



# Aula 0: Afinal o que é testar?

Prof. Thiago

Departamento de Informática e Matemática  
Aplicada - DIMAp

Quando pensamos em  
Teste de Software algumas  
perguntas surgem...



Teste = Qualidade??

Afinal o que é testar??

Teste = Qualidade??

---

Afinal o que é testar??

Vamos fazer um paralelo...

# Linka de Montagem



muito se gasta na construção...



Se o resultado for...



Se o resultado for...



não tem teste que dê jeito

Ou seja...

"A qualidade é um compromisso de TODOS: desenvolvedores, gerentes, arquitetos, testadores"

James Whitaker



" O teste é parte integrante do processo de construção. "

"Não é algo apenas feito para garantir qualidade ao final."

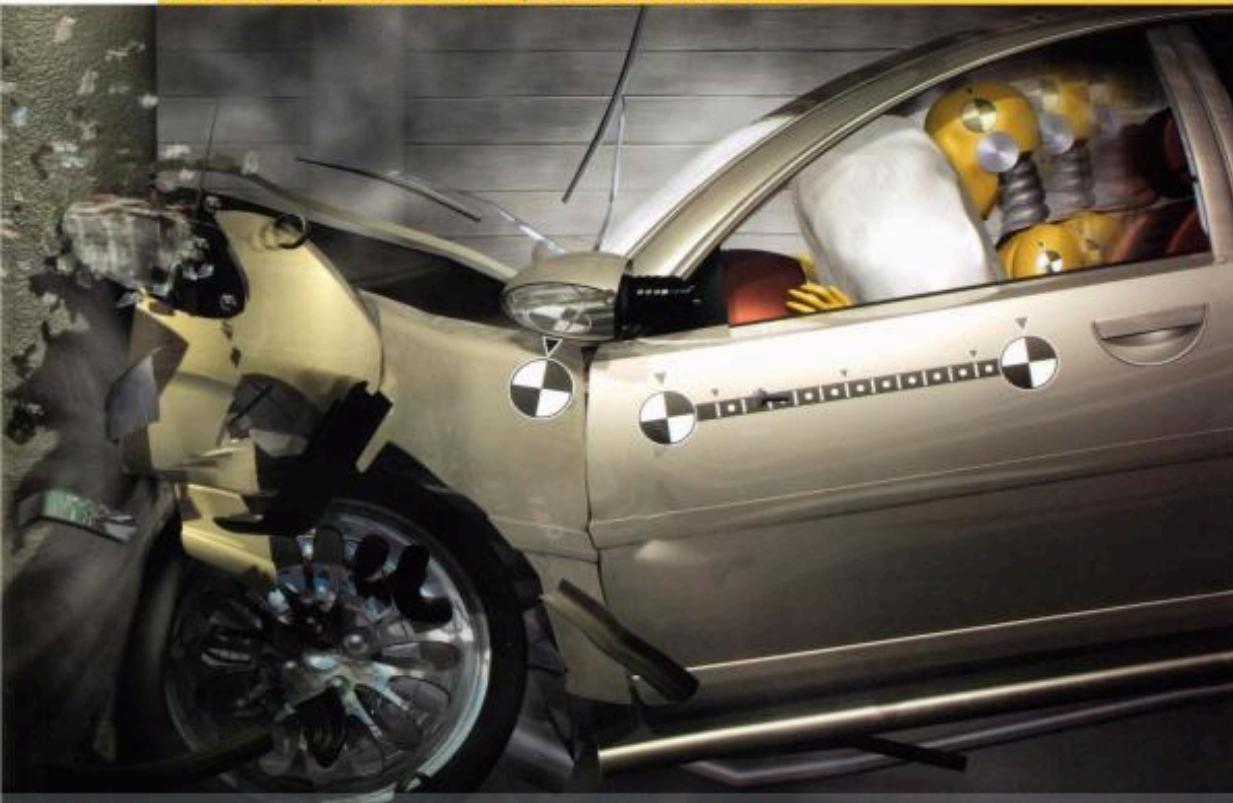
James Whitaker

Uma vez passado pelo teste do desenvolvedor, os testadores podem fazer testes mais rigorosos



# SOFTWARE TESTING AND ANALYSIS

PROCESS, PRINCIPLES, AND TECHNIQUES

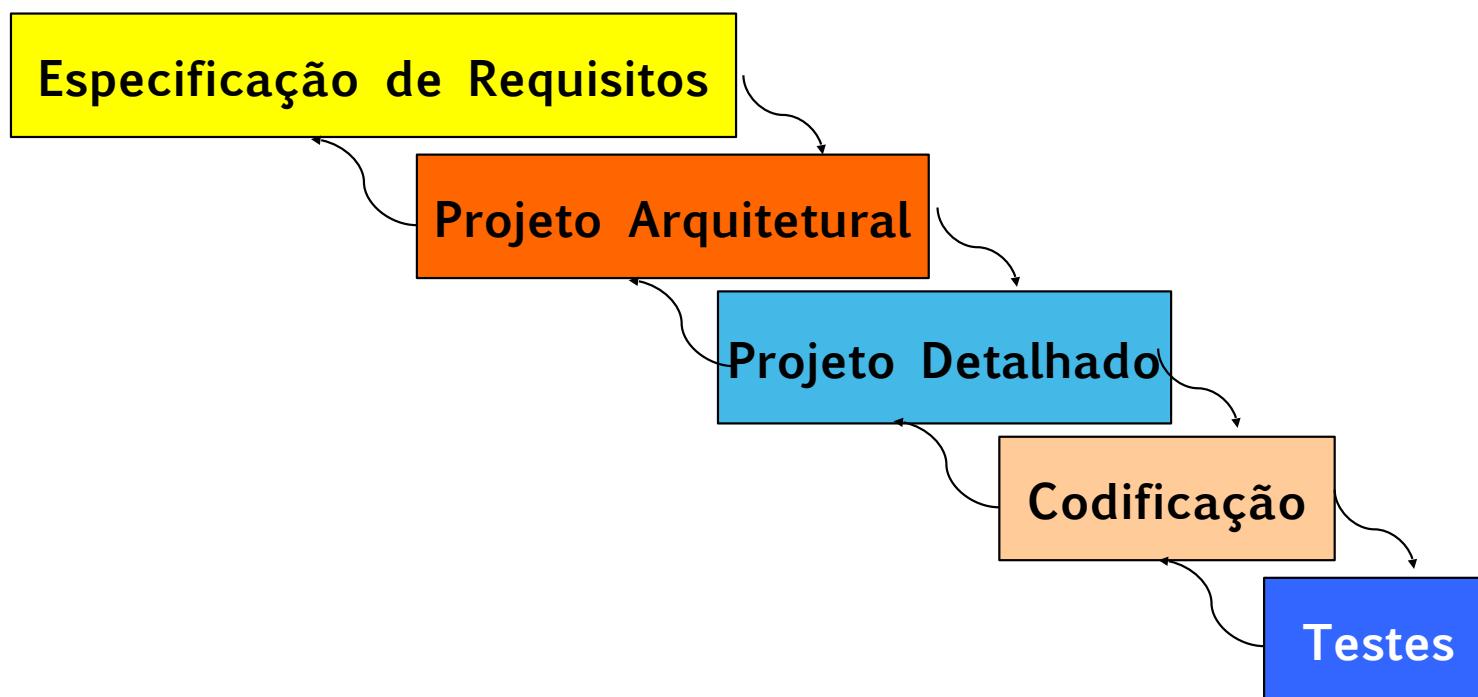


Front Cover

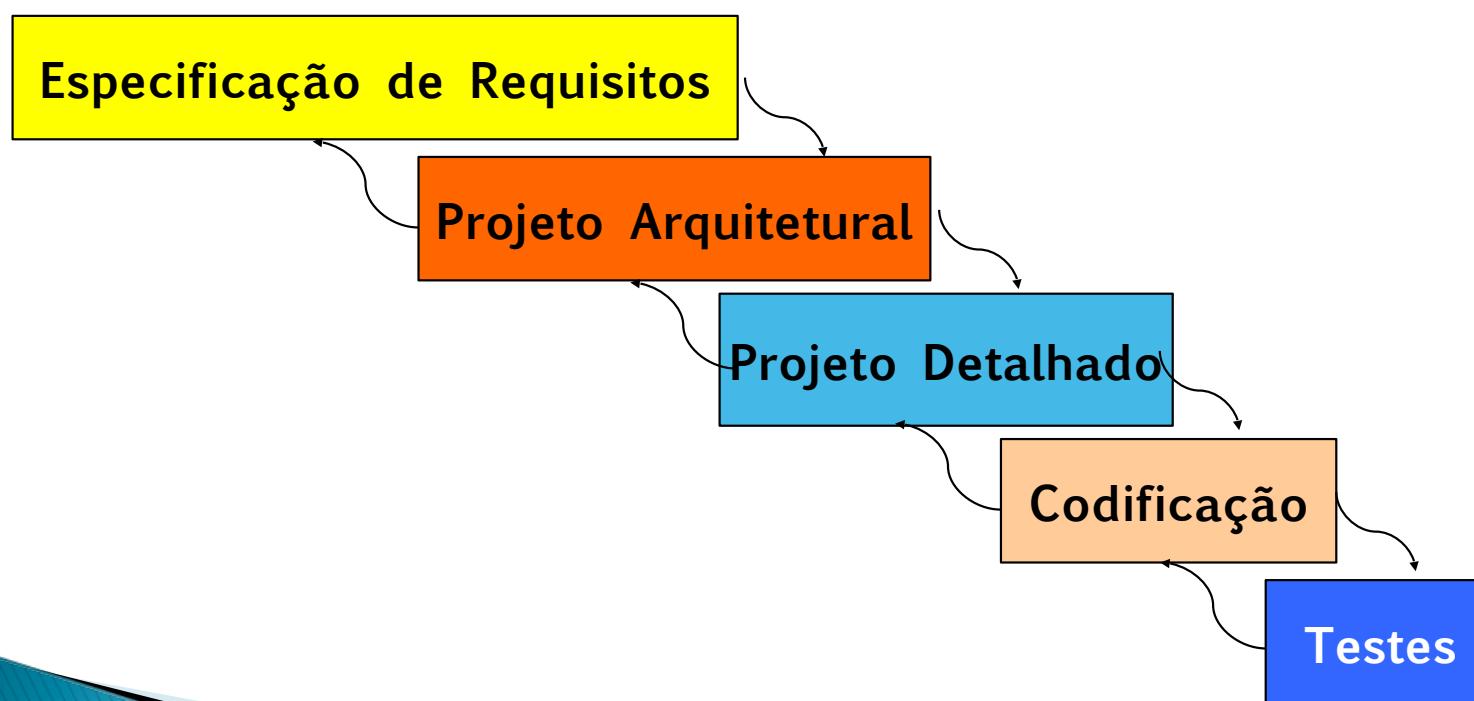
Mauro Pezzè  
Michal Young

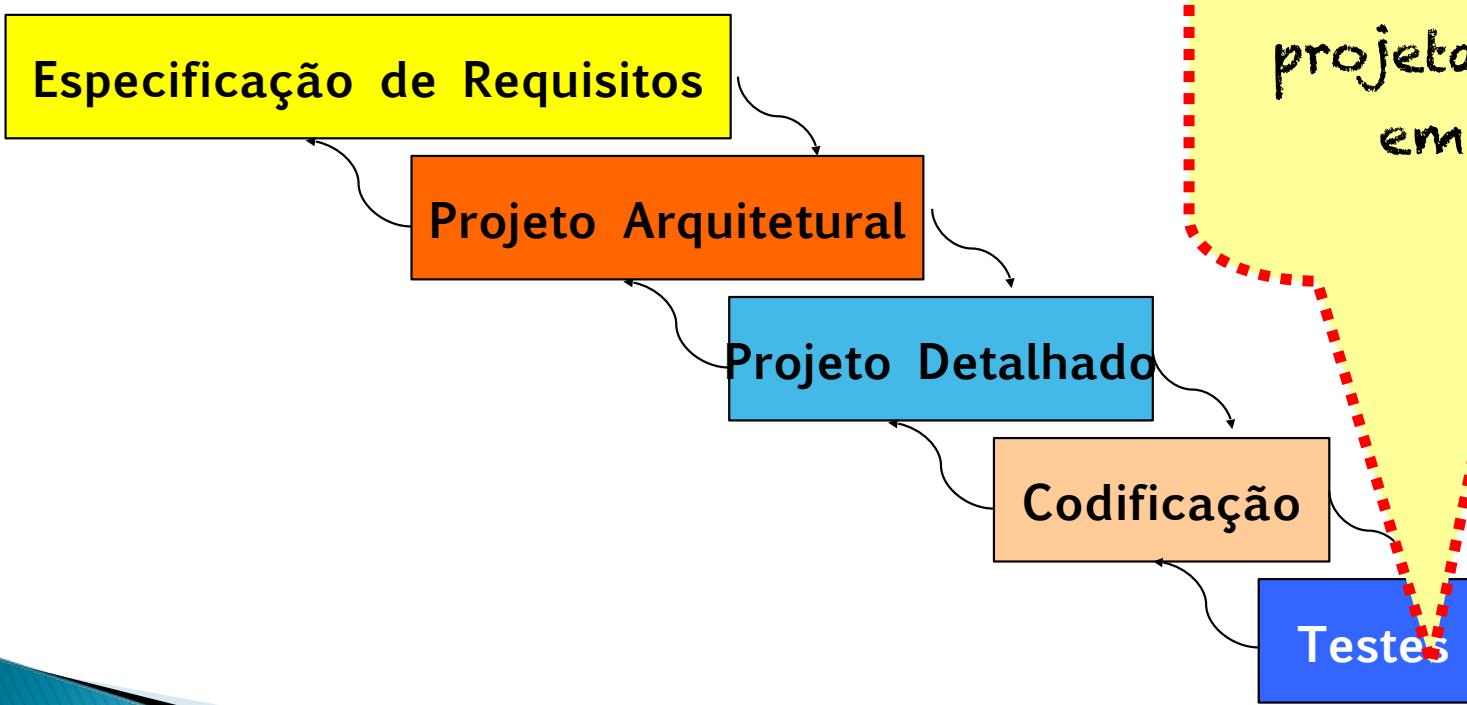
Testes não é algo que você deve  
pensar só no final de cada  
iteração...

Até pouco tempo atrás... os testes eram vistos como uma etapa posterior a implementação (Big-Bang Testing)

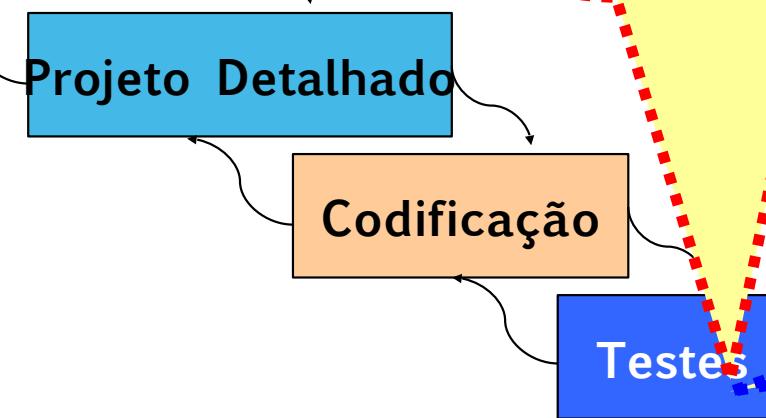
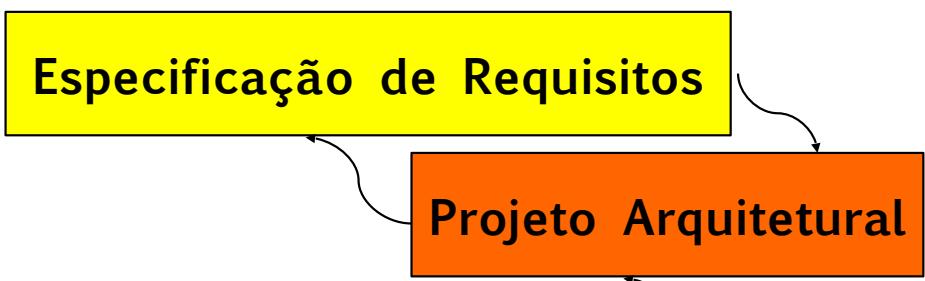


Voce vê algum  
problema nisso??





Se o software não foi  
projeto pensando  
em testes...??



Se o software não foi  
projeto pensando  
em testes...??

Agora Inês é  
morta....



## How To Write Hard To Test Code? (Misko Hevery - Google Talk)



Se o software não foi  
projeto pensando  
em testes...??

Agora Inês é  
morta....

*Teste não é a cobertura do bolo!*



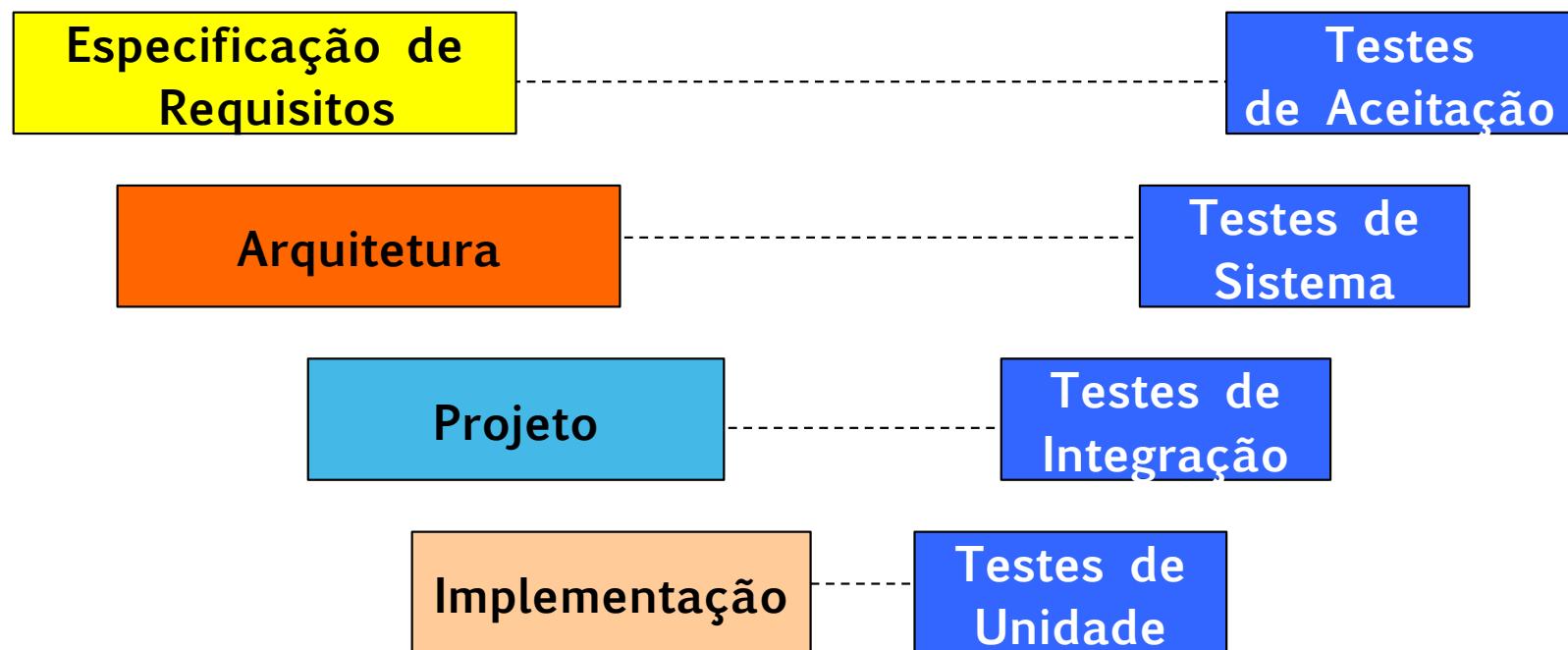
# Quando Testar?

Testes não são mais vistos como **uma fase** que inicia após a implementação.

Testes são **um processo PARALELO** ao processo de desenvolvimento.

# Quando Testar?

Modelo V



E então....

Teste = Qualidade??

"Quality is not equal to test. Quality is achieved by putting development and testing into a blender and mixing them until one is indistinguishable from the other."

James Whitaker

Não. O teste contribui para a qualidade como todas as etapas do desenvolvimento.

Teste = Qualidade??

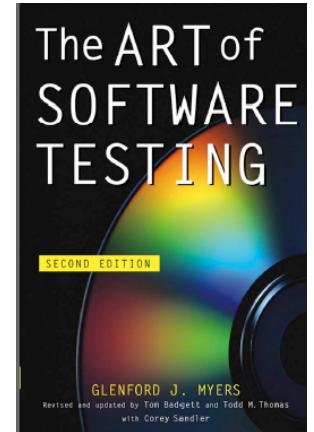
Afinal o que é testar??

Teste = Qualidade??

Afinal o que é testar??



# Entrevista



Durante uma entrevista  
desenvolvedores precisavam  
responder a esta pergunta.

# Algumas Respostas

1. "O objetivo dos testes é mostrar que **erros não estão** presentes no software."
2. "Testes procuram **assegurar** que o programa faz o que se supõe que ele faça."

# Algumas Respostas

1. "O objetivo dos testes é garantir que **erros não estão presentes no software.**"
  2. "Testes garantem que o sistema faz o que se supõe que ele faça."
- Você vê algum problema nestas definições?*

**funcao(int a, int b, int c)**



```
funcao(int a, int b, int c)
```

O que preciso fazer para  
assegurar que a função não  
contém falhas??





Para **ASSEGURAR**  
você precisaria  
executar a função  
para **TODAS AS**  
**ENTRADAS**  
**POSSÍVEIS!!**

```
funcao(int a, int b, int c)
```

int varia de -2.147.483.648 a 2.147.483.647

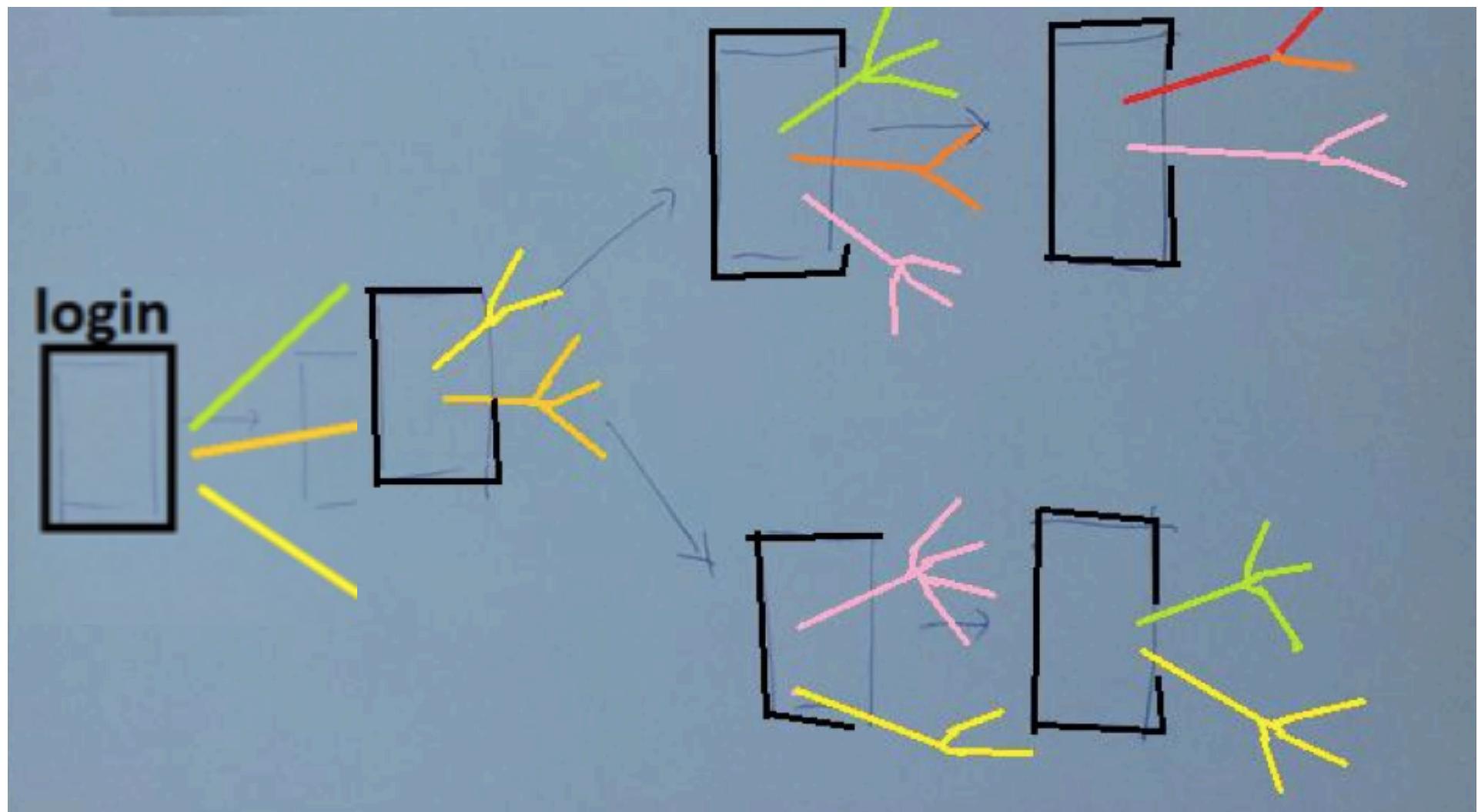
Então o número de entradas possíveis:

9.903.520.314.283.042.199.192.993.792



**Agora imagine em um  
sistema com...**

**Muuitoos casos  
de uso, com  
várias telas, e  
vários campos  
cada um!!**



O número de casos de testes possíveis é  
**ASTRONÔMICO** para a maioria dos sistemas!!!



# Dá para perceber que...



Dá para perceber que...

Testar todas as entradas é  
praticamente inviável!!

Vamos sempre testar para um  
subconjunto das entradas.

Testar é como procurar  
agulhas num palheiro...





Se encontramos uma não significa  
que não existam outras...

# Incompletude dos Testes

“O teste pode revelar a presença de falhas em um software, mas nunca a sua ausência.”

Dijkstra

Um exemplo...

# Testar consiste em...

**DADOS DO DISCENTE**

Matrícula: \*

Nível: \*

A pessoa é estrangeira e não possui CPF

CPF: \*

Nome: \*

RG: \*  (Digite apenas os números)

Data de Nascimento: \*

E-Mail: \*

Ano/Semestre Inicial  -  (Ex.: 2006-2)

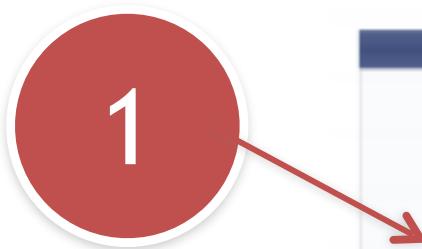
Login: nome.sobrenome

Senha: \*

Confirmar Senha: \*

# Passo 1: Pensar em dados de entrada

1



**DADOS DO DISCENTE**

Matrícula: ★ 2012000482

Nível: ★ GRADUAÇÃO

A pessoa é estrangeira e não possui CPF

CPF: ★ 999.999.999-99

Nome: ★ Nome Sobrenome

RG: ★ 4567897 (Digite apenas os números)

Data de Nascimento: ★ 12/10/1990

E-Mail: ★ email@email.com.br

Ano/Semestre Inicial ★ 2012 - 1 (Ex.: 2006-2)

Login: nome.sobrenome

Senha: ★ \*\*\*\*\*

Confirmar Senha: ★ \*\*\*\*\*

# Passo 2: Checar a saída

**DADOS DO DISCENTE**

Matrícula: ★ 2012000482  
Nível: ★ GRADUAÇÃO  
 A pessoa é estrangeira e não possui CPF  
CPF: ★ 999.999.999-99  
Nome: ★ Nome Sobrenome  
RG: ★ 4567897 (Digite apenas os números)  
Data de Nascimento: ★ 12/10/1990  
E-Mail: ★ email@email.com.br  
Ano/Semestre Inicial ★ 2012 - 1 (Ex.: 2006-2)  
Login: nome.sobrenome  
Senha: ★   
Confirmar Senha: ★

 • Inscrição cadastrado(a) com sucesso!

2

## Passo 2: Checar a saída

DADOS DO DISCENTE

Matrícula: ★ 2012000482

Nível: ★ GRADUAÇÃO

Mas será que pra outra entrada  
vai funcionar???



2

# Desafio do Teste

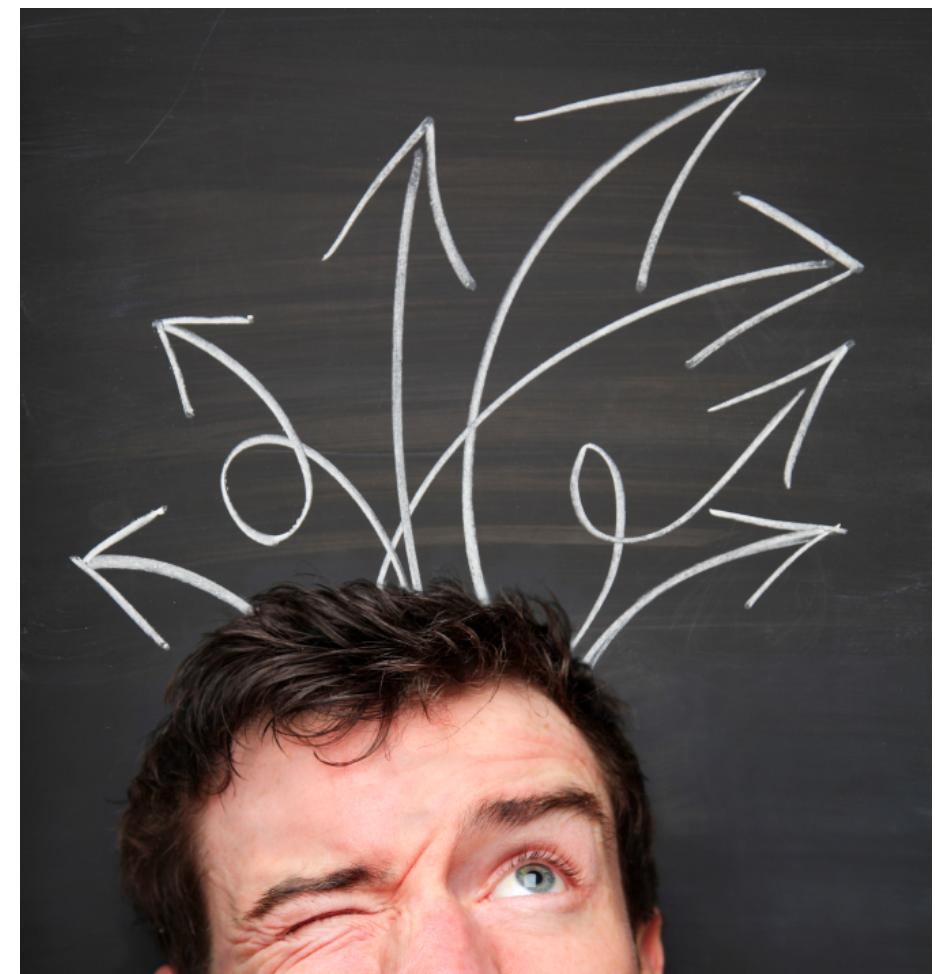
Escolher entradas  
com **maiores chances**  
de revelar falhas...



# Critérios de Teste

Critérios de Teste = critérios de escolha.

Guiam na escolha  
das entradas com  
**\*\*maiores chances\*\***  
de revelar falhas



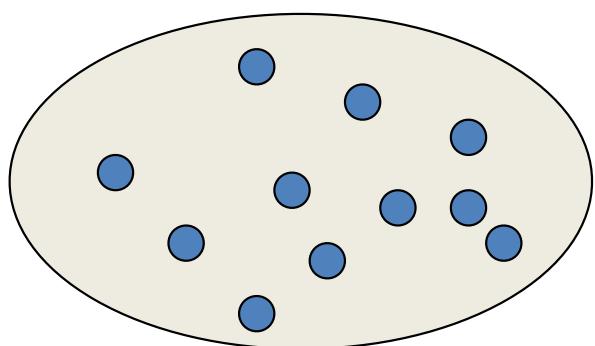
# Tipos de Critérios

Caixa  
Branca  
(Estrutural)

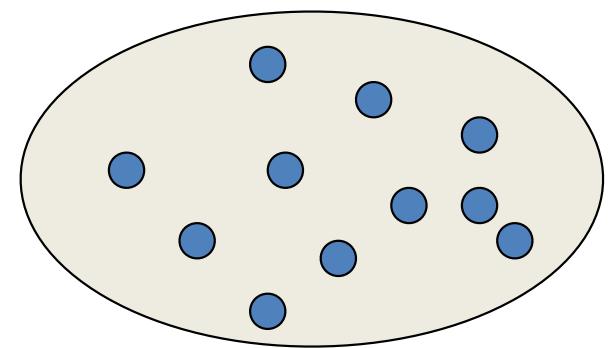
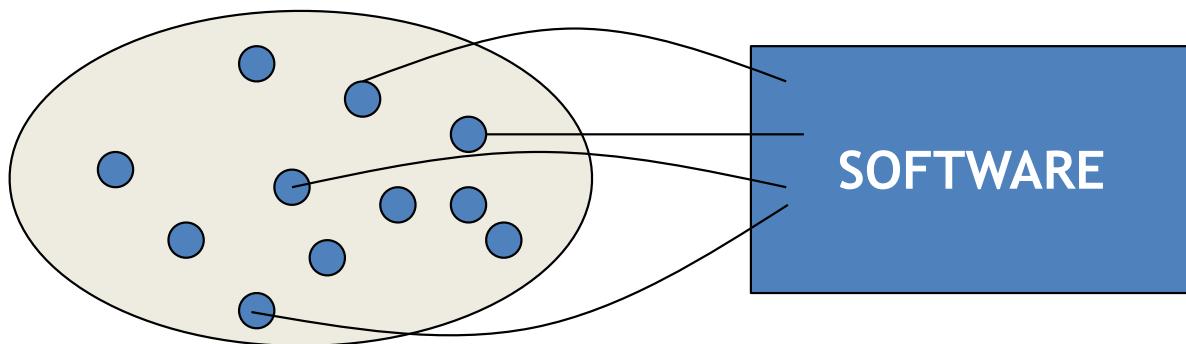
Caixa Preta  
(Funcional)

Baseado  
em  
Defeitos

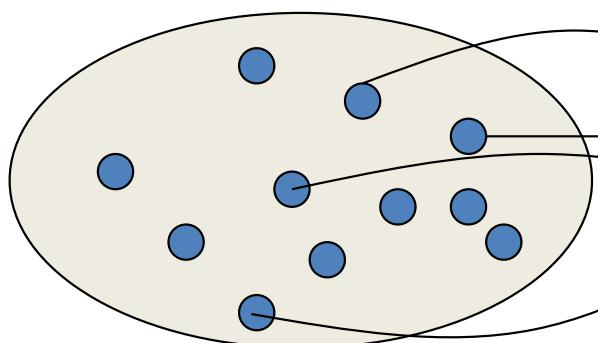
## **DOMÍNIO DE ENTRADA**



## DOMÍNIO DE ENTRADA



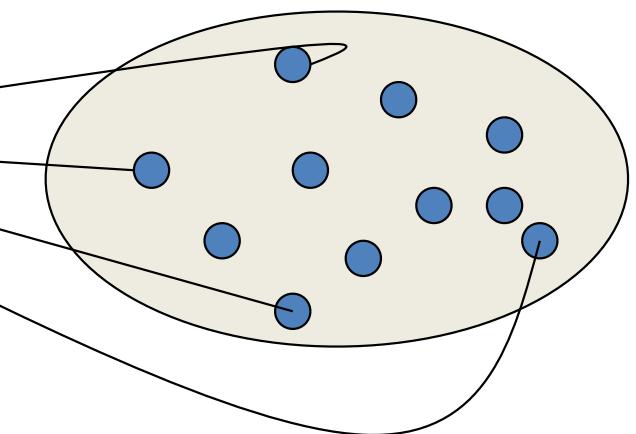
**DOMÍNIO DE ENTRADA**



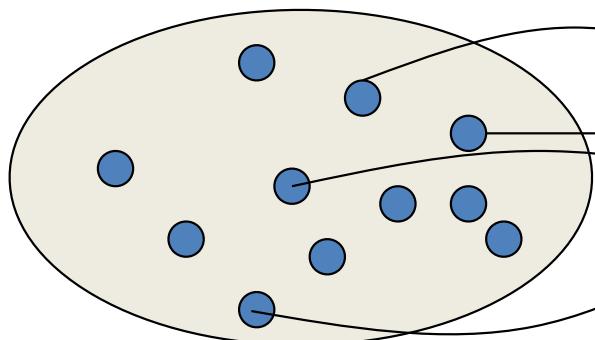
**SOFTWARE**



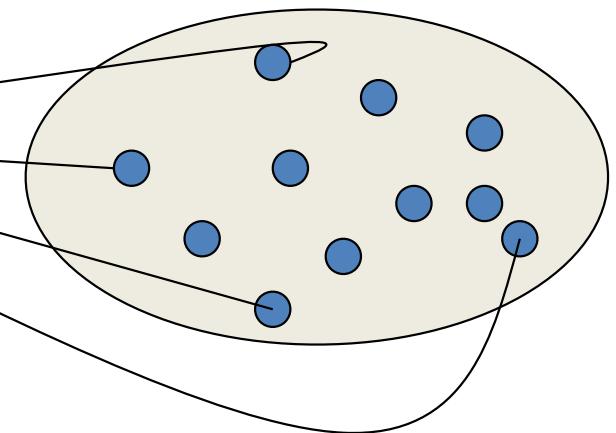
**DOMÍNIO DE SAÍDA**



**DOMÍNIO DE ENTRADA**



**DOMÍNIO DE SAÍDA**



**Oráculo**



E então....

O que é testar?

Testar consiste em  
executar o software a  
procura de falhas...

Para testar é necessário ter uma  
atitude **DESTRUTIVA!!**

Quanto mais falhas encontrarmos  
melhor!



Esta atitude é **CONSTRUTIVA** a longo prazo.

Bugs encontrados são corrigidos!



# Discutimos

Teste != Qualidade

O objetivo dos testes é revelar falhas.

A incompletude dos Testes.



# Conceitos Básicos de Teste de Software

Profa. Roberta Coelho

Departamento de Informática e Matemática  
Aplicada - DIMAp

# Teste

- Conjunto de atividades e técnicas relacionadas com a procura de defeitos em um programa

# Terminologia: Falta e Falha

- IEEE STD. 982.2-1988 (<http://standards.ieee.org/>)
  - Fault (falta, bug ou defeito): problema (especificação, projeto ou implementação)
  - Failure (falha): manifestação do problema. O que é observado pelo usuário ou (outro sistema)

# Quiz: Localize falta e falha

```
// pre condicao: v != null
public static void sort(int[] v) {
    for (int i = 0; i <= v.length; i++) {
        ...v[i] ...
    }
}
```

# Quiz: Localize falta e falha

```
// pre condicao: v != null
public static void sort(int[] v) {
    for (int i = 0; i <= v.length; i++) {
        ...v[i]...
    }
}
```

manifestação do defeito

defeito

# Terminologia: Defeito e Falha

1 defeito → ? falha(s)

# Terminologia: Defeito e Falha

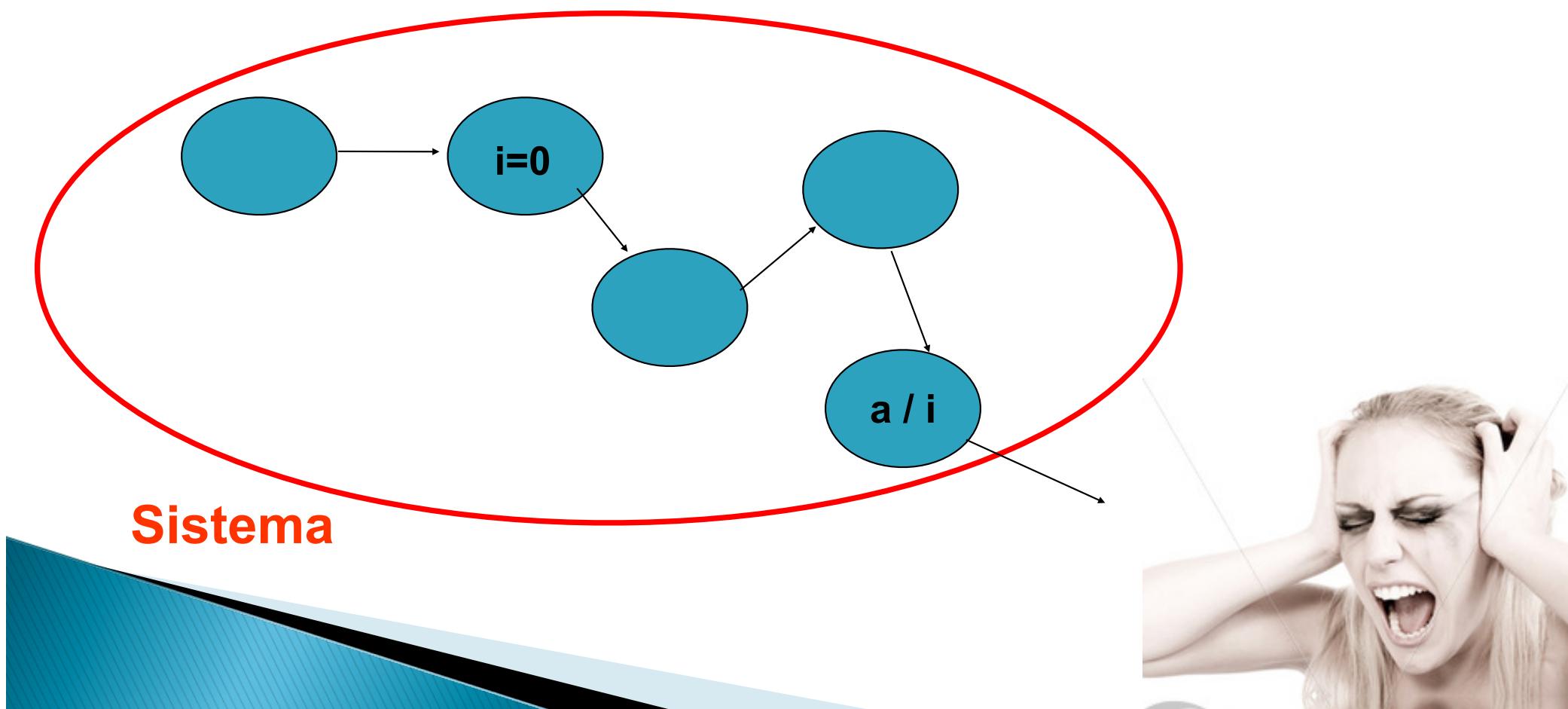
1 defeito → 0 ou mais falhas



Avizienis, A.; Laprie, J-C.; Randell, B.; "Dependability and Its Threats: A Taxonomy"; Proceedings of the IFIP 18th World Computer Congress: Building the Information Society, 2004; pags 91-120 - <http://rodin.cs.ncl.ac.uk/Publications/avizienis.pdf>

# Terminologia: Erro

- Erro: estado inconsistente (intermediário entre o defeito e sua manifestação).



# O que é um caso de testes?

- ▶ “Um conjunto de dados de entrada, condições de execução, e resultados esperados que tem o objetivo de exercitar o sistema para verificar se os requisitos foram atendidos” [IEEE, do178b]

# Um caso teste é composto:

- ▶ “Dados de entrada” = **entrada (dado + sequência)**
- ▶ “Condições de Execução” = **precondições**
- ▶ “Resultados Esperados” (oracle) = **resultado esperado (oráculo)**

# Caso de Teste (nível: Unidade)

```
1. public class ContaTest extends TestCase {  
2.     public void testSaqueBemSucedido () {  
3.         Conta c1 = new Conta ("123", 50.0);  
4.         c1.debitar (49.0);  
5.         assertEquals (c1.getSaldo (), 1.0);  
6.     }  
7. }
```

# Caso de Teste

## (nível: sistema)

**CT\_COMPRA\_001**

**Descrição:** Realização bem sucedida de uma venda, seguindo o fluxo principal do caso de uso Realizar Compra (sem múltiplos endereços)

**Requisito Associado:** RF007

**Tipo de Teste:** Funcionalidade

**Estágio de Teste:** Sistema

**Pré-Condições:**

- Cliente já cadastrado
- Carrinho de compra previamente cadastrado

**Pós-Condições:**

- Venda registrada na base
- Sistema financeiro notificado
- Carrinho de compra marcado como despachado na base

**Critérios de Sucesso:**

- Todas as pós-condições do caso de uso são atendidas

**Utilizando o Mind Map...**

<https://www.mindmeister.com/1155446999?t=a6Yhdx5QNk>



<https://mm.tt/1155446999?t=a6Yhdx5QNk>

# Tipos de Teste

Tipo do Teste	Descrição
Teste Funcional	<ul style="list-style-type: none"><li>- Testa a funcionalidade geral (regras de negócio).</li><li>- Condições válidas e inválidas.</li></ul>

# Tipos de Teste

Tipo do Teste	Descrição
Teste de desempenho	<ul style="list-style-type: none"><li>- Verifica o tempo de resposta e processamento para diferentes configurações:<ul style="list-style-type: none"><li>- número de usuários ou</li><li>- tamanho do BD.</li></ul></li><li>- Geralmente ocorre paralelamente ao teste de carga.</li></ul>

# Tipos de Teste

Tipo do Teste	Descrição
Teste de recuperação de falhas	<ul style="list-style-type: none"><li>- O software é forçado a falhar de diversas maneiras</li><li>- Verificamos procedimentos de recuperação.</li></ul>

# Tipos de Teste

Tipo do Teste	Descrição
Teste de Usabilidade	<ul style="list-style-type: none"><li>- Verifica:<ul style="list-style-type: none"><li>- navegação</li><li>- consistência</li><li>- aderência a padrões</li><li>- acessibilidade</li></ul></li></ul>

# Tipos de Teste

Tipo do Teste	Descrição
Teste de Segurança	<ul style="list-style-type: none"><li>- Verifica falhas de segurança:<ul style="list-style-type: none"><li>- sql injection</li><li>- script injection</li></ul></li></ul>

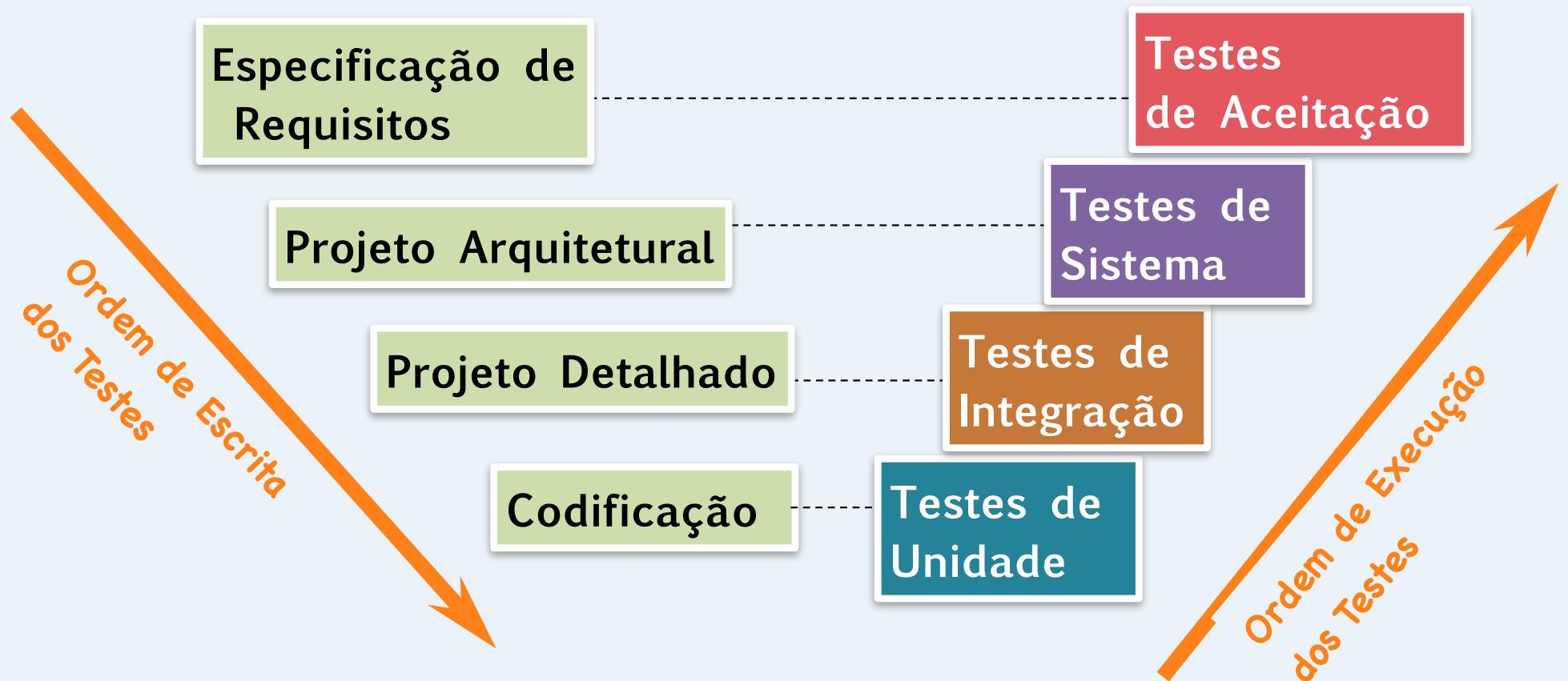


<https://mm.tt/1155446999?t=a6Yhdx5QNk>



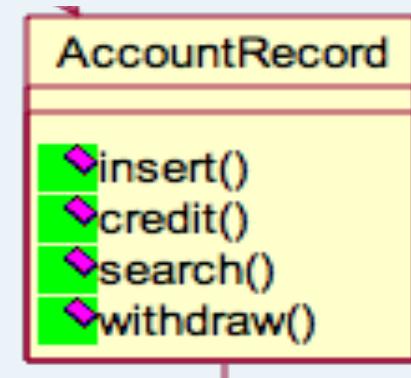
<https://mm.tt/1155446999?t=a6Yhdx5QNk>

# Níveis de Teste



# Teste de Unidade (Developer Testing)

**OBJETIVO:** Testar componentes individuais (ex: classes, métodos), durante a implementação.



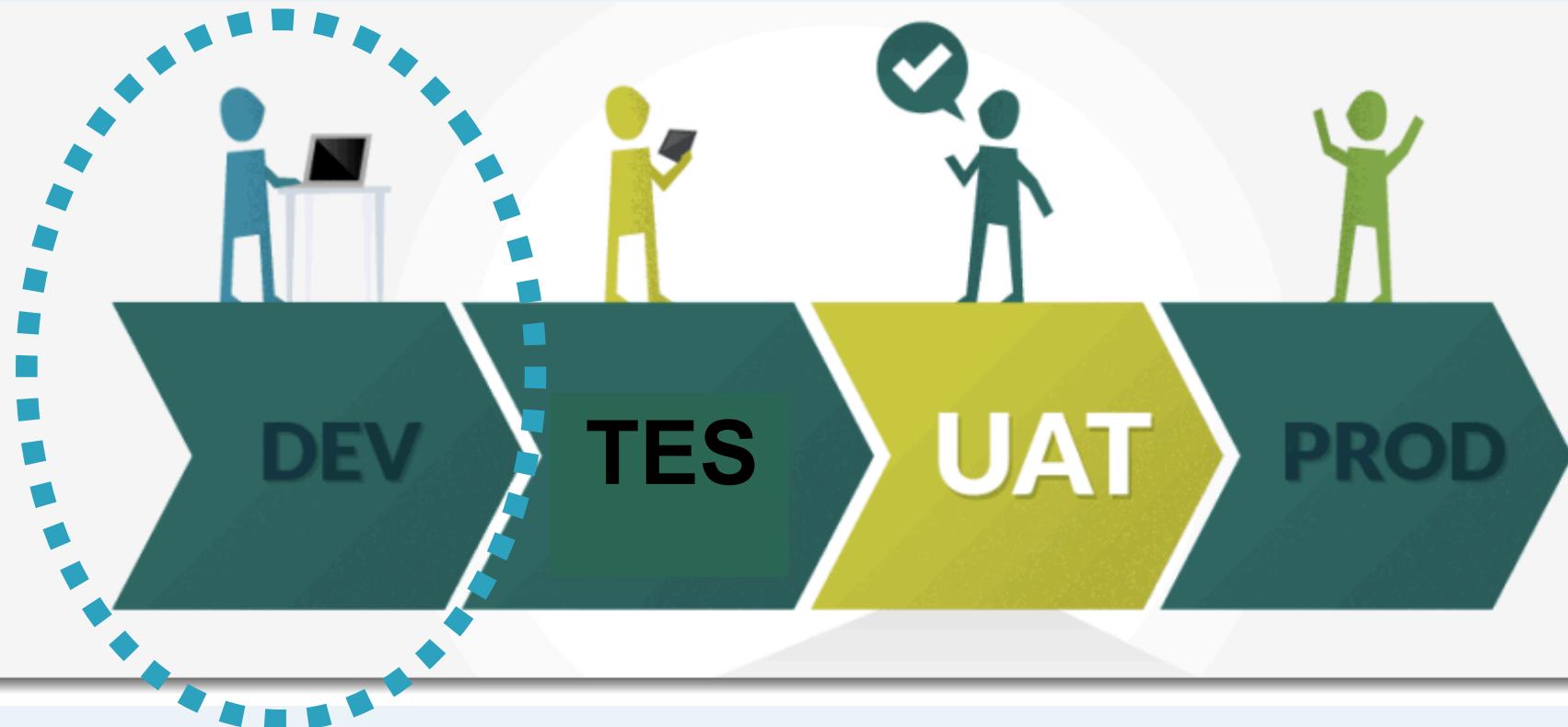
**QUE É TESTADO:** a interface, condições de Limite, tratamento de exceções dentro das fronteiras da unidade, ...

*Test Driven Development : testes antes!*

# Teste de Unidade (Developer Testing)



# Teste de Unidade (Developer Testing)



# Teste de Unidade

- **Vantagem:**

- Testa classes isoladamente
- Pode simular condições de erro
- Executa mais rápido que os testes anteriores

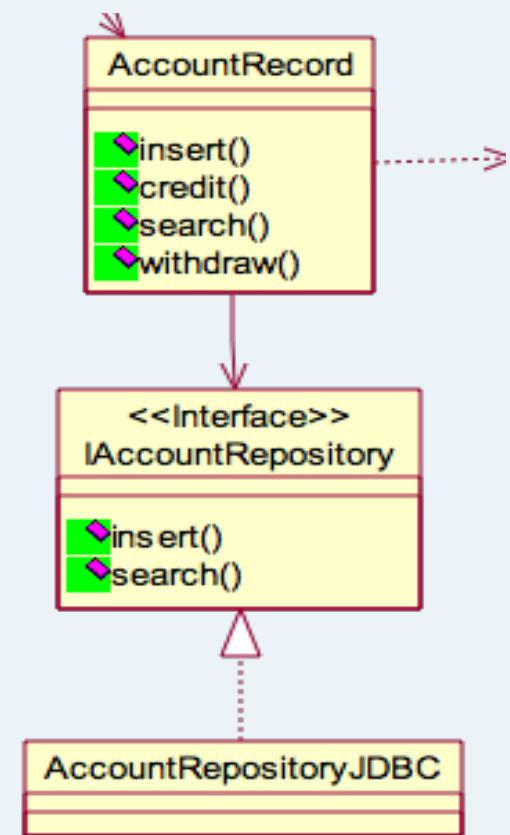
- **Desvantagem**

- Testa partes do sistema isoladamente
- Bugs de integração e sistema não são detectados

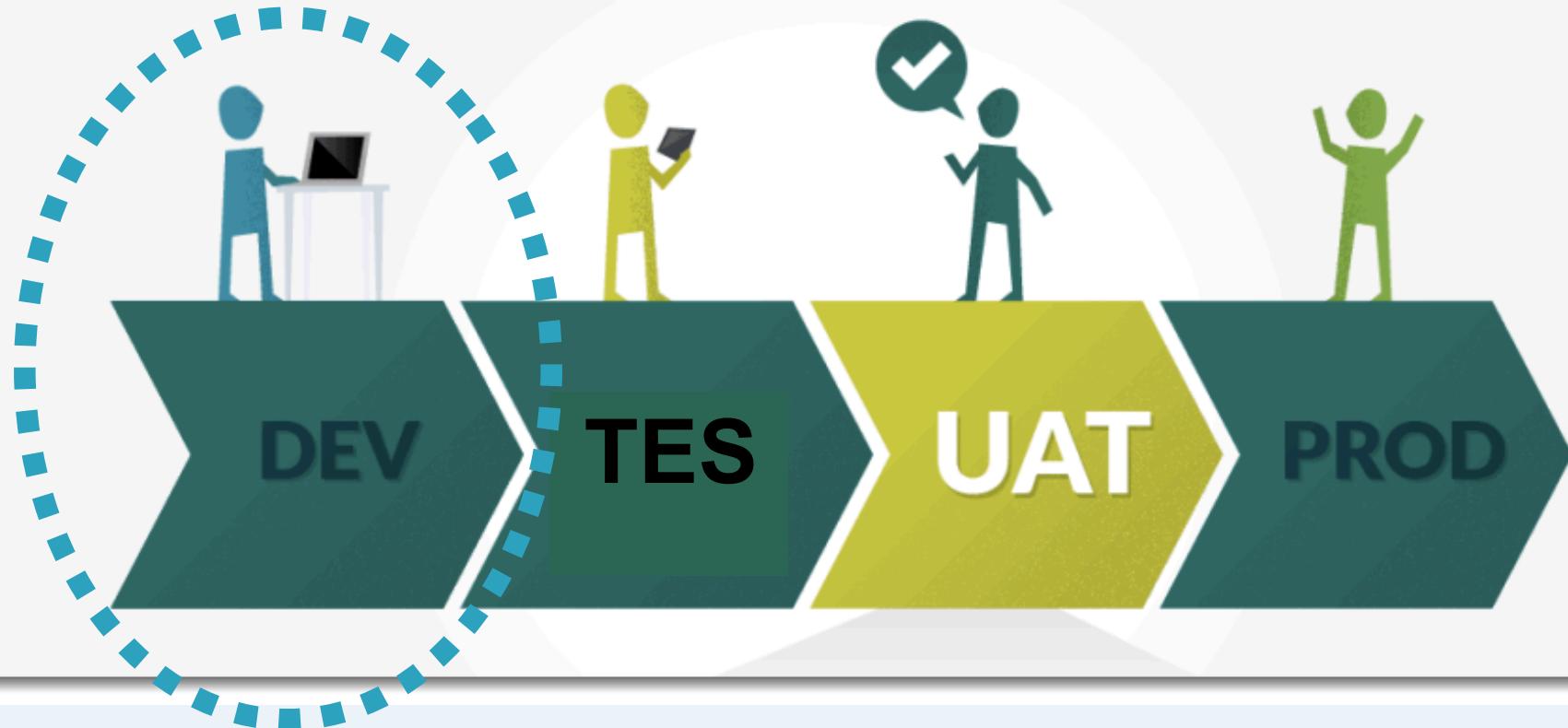
# Teste de Integração

**OBJETIVO:** Integrar conjuntos de módulos e testá-los (incremental).

**O QUE É TESTADO:** a interface entre os módulos.



# Teste de Integração



# Teste de Integração

- Vantagem:

- Ganhamos confiança nos subsistemas
- Testes focam na interação entre as classes

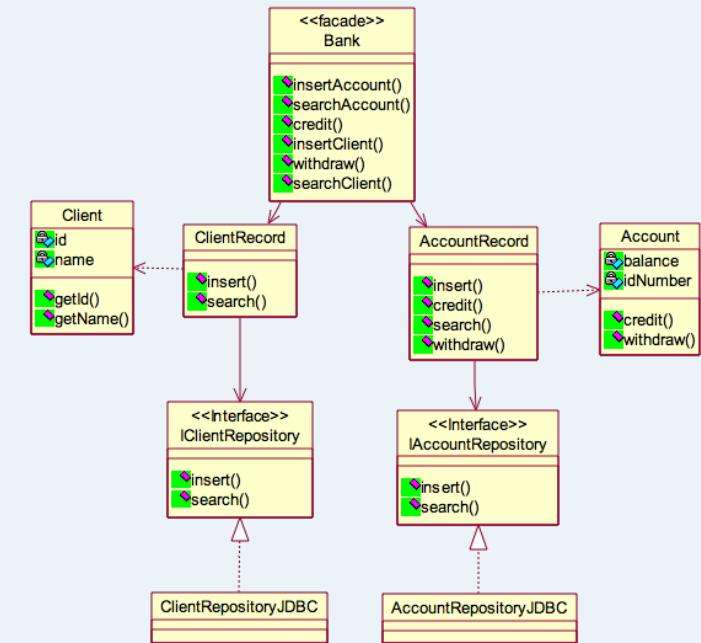
- Desvantagem

- É mais fácil de simular cenários de falha mas...
- ... ainda precisamos de um debugger para descobrir o que deu errado...

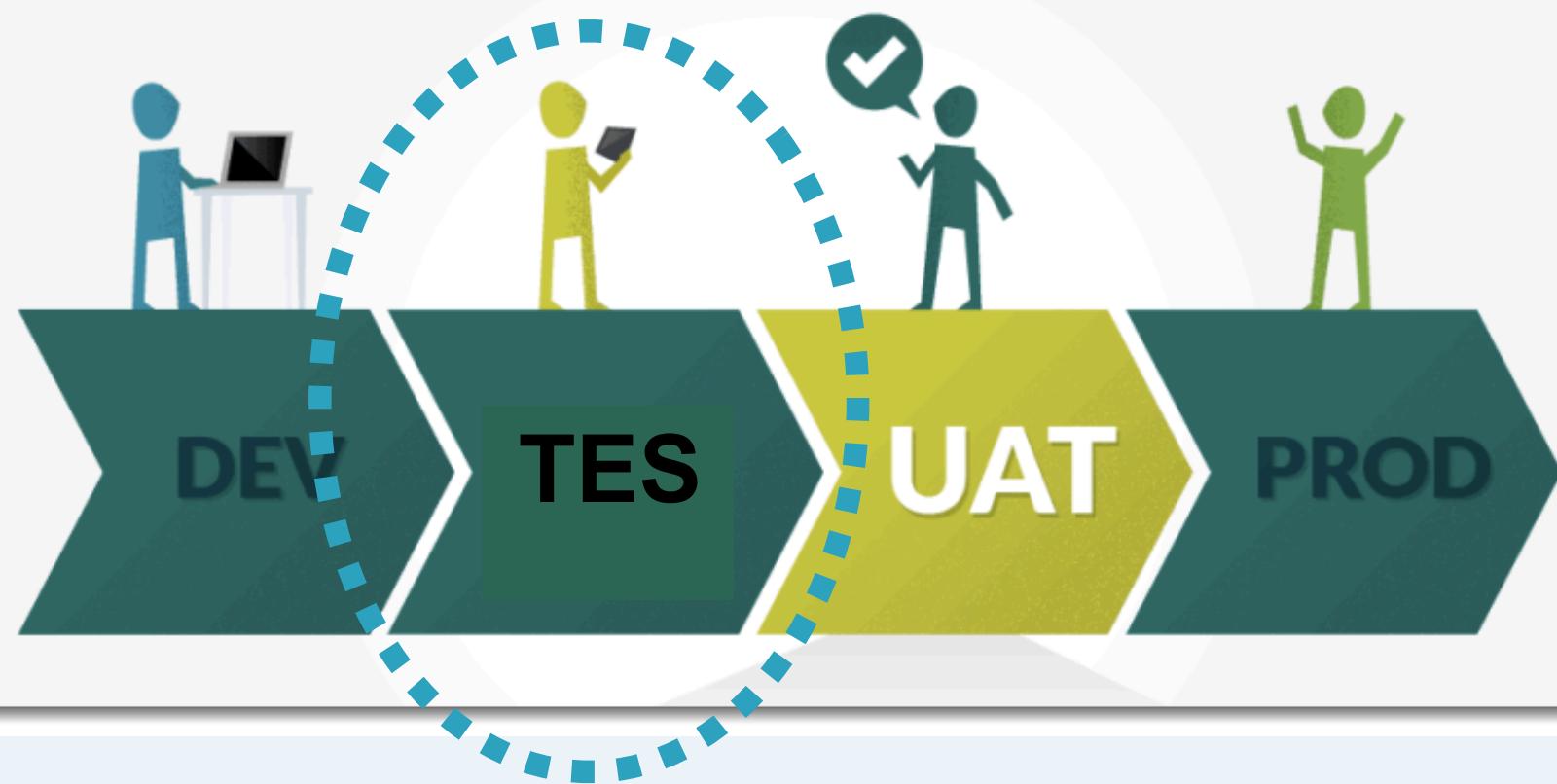
# Teste de Sistema

**OBJETIVO:** Verificar o funcionamento da aplicação como um todo em um ambiente operacional similar ao de produção.

**O QUE É TESTADO:** Verificar requisitos funcionais e não funcionais.



# Teste de Sistema



# Testes de Sistema

- **Vantagem:**
  - Ganhamos confiança nos “happy paths” do sistema
- **Desvantagem**
  - Custosos.
  - Necessário Debug para descobrir a causa da fala

# Teste de Aceitação

**OBJETIVO:** Verificar se o sistema atende as necessidades do Cliente.

**O QUE É TESTADO:** O sistema como um todo é testado pelo cliente ou um representante do cliente.

Tipos: Testes ALFA, Testes Beta, Crowd Testing

# Teste de Aceitação



# Testes de Aceitação

- **Vantagem:**

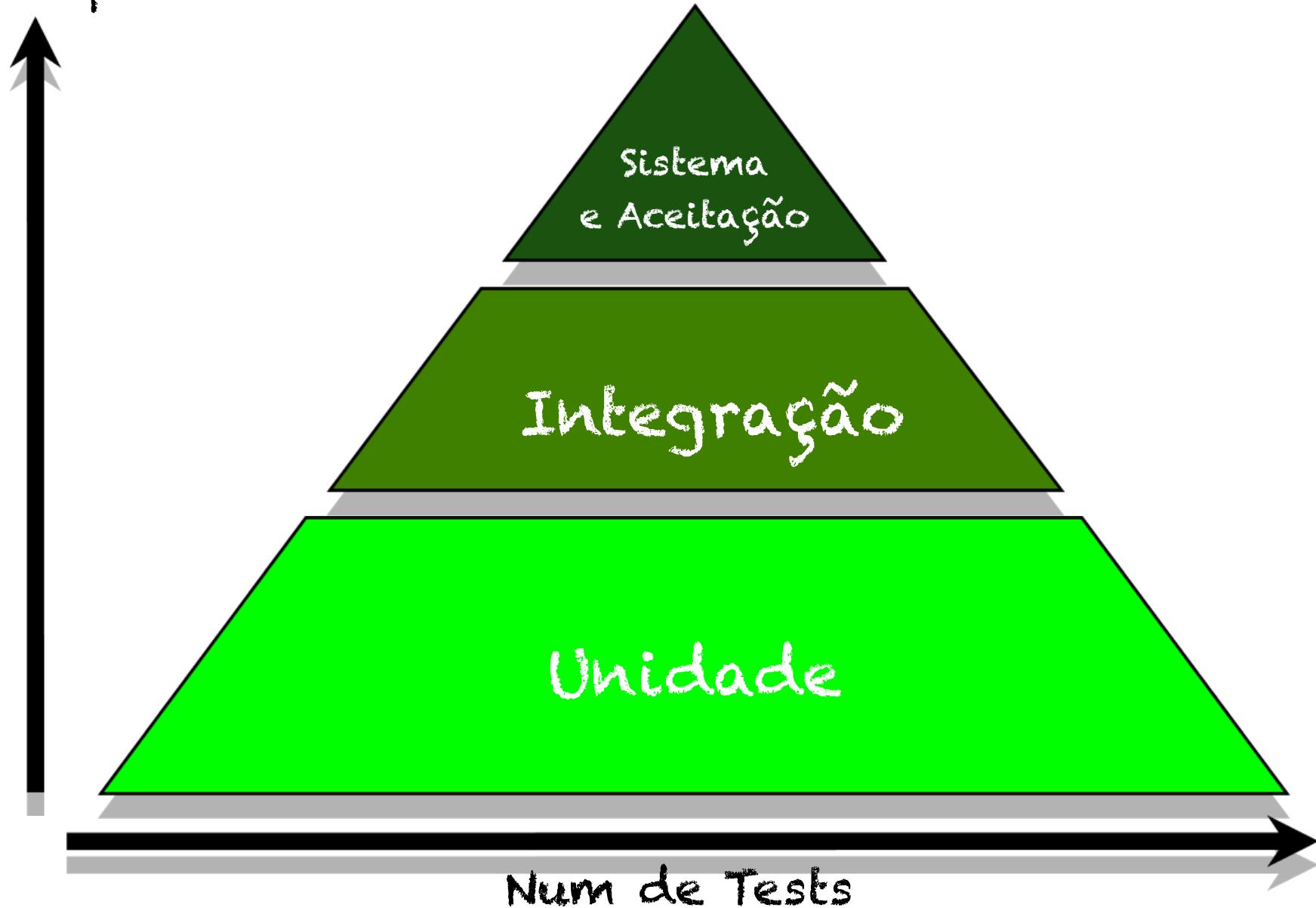
- Testes realizados pelo Cliente ou representante do cliente.
- Testar o sistema em um contexto real.
- Bugs realistas.

- **Desvantagem**

- Custosos.
- Necessário Debug para descobrir a causa da fala

# Testes: Desempenho x Quantidade

Tempo de exec.



# Importância de Liberar versões Alfa e Beta

Muitos bugs interessantes serão reportados – diferentes ambientes de execução, e cenários de uso reais.

Alternativa recente: Crowd Testing

Durante os teste falhas  
s o encontradas...



Falhas sobrarão após  
release



Precisamos ficar olho nas falhas  
pré e pós release

Falhas não  
são para se  
esconder  
baixo do  
tapete...



Falhas são uma  
preciosa fonte de  
informação...



Devemos analisar os tipos de bugs que escaparam e melhorar os próximos testes.

Quadro de tarefas

Abrir tarefa

Registrar Horário

Acompanhar Build...

**ROBERTA DE SOUZA COELHO**

Controle de Qualidade

Tarefas

Testes

Integração

Suporte

Relatórios

Administração

Interação

Adm. Pessoal

**GERÊNCIA DE FALHAS > EXCEÇÕES POR PERÍODO**

Esta operação permite gerar relatórios quantitativos das exceções que ocorreram no sistema durante um determinado período.

**FILTROS DA CONSULTA**

Período: \* 01/11/2013 a 03/11/2013

**FILTROS OPCIONAIS**

Sistema:  TODOS  
 Gerar Gráfico

AMBIENTES  
 COMUM  
 IPROJECT  
 PORTAIS  
 PORTAL INSTITUCIONAL  
 REDE SOCIAL  
 SIGAA  
 SIGADMIN  
 SIGED  
 SIGELEICAO  
 SIGEVENTO  
 SIGPP  
 SIGRH  
 SIPAC  
 TRANSPARENCIA ATIVA

**FILTROS OPCIONAIS** Sistema:

SIGRH

 Gerar Gráfico**Gerar Relatório** **Cancelar** **Listar Quantitativo de Exceções** **Listar Quantitativo de Causas de Exceções****LISTA DE ERROS POR SUBSISTEMA**

Sistema	Subsistema	Qtd. de Exceções	Qtd. de Ocorrências
SIGRH	DAP	1	1
SIGRH	SERVIDOR	1	4
<b>Totais:</b>		<b>2</b>	<b>5</b>



## EXCEÇÕES POR SISTEMA

Sistemas

SIGRH

SIGRH

<< Voltar

## Project Sistema Integrado de Gerência de Projetos

Click to go back, hold to see history

controle de Qualidade

### GERÊNCIA DE EXCEÇÕES > RELATÓRIO DE FALHAS (TREEMAP)

O gráfico abaixo é uma representação em Treemap das quantidades de exceções lançadas nos sistemas.

#### Instruções:

- Clique com o botão esquerdo do mouse no Sistema que deseja visualizar. Para voltar, utilize o botão direito.
- Nos subsistemas, as quantidades são mostradas no padrão *Nome (E/O)*, onde "E" é a quantidade de exceções e "O" é a quantidade de Ocorrências de exceções.
- Na escala de cores, quanto mais próximo do verde menor é o número de ocorrências. Analogamente, quanto mais próximo do vermelho maior o número de ocorrências.



Como os bugs dos  
sistemas que vcs trabalham  
são armazenados?

Quais os tipos de bugs  
mais frequentes??

To do: Elaborar o Plano de Testes para o sistema que vc está trabalhando no momento.

# Referencias

- Cap 6 do Livro The Art of Software Testing
- Artigo da IBM: [http://www.ibm.com/developerworks/br/local/rational/criacao\\_geracao\\_planos\\_testes\\_software/index.html](http://www.ibm.com/developerworks/br/local/rational/criacao_geracao_planos_testes_software/index.html)
- Growing Object-Oriented Software, Guided by Tests: Steve Freeman
- How we test Software at Microsoft
- How Google tests Software