,INI e Excel

sintaxe:

Banco de Dados: <nome>

- <propriedade> é "<valor>"



banco de dados - exemplos

Banco de Dados: Usuarios

- tipo é "json"
- caminho é "/caminho/ate/usuarios.json"

Banco de Dados: BD de Teste

- tipo é "mysql"
- nome é "testdb"
- usuário é "root"
- senha é ""



banco de dados - propriedades

uso **depende** do banco de dados utilizado

muitas assumem valores *default* do banco de dados
por exemplo, a **porta** padrão do MySQL é 3050

propriedades

- tipo
- nome
- caminho
- hospedeiro (ou host)
- porta
- usuário
- senha
- opções



constantes

bloco de declarações, somente um por arquivo

sintaxe

Constantes:

- "<nome>" é <valor>
- "<nome-N>" é <valor-N>

exemplo

Constantes:

- "Login" é "/login"
- "PI" é 3.14159
- "Tamanho Mínimo" é 2



referências

para Elementos de IU

{Nome do Elemento} ou {Funcionalidade:Nome do Elemento}

para Constantes, Tabelas e Bancos de Dados

[Nome]

para Estados

~Estado~



referências – uso

Elementos de IU podem ser referenciados em

- Variantes

- Elementos de IU (propriedades)

- Consultas

Constantes podem ser referenciadas em

- Variantes

- Elementos de IU (propriedades)

- Consultas

- Bancos de Dados (propriedades)

Tabelas e Bancos de Dados podem ser referenciados em consultas

Estados podem ser referenciados em passos de **Variantes**

referências – exemplos

Banco de Dados: Usuarios

- tipo é "json"
- caminho é "usuarios.json"

Constantes:

- "Flag de Admin" é "A"
- "Cores" é ["Verde", "Azul", "Preto"]

Elemento de IU: Usuario

- valor vem de "SELECT usuario FROM [Usuarios] WHERE flag = [Flag de Admin]"

Elemento de IU: Cor do Tema

- valor está em [Cores]

casos de teste baseado em regras

required

REQUIRED_FILLED
REQUIRED_NOT_FILLED

format

FORMAT_VALID
FORMAT_INVALID

value is in <set or query>

SET_FIRST_ELEMENT
SET_RANDOM_ELEMENT
SET_LAST_ELEMENT
SET_NOT_IN_SET

minimum/maximum value

VALUE_LOWEST
VALUE_RANDOM_BELOW_MIN
VALUE_JUST_BELOW_MIN
VALUE_MIN
VALUE_JUST_ABOVE_MIN
VALUE_ZERO
VALUE_MEDIAN
VALUE_RANDOM_BETWEEN_MIN_MAX
VALUE_JUST_BELOW_MAX
VALUE_MAX
VALUE_JUST_ABOVE_MAX
VALUE_RANDOM_ABOVE_MAX
VALUE_GREATEST

minimum/maximum length

LENGTH_LOWEST
LENGTH_RANDOM_BELOW_MIN
LENGTH_JUST_BELOW_MIN
LENGTH_MIN
LENGTH_JUST_ABOVE_MIN
LENGTH_MEDIAN
LENGTH_RANDOM_BETWEEN_MIN_MAX
LENGTH_JUST_BELOW_MAX
LENGTH_MAX
LENGTH_JUST_ABOVE_MAX
LENGTH_RANDOM_ABOVE_MAX
LENGTH_GREATEST

conclusões

possíveis usos para Concordia (1 de 3)

1. Especificar requisitos em mais de uma língua falada usando texto simples;
2. Validar requisitos com clientes;
3. Discutir requisitos e casos de teste entre a equipe de software (uso como mídia de comunicação);
4. Especificar casos de teste funcionais em linguagem natural (restrita);
5. Checar erros sintáticos, semânticos e lógicos em especificações;

possíveis usos para Concordia (2 de 3)

6. Gerar, executar e analisar casos de teste e scripts de teste funcional completos;
7. Usar fontes de dado externas, como bancos de dados, para criar restrições sobre elementos de IU e produzir casos de teste;
8. Descobrir defeitos, especialmente em aplicações recentes;
9. Verificar a correspondência de uma aplicação com sua especificação Concordia;
10. Apoiar a adoção de BDD, ATDD e SbE;

possíveis usos para Concordia (3 de 3)

11. Apoiar a adoção de testes funcionais em aplicações novas ou legadas;
12. Suportar *test-driven maintenance*;
13. Usar uma abordagem de manutenção orientada a modificação dos requisitos – isso é, muda os requisitos, usa a ferramenta para produzir os respectivos testes e então modifica a aplicação para passar nos testes;
14. Separar declarações de negócio das declarações em nível de teste;
15. Definir eventos de teste em linguagem natural (restrita) para configurar o estado de aplicações antes ou depois da execução de scripts de teste.



perguntas