

The background features a dark gray field with a grid of small, multi-colored dots in shades of teal, olive, and purple. In the top-left corner, there is a light green abstract shape. In the bottom-right corner, there is a light green abstract shape with several small, semi-transparent pink circles and triangles scattered near its edge.

# *ENGENHARIA DE SOFTWARE 2*

## *Teste de Software*

---

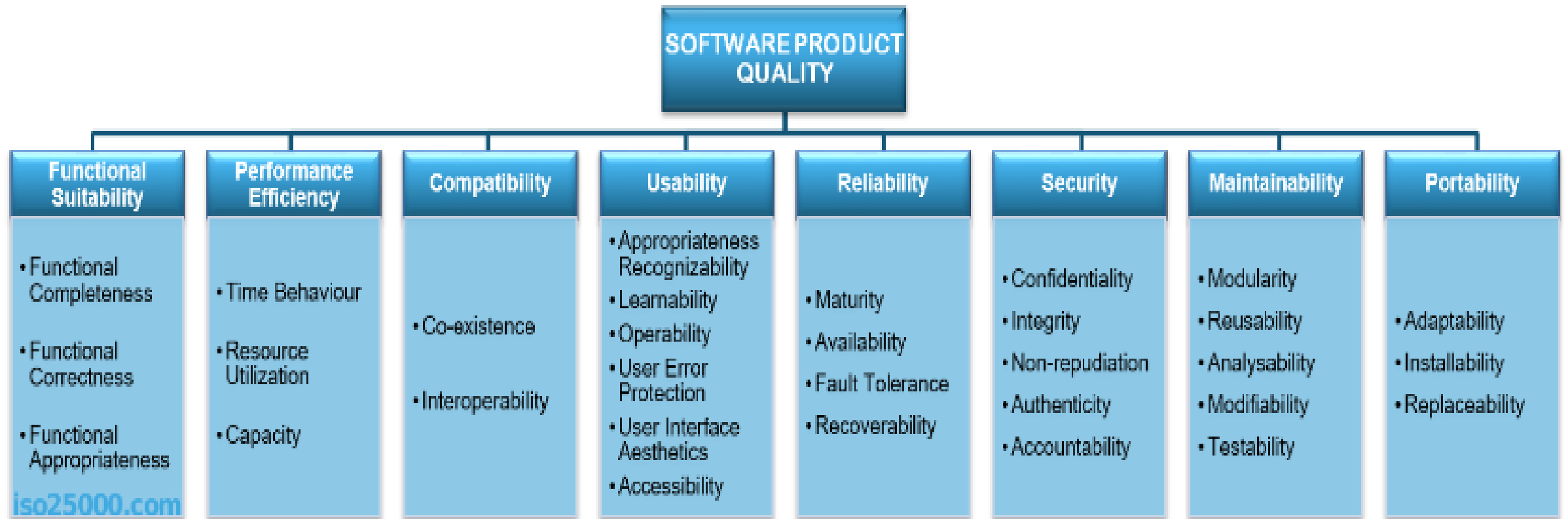
Profa Cristiane Palomar Mercado

# Por que testar?

- No mundo atual existe uma demanda não satisfeita por software de qualidade.
- 
- Para o desenvolvimento de software com qualidade, dentro de prazos e custos controlados e compatíveis com o mercado, é fundamental a melhoria dos processos da engenharia de software.

Modelos: SW-CMM, ISO/IEC 12207, ISO/IEC 15504 e CMMI

# Qualidade do produto definido na ISO/IEC 25010



# *O que é Teste de Software?*

---

Mostrar que um programa faz o que é proposto a fazer e para descobrir os defeitos do programa antes do uso (Sommerville)

Intenção de descobrir erros e defeitos em um Sistema. (Myers, 2004)

# *Erro*

Ação humana que produz um resultado incorreto.



# *Defeito*

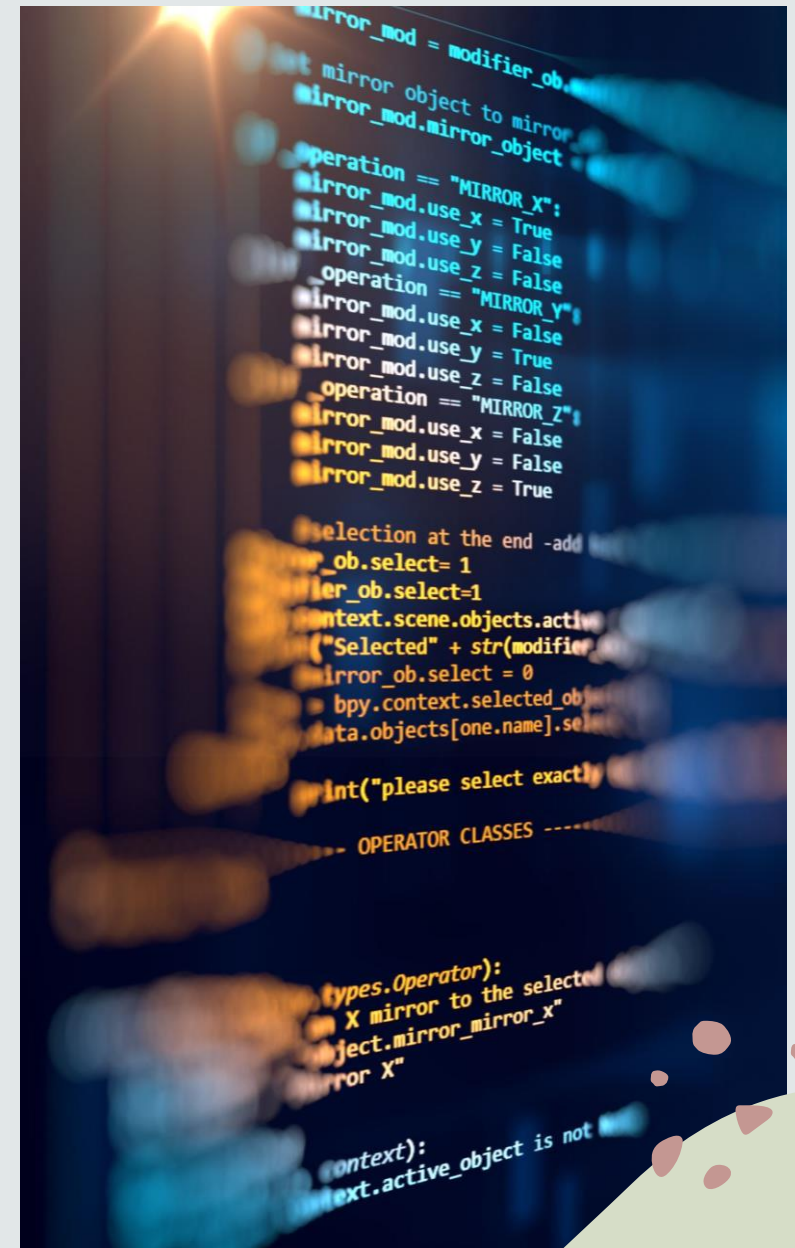
---

Defeito é uma imperfeição de um produto. O defeito faz parte do produto e, em geral, refere-se a algo que está implementado no código de maneira incorreta.

Considere, por exemplo, o código a seguir:

```
a = input ();
```

```
c = b/a; se a = 0 então erro de divisão por zero.
```



# *Falha*

---



Falha é o resultado errado provocado por um defeito ou condição inesperada.

Uma divisão por zero : programa pode funcionar durante anos, sem que jamais ocorra uma falha: tudo dependerá dos valores retornados pela rotina `input()`.

Os defeitos podem existir, mas nem sempre são visíveis.

Falhas também podem ocorrer por fatores externos ao programa, como corrupção de bases de dados ou invasões de memória por outros programas.



# Erro – Defeito - Falha



Uma pessoa  
comete um  
**erro**...

...que cria um  
**defeito** no  
software...



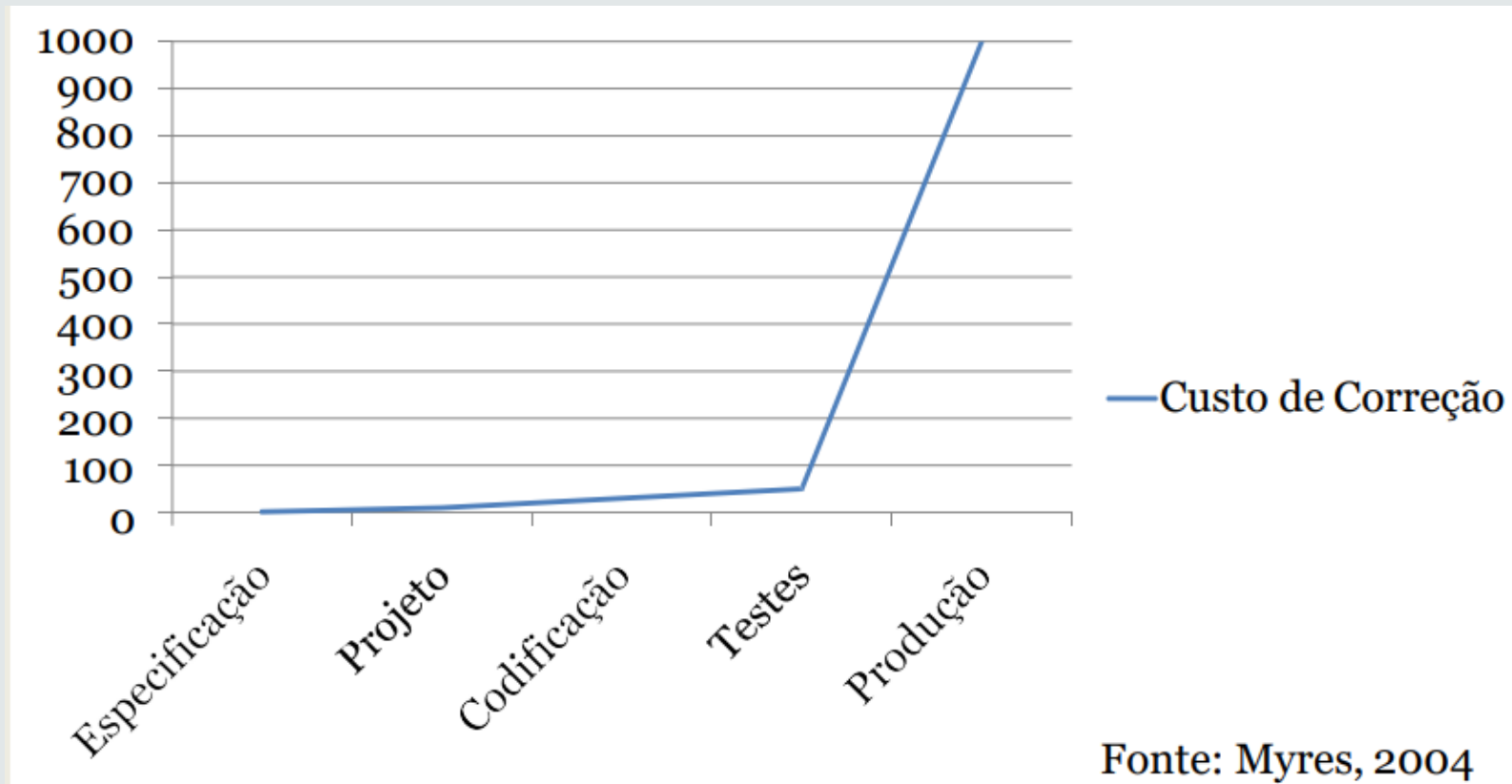
...que pode  
causar uma  
**falha** na  
operação.





# *Custos de Teste de Software*

O custo da correção de um defeito tende a ser cada vez maior quanto mais tarde ele for descoberto. [Myres, 2004]



# *A dificuldade em testar software é caracterizada por alguns pontos importantes*

---

---

o teste de software é um processo caro;

---

existe uma falta de conhecimento sobre a relação custo/benefício do teste;

---

há falta de profissionais especializados na área de teste;

---

existem dificuldades em implantar um processo de teste;

---

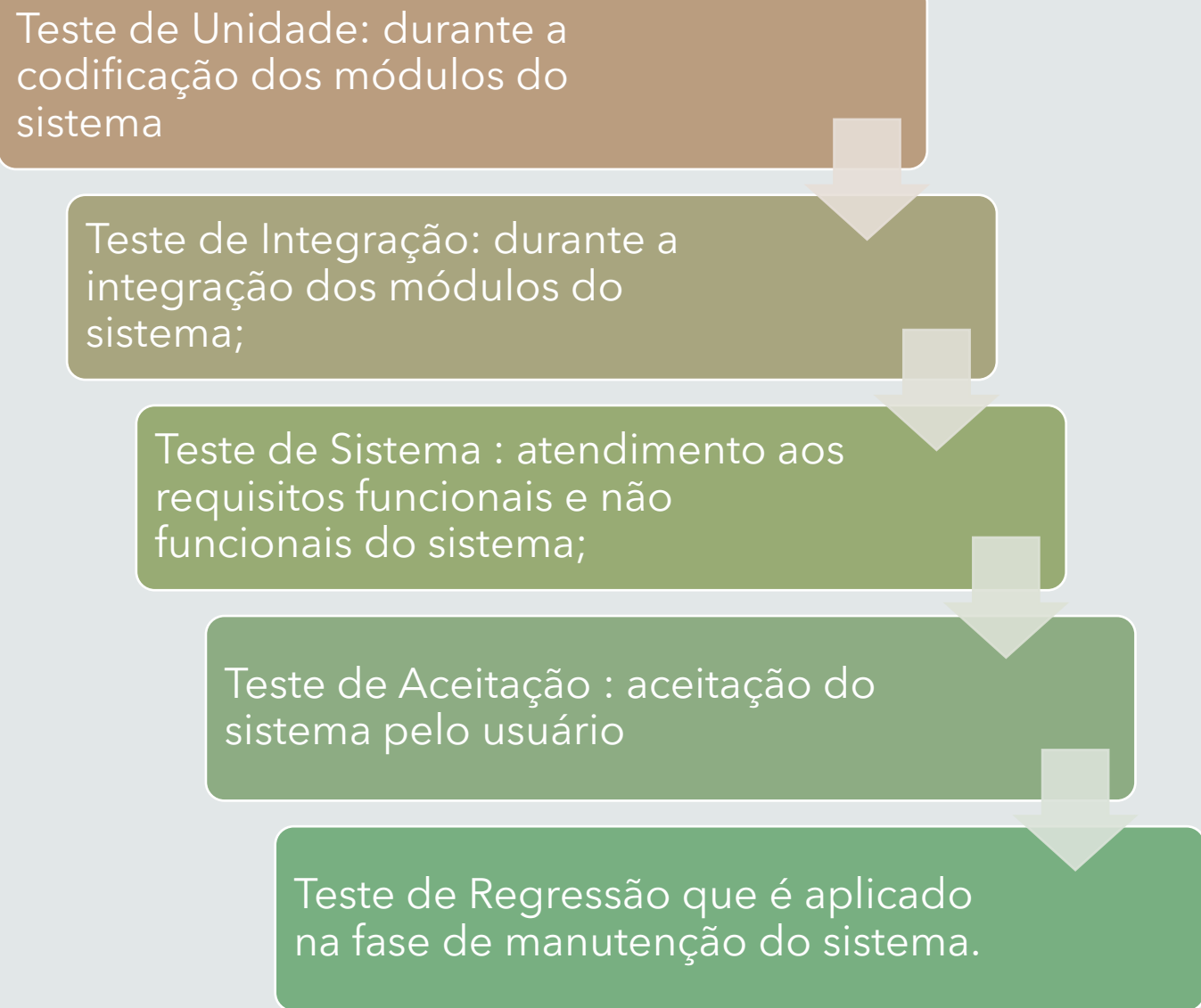
há o desconhecimento de técnicas de teste e planejamento adequadas;

---

preocupação com a atividade de teste somente na fase final do projeto.

*Nível de Teste:  
depende da fase do  
desenvolvimento  
do software:*

Teste de Unidade: durante a codificação dos módulos do sistema



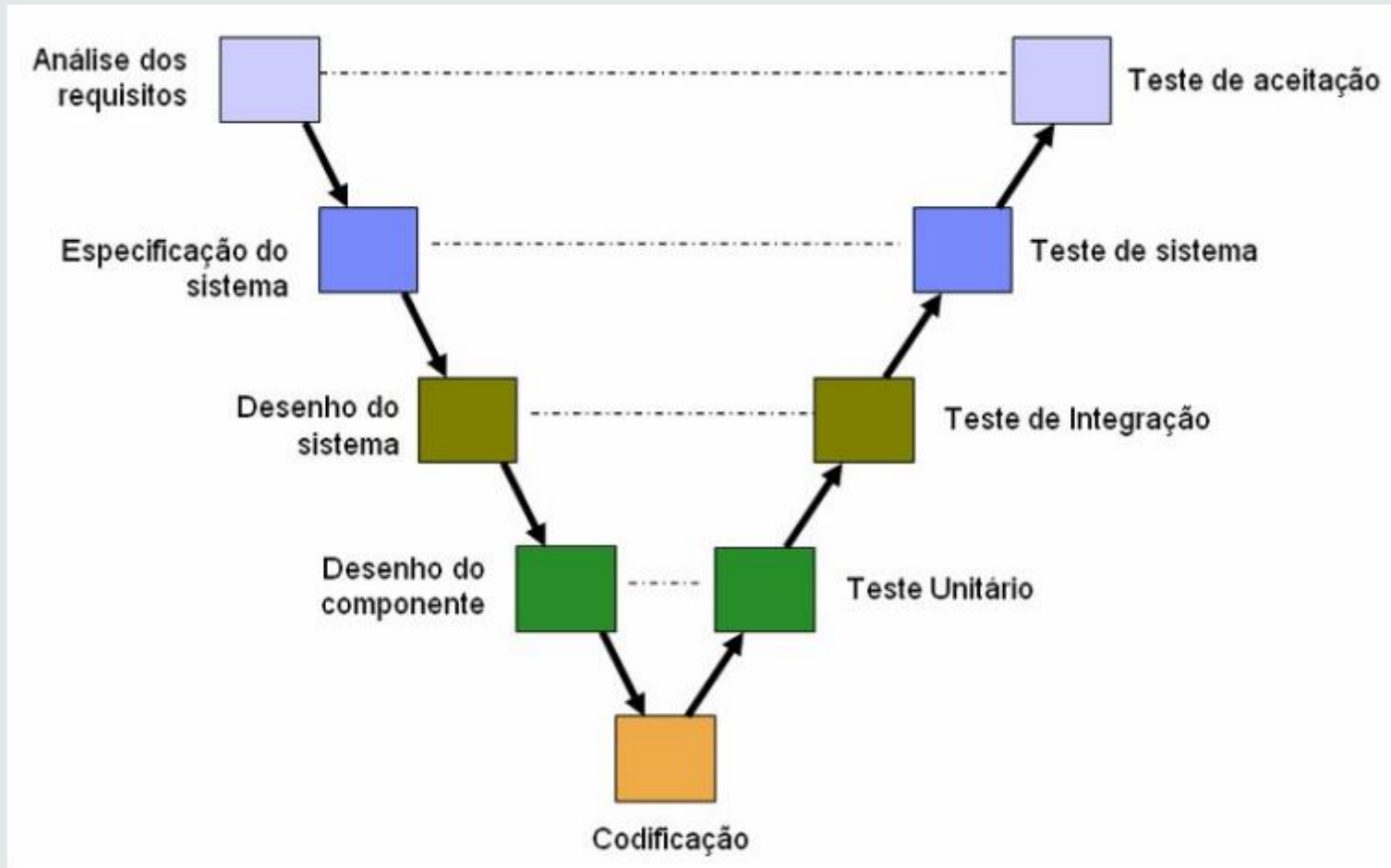
Teste de Integração: durante a integração dos módulos do sistema;

Teste de Sistema : atendimento aos requisitos funcionais e não funcionais do sistema;

Teste de Aceitação : aceitação do sistema pelo usuário

Teste de Regressão que é aplicado na fase de manutenção do sistema.

# Níveis de Teste



## *Técnicas de teste:*



- Teste Estrutural (Caixa branca) - Identifica defeitos nas estruturas internas do software, através de situações que exercitem adequadamente todas as estruturas utilizadas na codificação;
- Teste Funcional (Caixa Preta) - Garante que os requisitos do software que foi construído sejam plenamente atendidos.

## *Elementos de teste:*



Normalmente os elementos de um software que podem ser testados são:

- as linhas de comando;
- As funções implementadas;
- as variáveis definidas no software;
- os *loops* existentes no software;
- Todos os ramos de uma decisão;
- e os requisitos do software.

## *Tipos de teste:*

Teste de Funcionalidade: Tem como função avaliar as funções do sistema observando se estão funcionando corretamente

Teste de Interface: Experimenta mecanismos de interação e valida aspectos estéticos da interface de usuário.;

Teste de Desempenho: testar o desempenho em tempo de execução do software dentro do contexto de um sistema integrado;



## *Tipos de teste:*

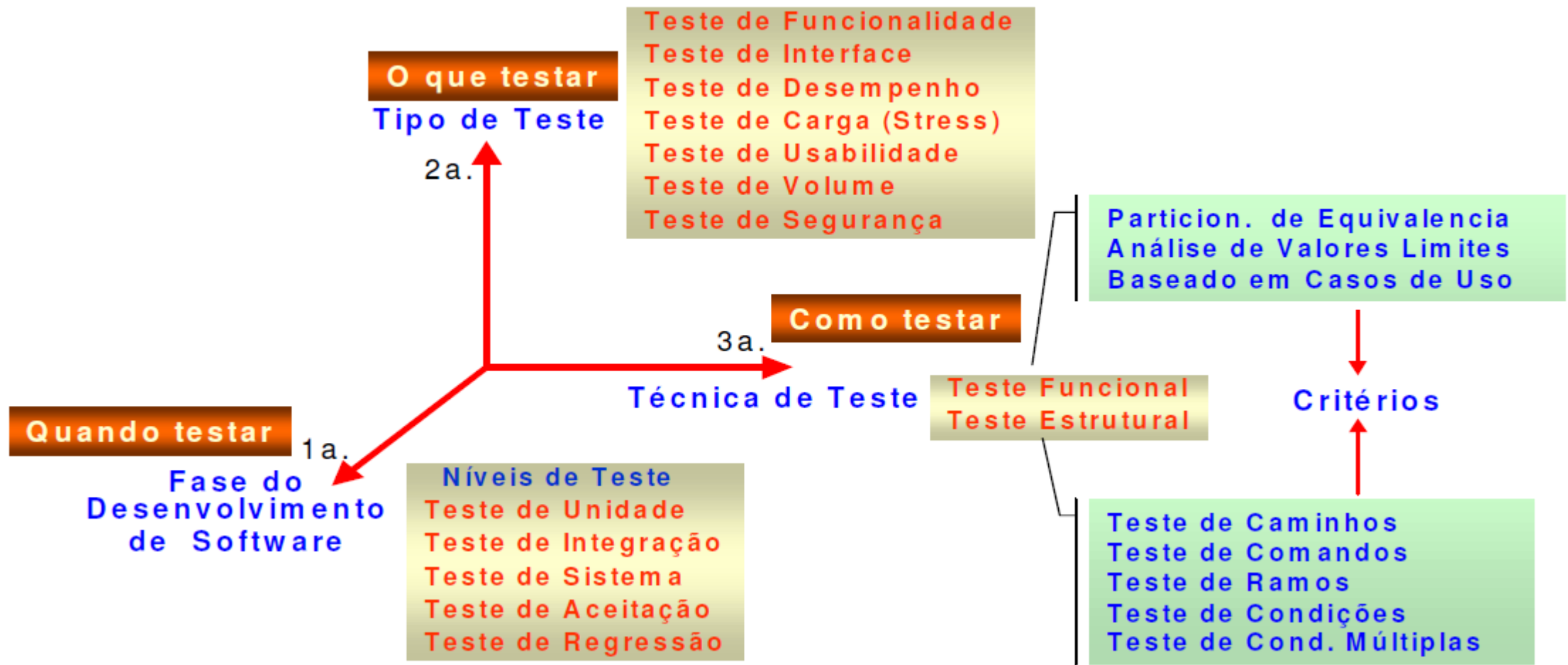
Teste de Carga (*Stress*): determinar como a aplicação e seu ambiente do lado do servidor responderá a várias condições de carga.: número de usuários concorrentes , número de transações on-line por usuários por unidade de tempo, carga de dados processados pelo servidor por transação

Teste de Usabilidade: são projetados para determinar o grau com o qual a interface da aplicação facilita a vida do usuário.;

Teste de Volume: realizado para testar um aplicativo de **software** com uma determinada quantidade de dados;

Teste de Segurança: tenta verificar se os mecanismos de proteção incorporados ao sistema vão de fato protegê-lo contra acesso indevido

# Relação entre níveis, tipos e técnicas de teste




# Testando Aplicações Para Web (Pressman)



---

O teste de WebApp é um conjunto de atividades relacionadas com um único objetivo: descobrir erros no conteúdo, na função, na usabilidade, na navegabilidade, no desempenho, na capacidade e na segurança da WebApp.



- Conteúdo;
- Função;
- Estrutura;
- Utilização;
- Navegabilidade;
- Desempenho;
- Compatibilidade;
- Interoperabilidade;
- Capacidade;
- Segurança

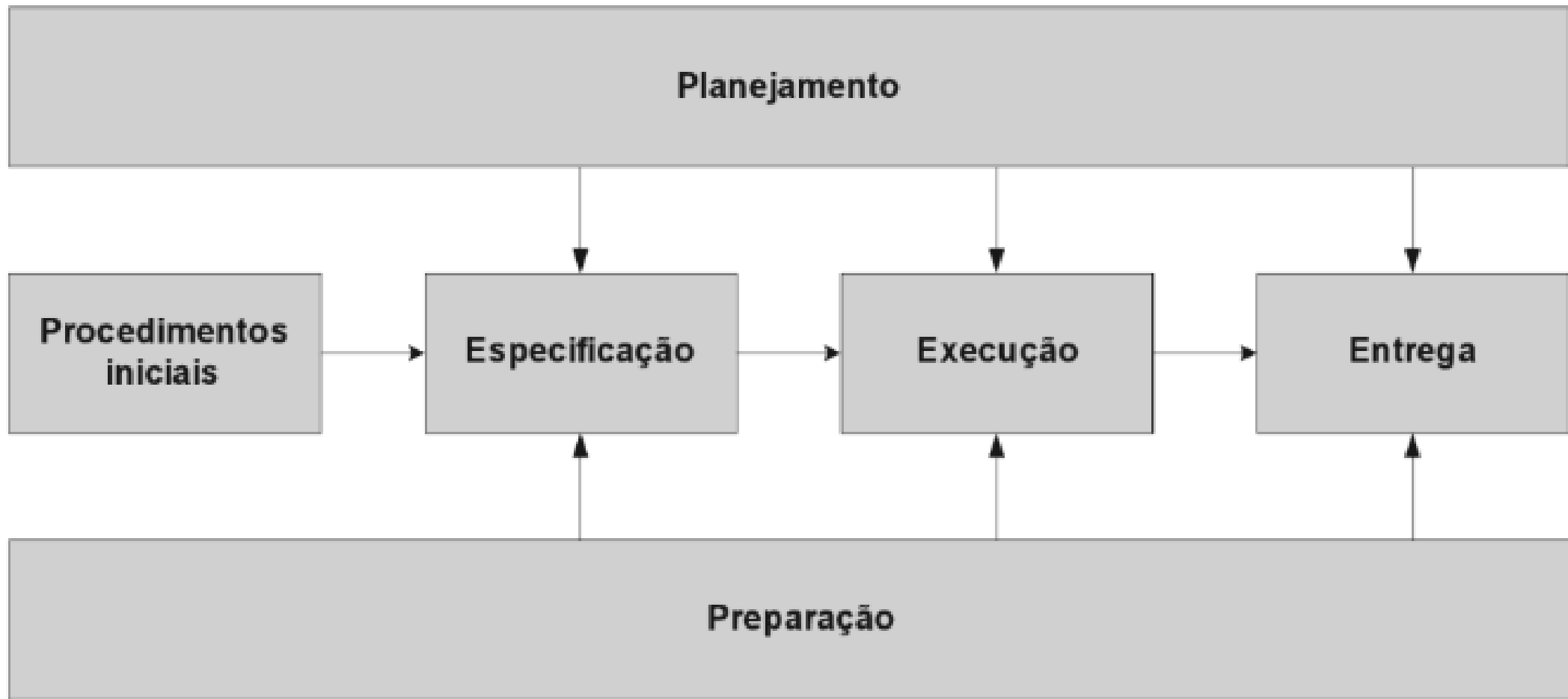
# *Fases de Processo de Testes*

---

O início de um processo de testes começa com o recebimentos dos requisitos de software.



# *Fases de Processo de Testes*



# *Plano de Testes*

---

A finalidade do Plano de Teste é definir e comunicar a intenção do esforço de teste em determinada programação.

Ele direciona, orienta e restringe o esforço de teste, priorizando os produtos liberados úteis e necessários.







- Caso de teste
  - Especificação de uma entrada para o programa e a correspondente saída esperada
    - Entrada: conjunto de dados necessários para uma execução do programa
    - Saída esperada: resultado de uma execução do programa
      - Oráculo
  - Um bom caso de teste tem alta probabilidade de revelar um erro ainda não descoberto

# Caso de Teste - Exemplo

Usar o mesmo script de teste para testar diversas configurações.

---

Testar um script de login em três navegadores diferentes, como o Firefox, o Internet Explorer e o Safari:

Três cenários de teste:

- Caso de teste 1: Firefox e script de teste de login
- Caso de teste 2: Internet Explorer e script de teste de login
- Caso de teste 3: Safari e script de teste de login

## *Processo de teste para WebApp (Pressman)*

---

*O processo de teste  
abrange sete  
diferentes tipos de  
teste.*

Teste de conteúdo : descobrir erros de sintaxe e semântica que afetam a exatidão do conteúdo ou a maneira pela qual ele é apresentado ao usuário final.

Teste de interface: exercita os mecanismos de interação. A finalidade é descobrir erros que resultam de mecanismos de interação mal-implementados ou de omissões, inconsistências ou ambiguidades em semânticas de interface.

Teste de navegação: Os mecanismos de navegação são verificados para assegurar que quaisquer erros que impedem a realização de um caso de uso sejam identificados e corrigidos.

# *Processo de teste para WebApp (Pressman)*

---

Teste de componente: experimenta as unidades de conteúdo e funcional da WebApp.

Teste de configuração tenta descobrir erros e/ou problemas de compatibilidade específicos de um ambiente particular de cliente ou servidor.

Teste de segurança : finalidade é encontrar brechas de segurança.

Teste de desempenho abrange uma série de testes projetados para

Avaliar o tempo de resposta e a confiabilidade da WebApp quando aumentam as demandas de recursos do servidor.

# Exemplos

---

<https://www.gov.br/economia/pt-br/assuntos/empresas-estatais-federais/central-de-conteudo/kits-governanca-ti/kit-1/processo-de-desenvolvimento-de-software/template-plano-de-testes.docx>

[https://www.trt1.jus.br/documents/22179/1274471/TestTest953347\\_15.PDF/3e001a6b-7253-9cc1-8991-2acb9d6c5c8f](https://www.trt1.jus.br/documents/22179/1274471/TestTest953347_15.PDF/3e001a6b-7253-9cc1-8991-2acb9d6c5c8f)

Acesso em 29/10/2023

# Referências

SOMMERVILLE, Ian. Engenharia de software. 10. ed. São Paulo: Pearson, 2019.

CRESPO, Adalberto Nobiato; DA SILVA, Odair Jacinto; BORGES, Carlos Alberto; SALVIANO, Clênio Figueiredo; DE TEIVE E ARGOLLO JUNIOR, Miguel; JINO, Mário. **Uma Metodologia para Teste de Software no Contexto da Melhoria de Processo**. In: SIMPÓSIO BRASILEIRO DE QUALIDADE DE SOFTWARE (SBQS), 3. , 2004, Brasília. **Anais** [...]. Porto Alegre: Sociedade Brasileira de Computação, 2004 . p. 204-218. DOI: <https://doi.org/10.5753/sbqs.2004.16194>.

<https://docente.ifsc.edu.br/joao.augusto/MaterialDidatico/2018-1/An%C3%A1lise%20e%20Projeto%20de%20Sistemas/Testes%20de%20Software>

<https://iso25000.com/index.php/en/iso-25000-standards/iso-25010>

[Myres, 2004] Myres , G. F. "The Art of Software Testing". Ed. John Wiley & Sons, Inc. New Jersey, 2004

Costa, Edjandir Corrêa Costa. Qualidade de Software.. Instituto Federal de Santa Catarina. Aula

Koscianski, André; Soares, Michel Qualidade de Software. Aprenda as metodologias e técnicas mais modernas para o desenvolvimento de software. 2ª edição. Novatec.

Pressman, Roger S. Engenharia de software [recurso eletrônico] : uma abordagem profissional / Roger S. Pressman ; 7. ed. - Dados eletrônicos. - Porto Alegre : AMGH, 2011.

<https://www.ibm.com/docs/pt-br/engineering-lifecycle-management-suite/lifecycle-management>