

**FIAP** GRADUAÇÃO

**DISCIPLINA: PROJETO DE SISTEMAS APLICADO AS MELHORES PRÁTICAS EM  
QUALIDADE DE SOFTWARE E GOVERNANÇA DE TI**

**AULA:**

**13 – TESTE DE SOFTWARE**

**PROFESSOR:**

**RENATO JARDIM PARDUCCI**

**PROFRENATO.PARDUCCI@FIAP.COM.BR**

**Renato Parducci - YouTube**

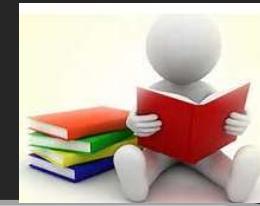
## AGENDA DA AULA

- ✓ CMMI nível 3 - VER/VAL
- ✓ MPS.br nível D - VER/VAL
- ✓ Níveis, tipos e técnicas de teste
- ✓ Teste de software - modelo V

PRÁTICAS E NÍVEL 3 –TS,  
VER/VAL

Estratégia de Teste de Software

## ESTUDO DE CASO SIMULADO



Dilan continua a insistir que acredita ter na má qualidade dos testes, o seu maior problema na condução de projetos com os clientes.

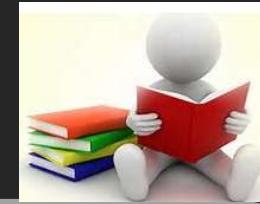
Ele percebe que em muitos projetos, a equipe perde mais horas para testar do que levou para desenvolver o software, o que torna o preço do projeto e o prazo, inviáveis para clientes ou muito caros.

Ele acredita que um trabalho de desenvolvimento de software excelente é o que produz programas sem bug da primeira vez, sem necessidade de testar; mas reconhece que isso é utopia.

O que Dilan pede é que os testes sejam feitos com mais inteligência e otimização de esforços e custos, trazendo resultados melhores que os atuais.

Considerando que a equipe hoje testa de forma Adhoc (tentativa e erro), Consuelo propôs que sejam aplicados modelos de planejamento e execução de testes referência de mercado e que se use muita automação para reduzir o esforço de executar e registrar o resultado de testes manualmente.

## ESTUDO DE CASO SIMULADO



Para começar essa caminhada na melhoria dos testes, Consuelo vai iniciar com um treinamento sobre como deve ser organizado o processo de liberação de um software, acompanhado dos testes necessários.

## O TESTE E A GOVERNANÇA E QUALIDADE DO SOFTWARE

Quando alguém fala em **QUALIDADE** de algo, pensamos logo que aquele algo foi muito bem **testado** e esperamos que **inexistam defeitos** no seu funcionamento que comprometam os resultados do uso e a segurança ao utilizar o objeto no dia a dia.

## O TESTE E A GOVERNANÇA E QUALIDADE DO SOFTWARE

A Qualidade e a Governança sobre planos, ações e resultados, são alcançados por dois princípios fundamentais:



Processo adequado para planejar, administrar e realizar um projeto e produto

Avaliação de resultados alcançados e da satisfação de quem recebe o produto final

## O TESTE E A GOVERNANÇA E QUALIDADE DO SOFTWARE

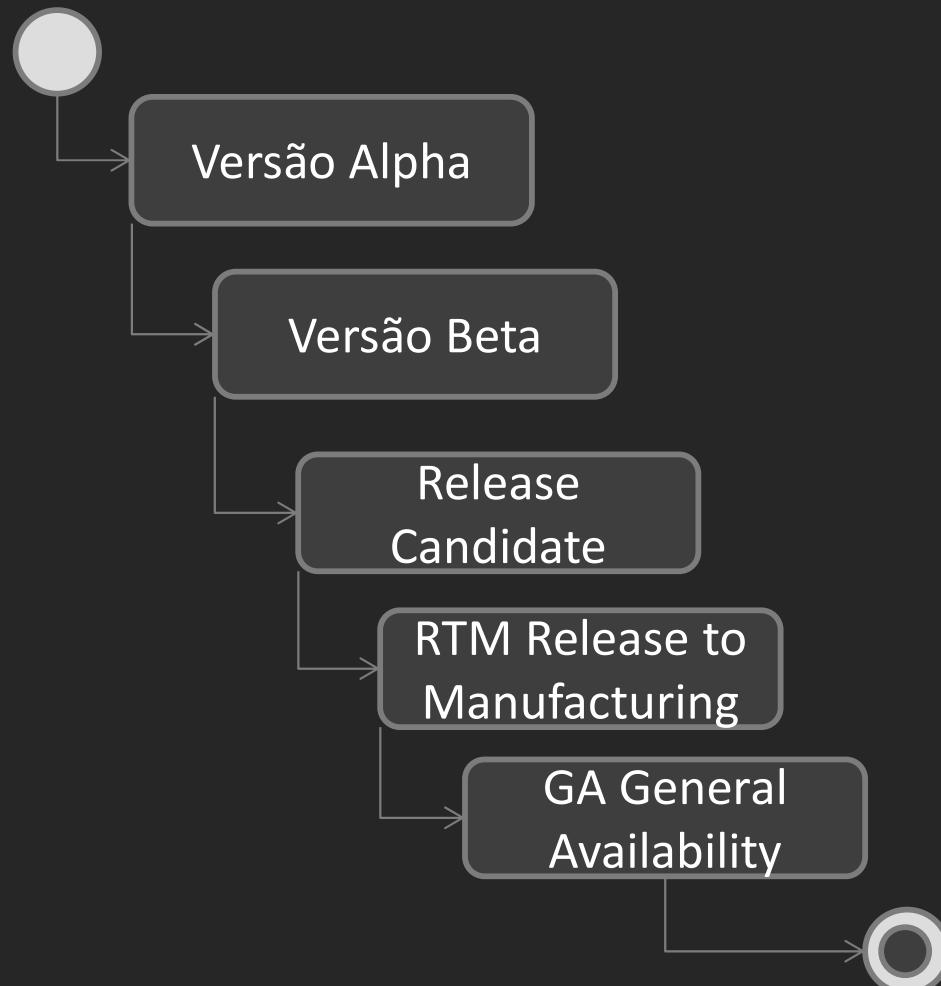
**Exploramos processos de planejamento e controle da produção de software em nossa disciplina.**

Nas disciplinas de desenvolvimento de software (programação), você aprende os processos de construção de produto.

É hora de aprender a avaliar os resultados durante e após o projeto do produto! **HORA DE APRENDER A PLANEJAR E FAZER TESTES!**

## GERENCIAMENTO DA LIBERAÇÃO DO SOFTWARE

A indústria de software segue um roteiro na hora de liberar um produto novo ou uma nova versão de um produto em uso:

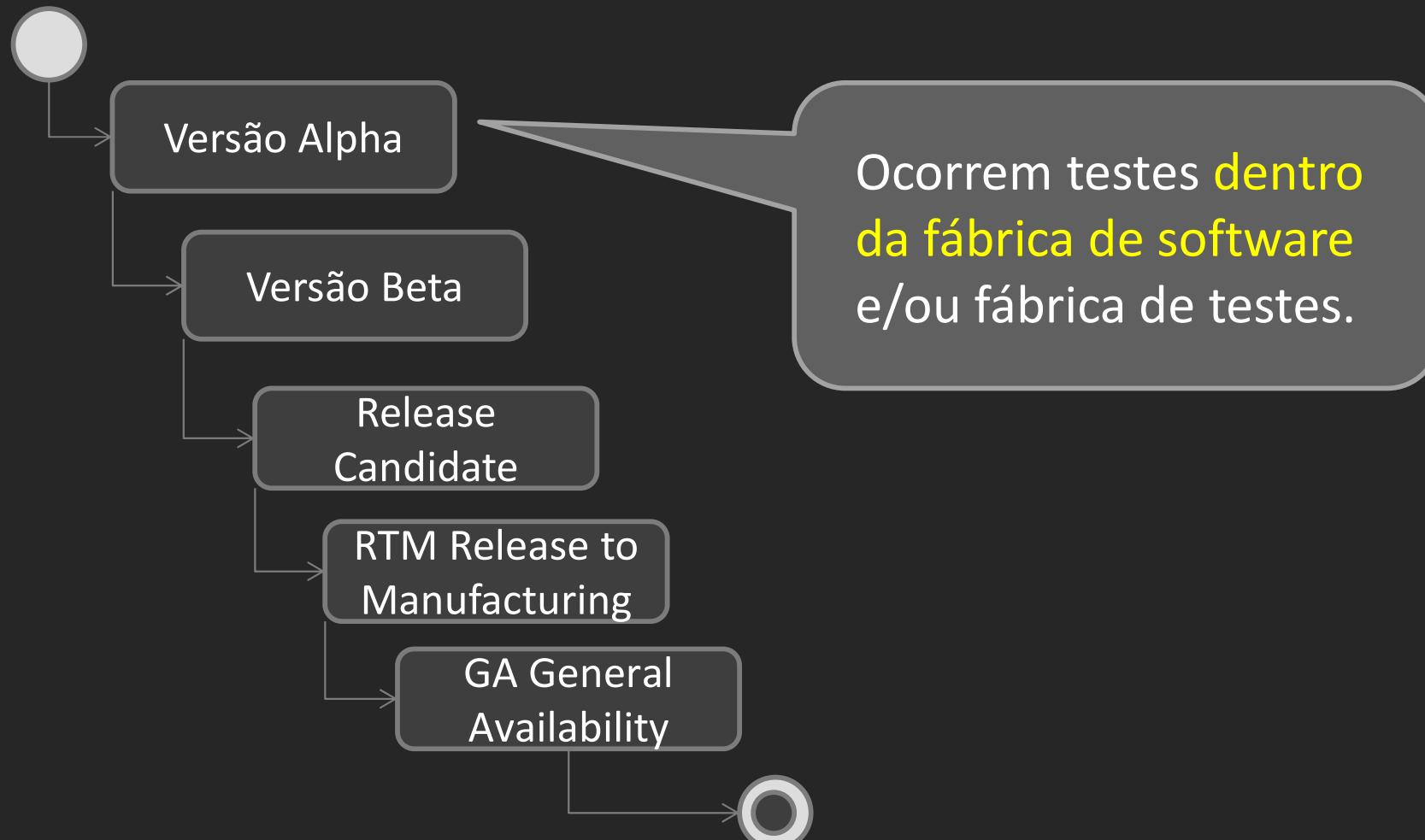


Modelo seguido por empresas como:



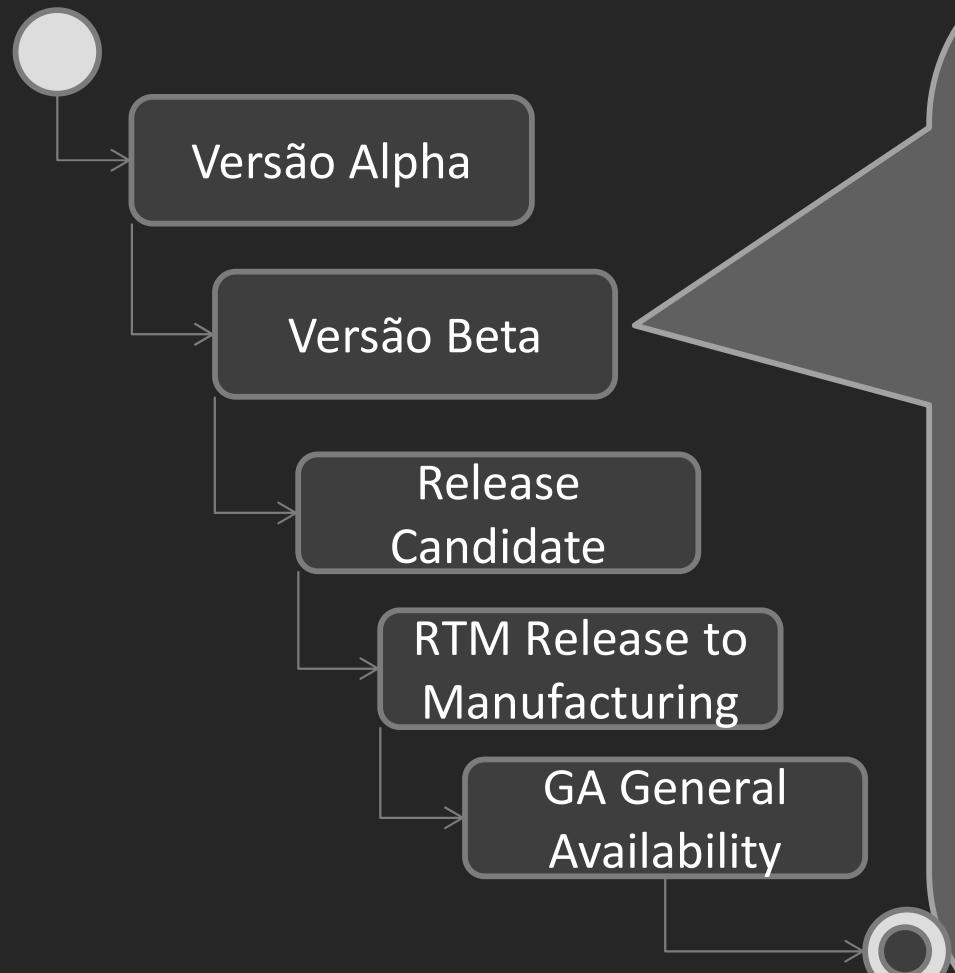
## GERENCIAMENTO DA LIBERAÇÃO DO SOFTWARE

Conhecendo as versões do software durante um programa de liberação...



## GERENCIAMENTO DA LIBERAÇÃO DO SOFTWARE

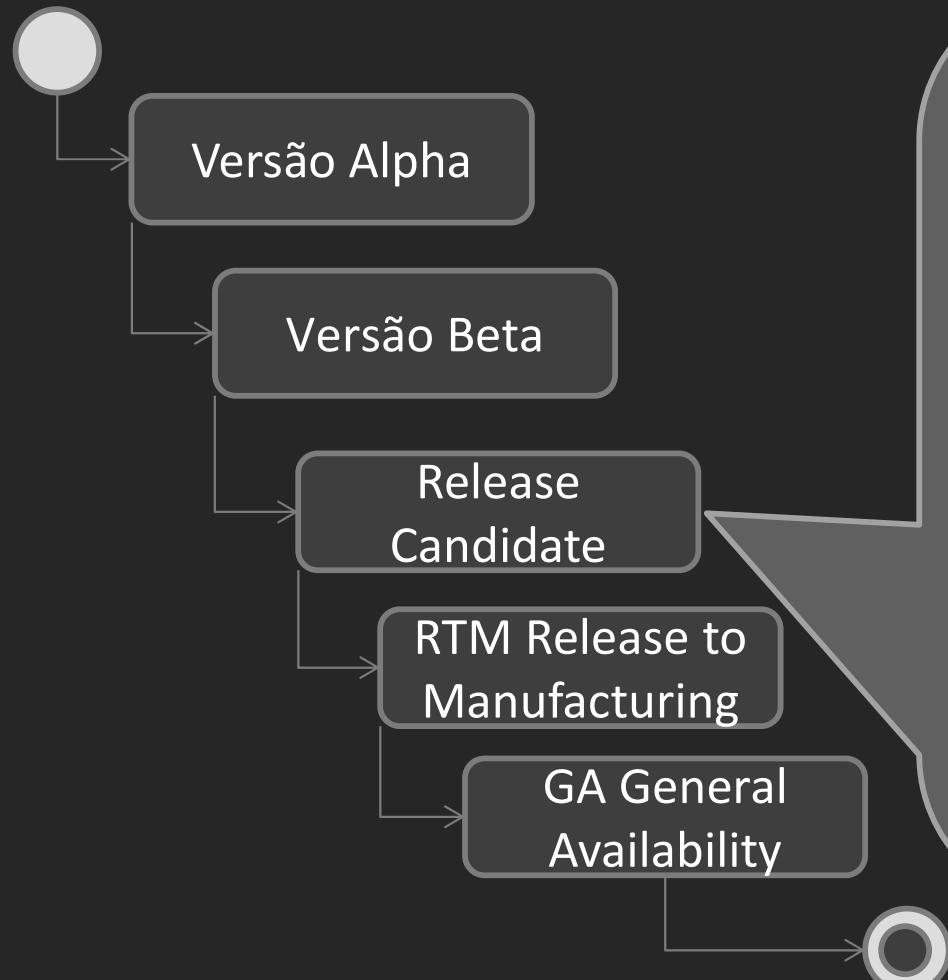
Conhecendo as versões do software durante um programa de liberação...



O foco na versão beta é reduzir impactos aos usuários, geralmente incorporando testes de utilização pelo cliente/consumidor do produto. Essa é a **primeira divulgação fora dos limites da organização** que o desenvolve. Os usuários de versões beta são chamados **beta testers** ("testadores beta"). Geralmente este grupo compõe-se de **consumidores selecionados que aceitam participar dos testes**, podendo ganhar algum incentivo por essa colaboração.

## GERENCIAMENTO DA LIBERAÇÃO DO SOFTWARE

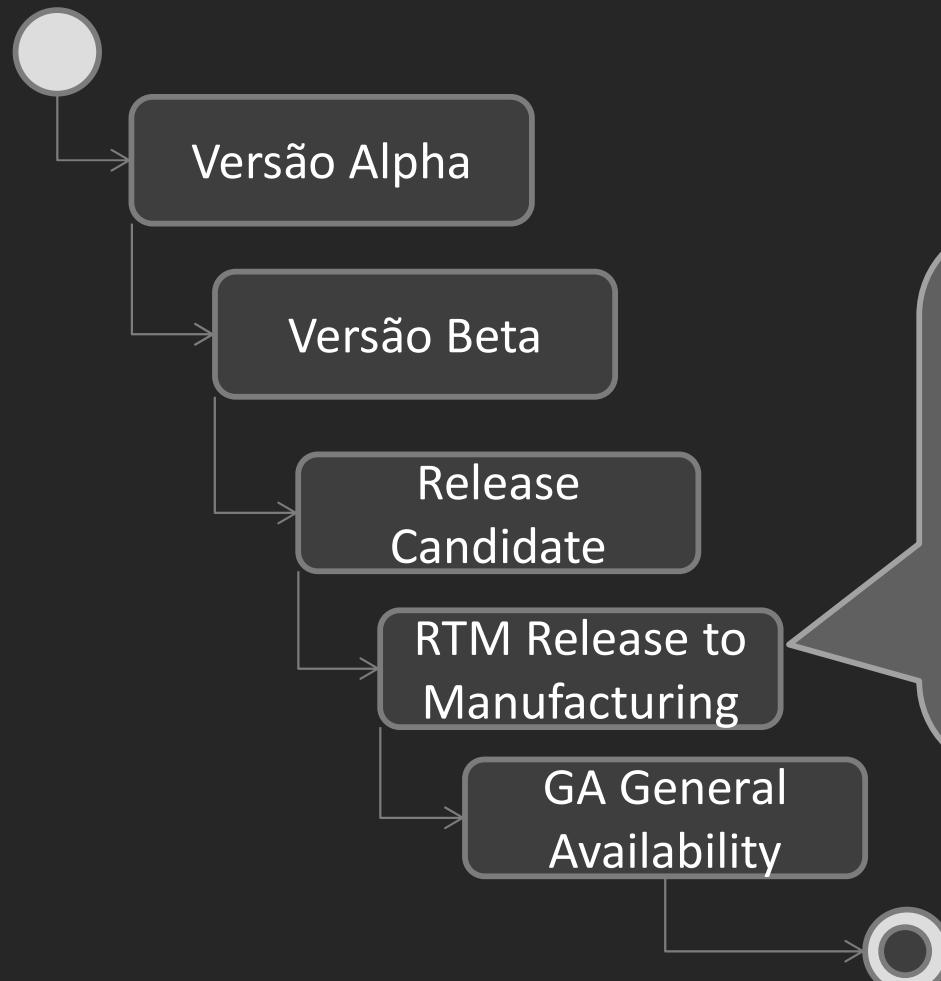
Conhecendo as versões do software durante um programa de liberação...



Refere-se a uma **versão com potencial para ser o produto final**, pronta para ser lançada a menos que surja alguma falha de grande impacto.  
Ela aguarda uma confirmação de data para empacotamento e liberação para comercialização.  
Podem ser selecionadas partes do produto para um lançamento definitivo.

## GERENCIAMENTO DA LIBERAÇÃO DO SOFTWARE

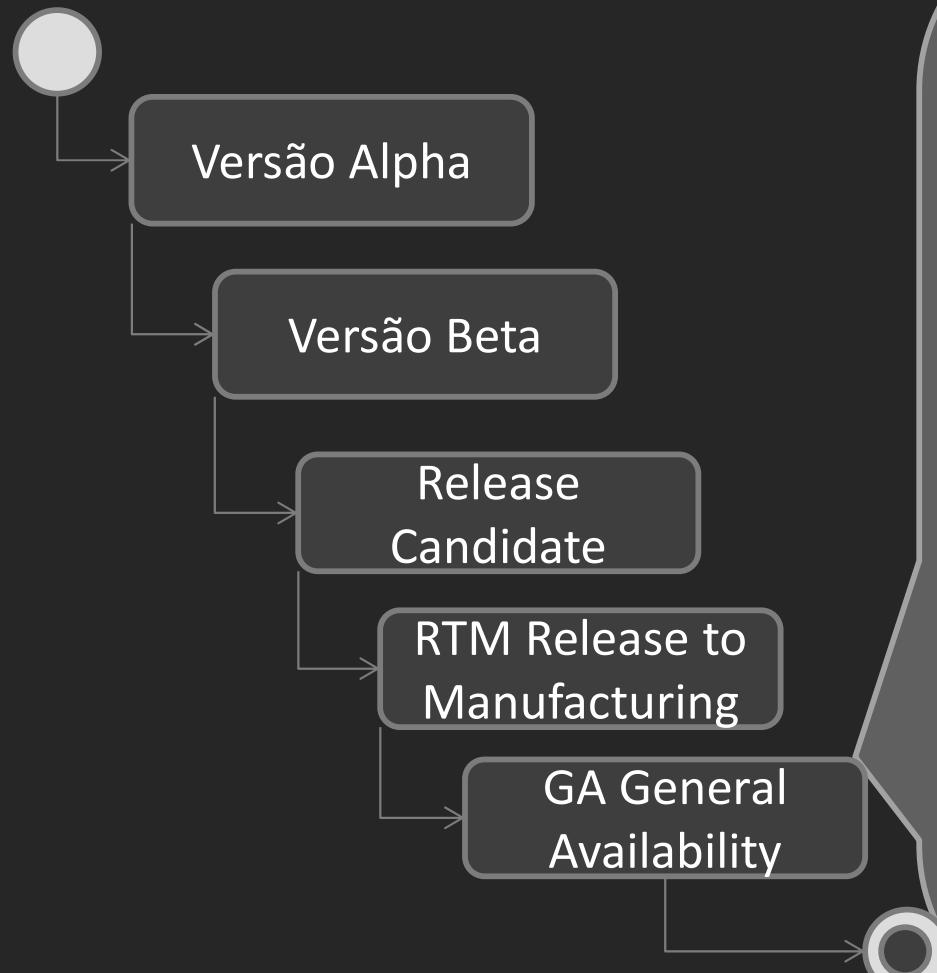
Conhecendo as versões do software durante um programa de liberação...



Versão empacotada para o mercado.  
Contém os **componentes de software definitivos** que compõem a versão que se será reproduzida (manufaturada) e distribuída em massa (com comercialização cobrada ou livre uso).

## GERENCIAMENTO DA LIBERAÇÃO DO SOFTWARE

Conhecendo as versões do software durante um programa de liberação...



O **produto** é considerado "vivo" e entra em fase de manutenção ou sustentação que inclui o suporte e atualizações para **ajustes** (manutenção corretiva para eliminar falhas, manutenção adaptativa para ajuste a uma mudança em regra de negócio, manutenção evolutiva para gerar inovação proativa em funcionalidades ou manutenção perfectiva para ajustar aspectos não funcionais à perfeição como o desempenho, visual e usabilidade).

## GERENCIAMENTO DE AMBIENTES DE INFRAESTRUTURA

Os ambientes de infraestrutura para atender a esse processo também devem ser cuidadosamente preparados e administrados:



Infra de desenvolvimento  
(envolve PC de  
desenvolvedores e servidores  
para criar e manter fontes e  
executáveis– versão Alpha)



Infra de teste de versões no  
ciclo de testes finais e  
liberação (versões Beta,  
Release Candidate, Release do  
Manufacturing)



Infra de uso diário  
para atender aos  
negócios / produtiva  
(versão General  
Availability)

*\*Os ambientes podem ser providenciados em nuvem ou em infra dedicada, com ou sem virtualização de máquinas e storage.*

## GERENCIAMENTO DE AMBIENTES DE INFRAESTRUTURA

Dados e aplicações são movimentados entre as plataformas:



Infra de desenvolvimento  
(versão Alpha)



Infra de (versões Beta, Release Candidate, Release do Manufacturing)



Infra de uso diário  
(versão General Availability)

\*Transportes levam novas versões de um ambiente para outro

\*Replicações de dados ocorrem do ambiente produtivo para testes e desenvolvimento para permitir avaliações de cargas de dados e estresse

\*Esses movimentos são **avaliados e autorizados pelos Comitês** de mudança e realizados por **ação conjunta de Desenvolvedores e Equipes de administração de Segurança, Redes, Servidores e Bancos de dados**.

## O TESTE E A GOVERNANÇA E QUALIDADE DO SOFTWARE

Diversas empresas dedicam muitas pessoas e horas de trabalho para liberar um software produzido, por conta da insegurança quanto a qualidade do produto.

Algumas empresas levam meses para testar um software que levou uma ou poucas semanas para ser produzido.

Estratégias comerciais e a eficiência operacional e financeira das empresas são comprometidas em razão desse tipo de acontecimento.



## O TESTE E A GOVERNANÇA E QUALIDADE DO SOFTWARE

Para que se tenha **eficiência, eficácia e efetividade em avaliações de software**, é preciso estratégia.

Em teste, uma **ESTRATÉGIA** envolverá **TÉCNICAS** para testar o software em diversos **NÍVEIS** de detalhe, quando ao **FUNCIONAMENTO** e aspectos **NÃO FUNCIONAIS**, em situações de **CRIAÇÃO** de algo novo ou **MANUTENÇÃO/MUDANÇA** de código existente.





**ESCUTE O PODCAST NO CANAL DO PROFESSOR  
INSPEÇÃO, CONTROLE E GARANTIA DA QUALIDADE E OS TESTES**

<https://youtu.be/x7LEfeJlqa8>

## O TESTE E A GOVERNANÇA E QUALIDADE DO SOFTWARE

Para começar a falar de qualidade, vamos conhecer um pouco das consequências da falta dela.

**Hoje, software está em tudo...**

**... nos computadores pessoais;...**

**... nos refrigeradores; ...**

**... nas casas; ...**

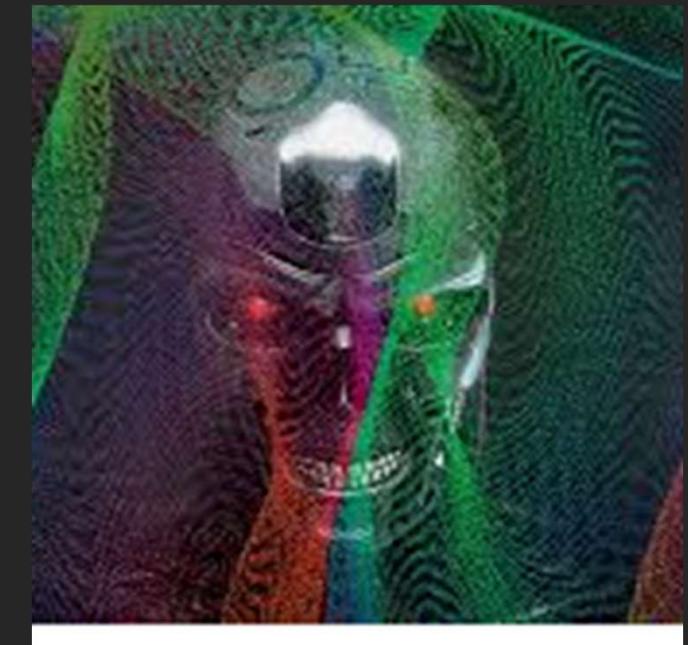
**... nos equipamentos hospitalares; ...**

**... na produção do seu remédio; ...**

**... na aplicação de herbicidas em alimentos;...**

**... na industrialização do seu alimento;**

**... nos automóveis; ...**



Esse é o tamanho da sua responsabilidade como desenvolvedor de software!

## O QUE IDENTIFICAMOS ATRAVÉS DE UM TESTE

O teste nos ajuda a identificar o que está errado.

### VÍDEOS – ACIDENTES NA F1

Nick Lauda

Airton Senna

## O QUE IDENTIFICAMOS ATRAVÉS DE UM TESTE

Quando algo dá errado, dizemos que algo FALHOU!

Os testes buscam por falhas!

Essas falhas, por sua vez podem ser causadas por defeitos e erros!

Erros são fruto da ação humana.

## O QUE IDENTIFICAMOS ATRAVÉS DE UM TESTE



Causada por



**DEFEITO** = CAUSA INTERMEDIÁRIA  
(defeito na suspensão quebrada)



Causado por

**ERRO** = CAUSA RAIZ  
(erro de aperto da suspensão pelo mecânico)

## O QUE IDENTIFICAMOS ATRAVÉS DE UM TESTE



Causada por

FALHA = APlicativo NÃO FUNCIONA

```
package project1;  
import java.util.Vector;  
  
public class TesteVector {  
    private Vector pessoas = new Vector();  
    public void addPessoa(String nome){  
        this.pessoas.addElement(nome);  
    }  
    public Vector getList(){  
        return this.pessoas;  
    }  
}
```

DEFEITO = CÓDIGO COM DEFEITO DE LÓGICA



Causado por

ERRO = PROGRAMAÇÃO OU ESPECIFICAÇÃO ERRADA

## O QUE IDENTIFICAMOS ATRAVÉS DE UM TESTE

**Erro:** Ação de um ser humano de cometer um engano.

**Defeito:** Componente que não funciona corretamente.

**Falha:** É o desvio do comportamento esperado.

## OBJETIVO DO TESTE

Se alguém coloca um carro nas suas mãos e pede que você teste ele, como você fará isso?

?



Vai só na maciota?

## OBJETIVO DO TESTE

Você bota pra quebrar!



## PROCEDIMENTO DO TESTE

Você seguiria uma sequência de passos para testar o carro?  
Quais seriam?

Engraçado?! Eu piso no pedal mas o carro  
não anda?! No carrinho de trombada do  
parquinho funciona!

?



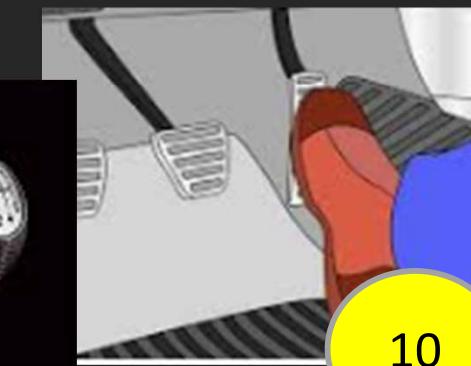
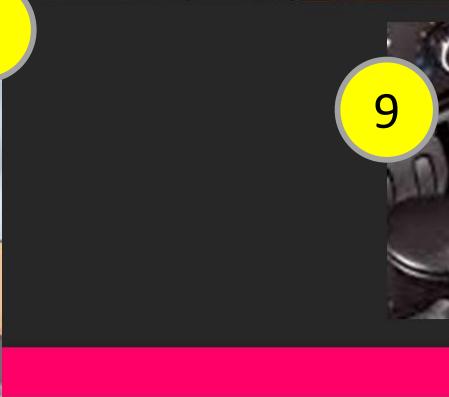
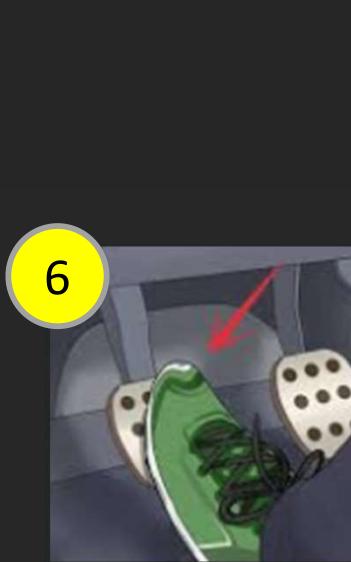
## PROCEDIMENTO DO TESTE

Sem procedimento, você simplesmente não consegue sair do lugar! Não consegue aplicar testes!



Nossa reação natural diante das práticas de teste: A PREGUIÇA!

Cacetada! Eu desisto de dirigir!



## PROCEDIMENTO DO TESTE

Sem procedimento, você simplesmente não consegue sair do lugar! Não consegue aplicar testes!

Se você vai testar a alteração ou a exclusão de um registro em uma tabela de um banco de dados, é necessário incluir um registro antes!

Com isso:

- 1º você testa a Inclusão de registros
- 2º Você testa a alteração do registro
- 3º Você testa a exclusão

Cumprir um procedimento de teste garante otimização do esforço de testar!

Passo 1  
Passo 2  
Passo “n”

## ORGANIZAÇÃO DO TESTE

Você deixaria quem fez o carro, testar o carro?

Você incluiria outras pessoas e responsabilidades no teste do carro?

?



## ORGANIZAÇÃO DO TESTE

O desenvolvedor sempre fará o teste dele!

Chamamos esse teste de Teste Unitário, cujo foco é verificar e validar um componente de software.

### CUIDADO!

O desenvolvedor geralmente tem a preguiça e desgosto por testar software.

Ele gosta de programar e em geral quer entregar logo tudo o que produz, pegando logo outra coisa nova pra fazer.



**ESCUTE O PODCAST NO CANAL DO PROFESSOR  
VERIFICAÇÃO E VALIDAÇÃO DE SOFTWARE**

<https://youtu.be/O9nVTrP5Bdg>

## CONCEITOS FUNDAMENTAIS DE TESTES DE SOFTWARE

Agora que já brincamos com exemplos cotidianos, vamos trabalhar Estratégias de Testes que nos levam a planejar adequadamente os testes de software, já que descobrimos que para testar software não basta somente sentar na frente do computador e sair usando!

Temos que saber o que queremos testar, temos que saber o método a aplicar e definir responsabilidades.

## CONCEITOS FUNDAMENTAIS DE TESTES DE SOFTWARE

Além do teste do desenvolvedor, precisamos aplicar outros testes com objetivos diferentes, sendo eles planejados e executados por pessoas diferentes.

Aplicamos em teste de software diversos Tipos de Testes, com diversas Técnicas de Testes para atender determinados Níveis de Testes que correspondem às diversas expectativas de pessoas sobre a qualidade de um software.

## NÍVEIS DE TESTE

Níveis de teste determinam ênfases de teste, com o objetivo de atender um determinado público/interesse, por exemplo:

- O usuário quer experimentar, quer operar o software livremente e perceber se ele é fácil e rápido de usar e se atende as funções que pediu;
- Arquitetos de solução querem confirmar se os componentes de software estão se comunicando adequadamente, se serviços estão sendo oferecidos e consumidos pelos programas corretamente;
- Os programadores querem garantir que a sua construção funciona (em geral aplicam testes querendo que a aplicação opere corretamente e não o contrário).



**ESCUTE O PODCAST NO CANAL DO PROFESSOR  
TESTE PARA FUNCIONAR E O TESTE DE VERDADE**

<https://youtu.be/M8gNB9E4ANE>

## NÍVEIS DE TESTE

Entendendo através de exemplos: caso do pneu furado



Seu pneu MURCHOU !



Você leva ao mecânico/borracheiro



## NÍVEIS DE TESTE

Entendendo através de exemplos: caso do pneu furado



2º - para que o pneu volte a funcionar de forma correta no carro...  
o mecânico procede a instalação do pneu e alinha e balanceia a roda (**TESTE DE COMPONENTES INTEGRADOS**)



1º - assim que o mecânico conclui o conserto... ele testa se o vazamento persiste, enchendo o pneu e colocando-o em uma banheira com água (**TESTE UNITÁRIO**)

## NÍVEIS DE TESTE

Entendendo através de exemplos: caso do pneu furado



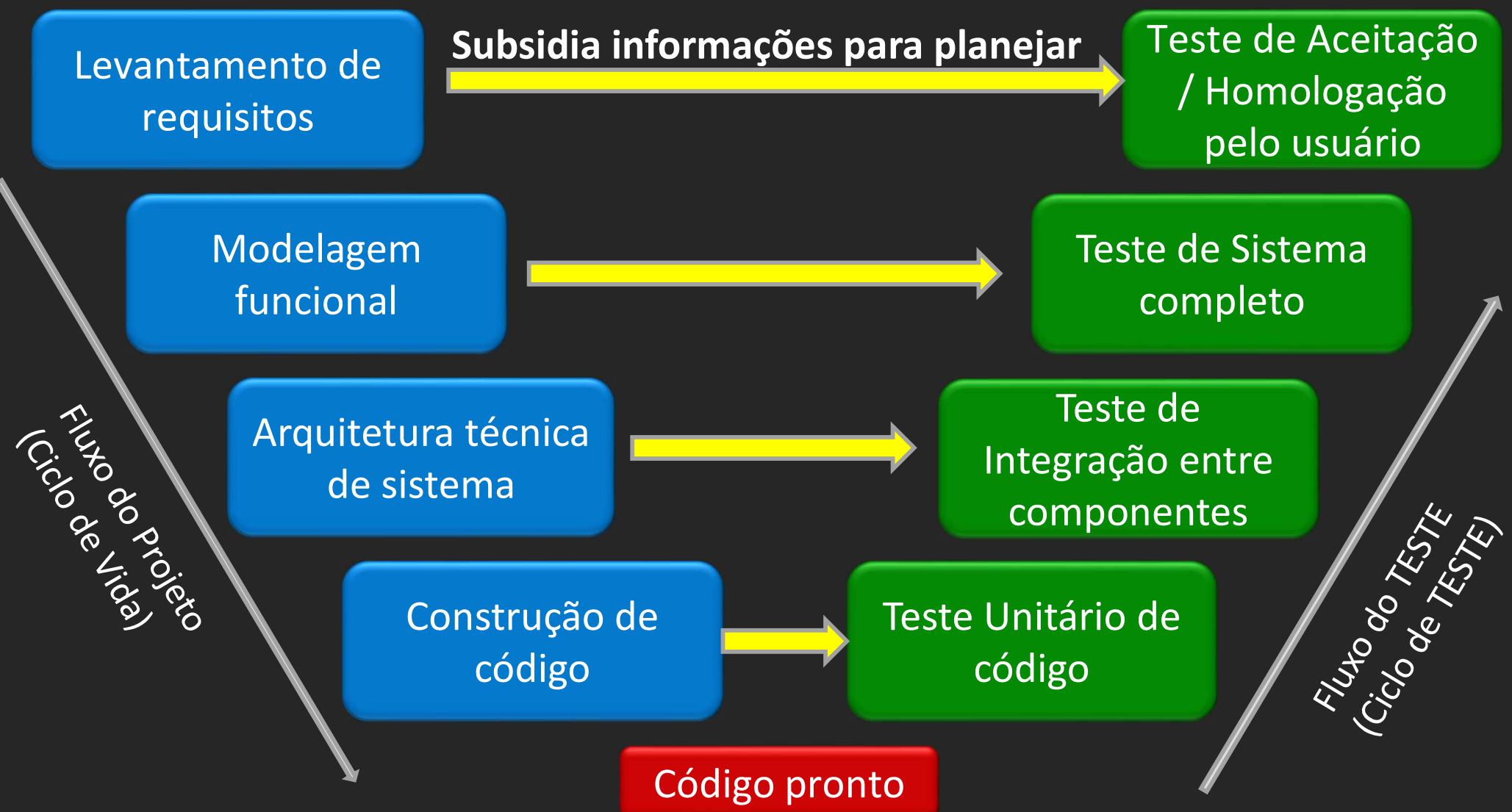
4º - você então, dirige o seu carro e confirma se o funcionamento está ok (**TESTE DE ACEITAÇÃO DO SISTEMA**)



3º - quando o mecânico termina o alinhamento e balanceamento ele procede um teste de rua (**TESTE DE SISTEMA FUNCIONANDO DE FORMA COMPLETA**)

## NÍVEIS DE TESTE

O modelo V de testes define esses níveis:



## NÍVEIS DE TESTE

O modelo V de testes define esses níveis:

Levantamento de requisitos

REQUERIMIENTOS	POJO No.	Completo	Incompleto	OBSERVACIONES
a) Presentar solicitud original y copia con la suma que indique: SE SOLICITA REGISTRO SANITARIO, con los siguientes datos: al Órgano al que se dirige: Dirección General de Regulación Sanitaria o prefectura de Región Departamental de Salud, según corresponda.				
b) Nombre y generales del propietario o representante legal del establecimiento y del apoderado legal				
c) Razón social o denominación de la sociedad				
d) Dirección exacta del establecimiento, incluyendo teléfono, fax, correo electrónico				
e) Datos y clasificación del producto: Nombre comercial y nombre genérico, fabricante, tipo de producto, al origen y/o fabricación y número de licencia Sanitaria, en el establecimiento que lo fabrica o lo comercializa, cuando el producto sea de origen nacional.				

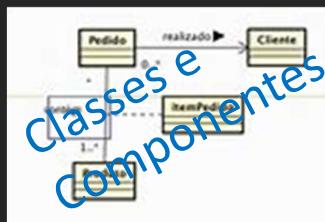
Teste de Aceitação / Homologação pelo usuário

Modelagem funcional



Teste de Sistema completo

Arquitetura técnica de sistema



Teste de Integração entre componentes

Construção de código



Teste Unitário de código

Código pronto

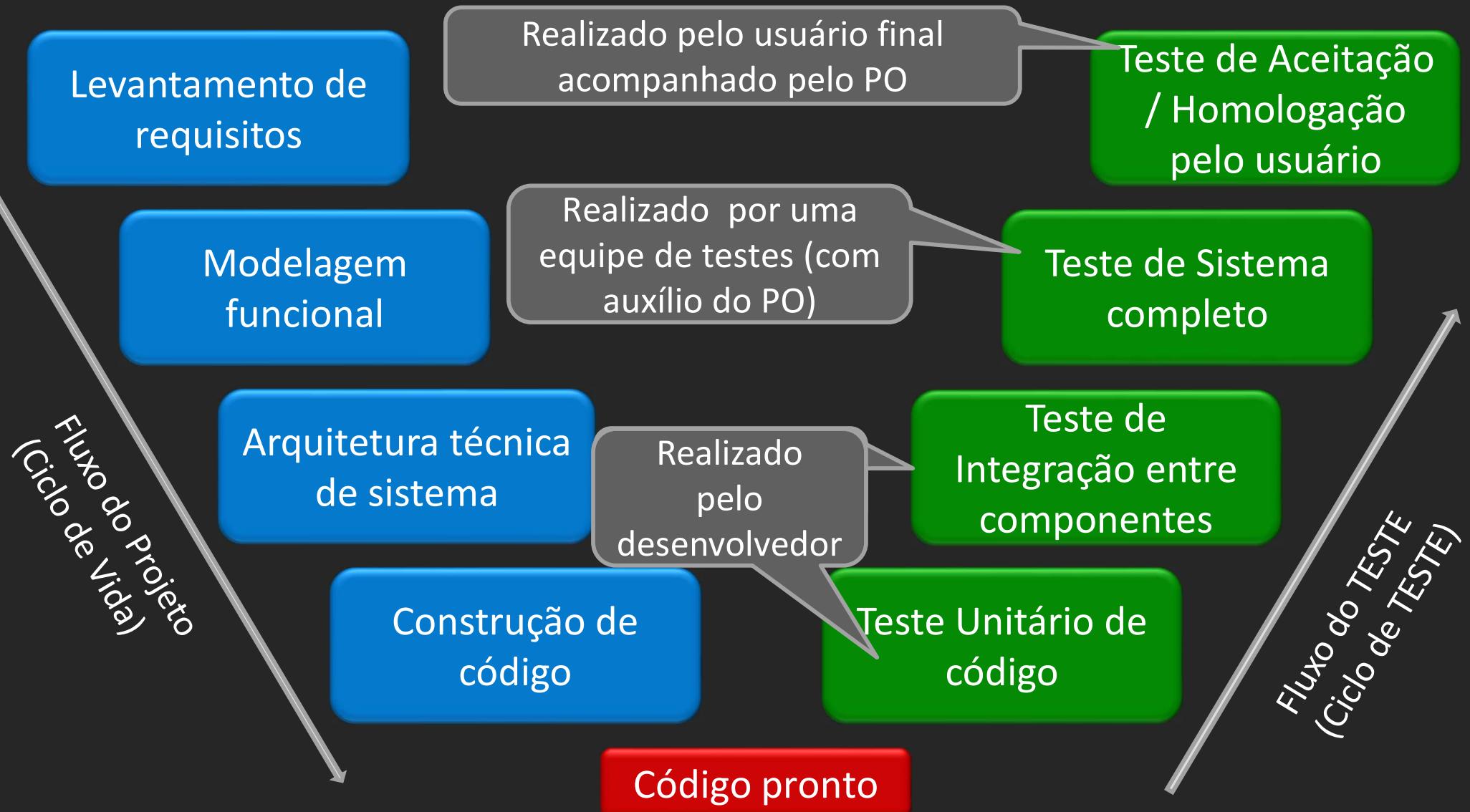
Fluxo do Projeto  
(Ciclo de Vida)

Lista de requisitos  
casos de Uso  
Classes e Componentes

Fluxo do TESTE  
(Ciclo de TESTE)

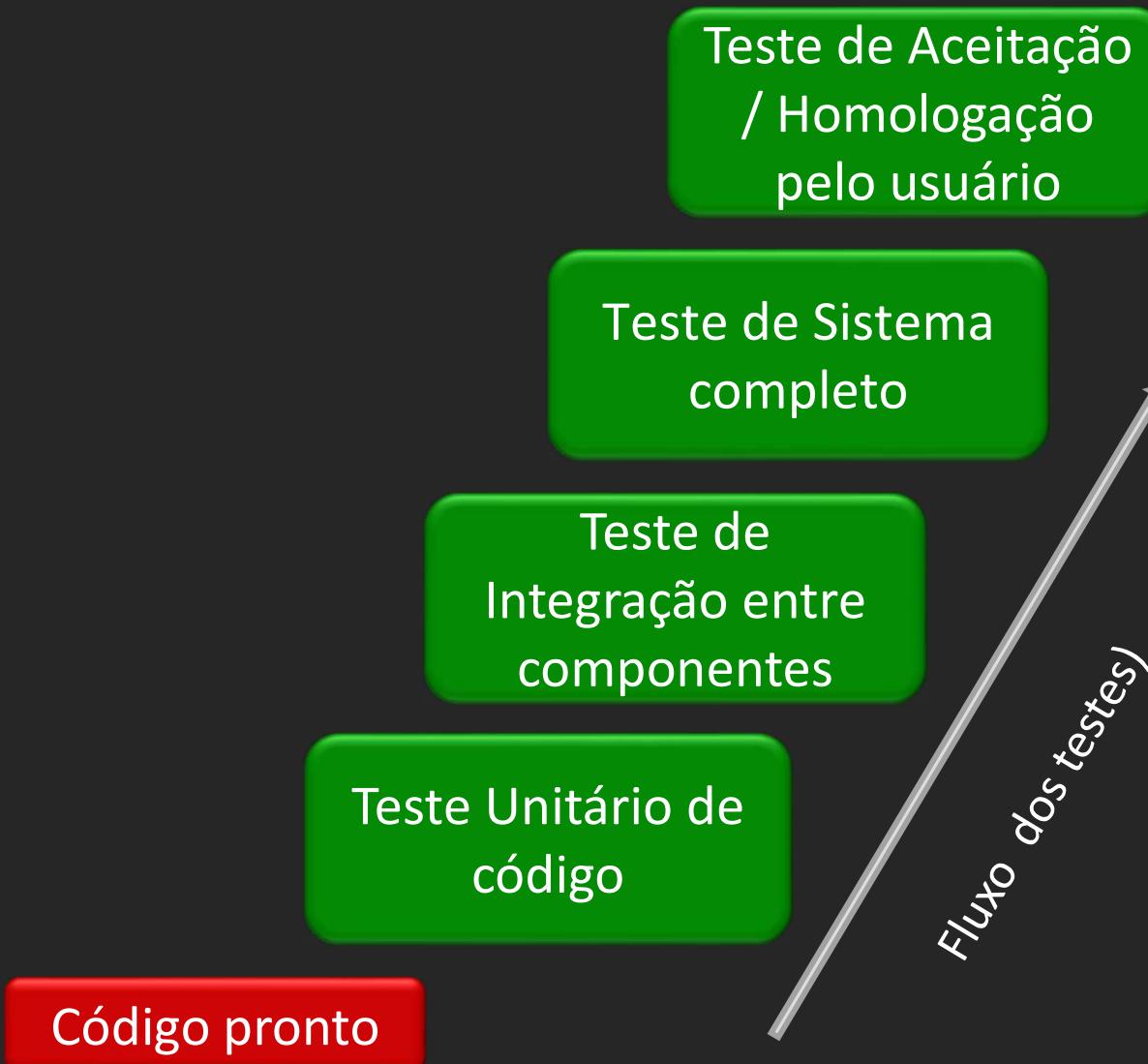
## NÍVEIS DE TESTE

O modelo V de testes define esses níveis:



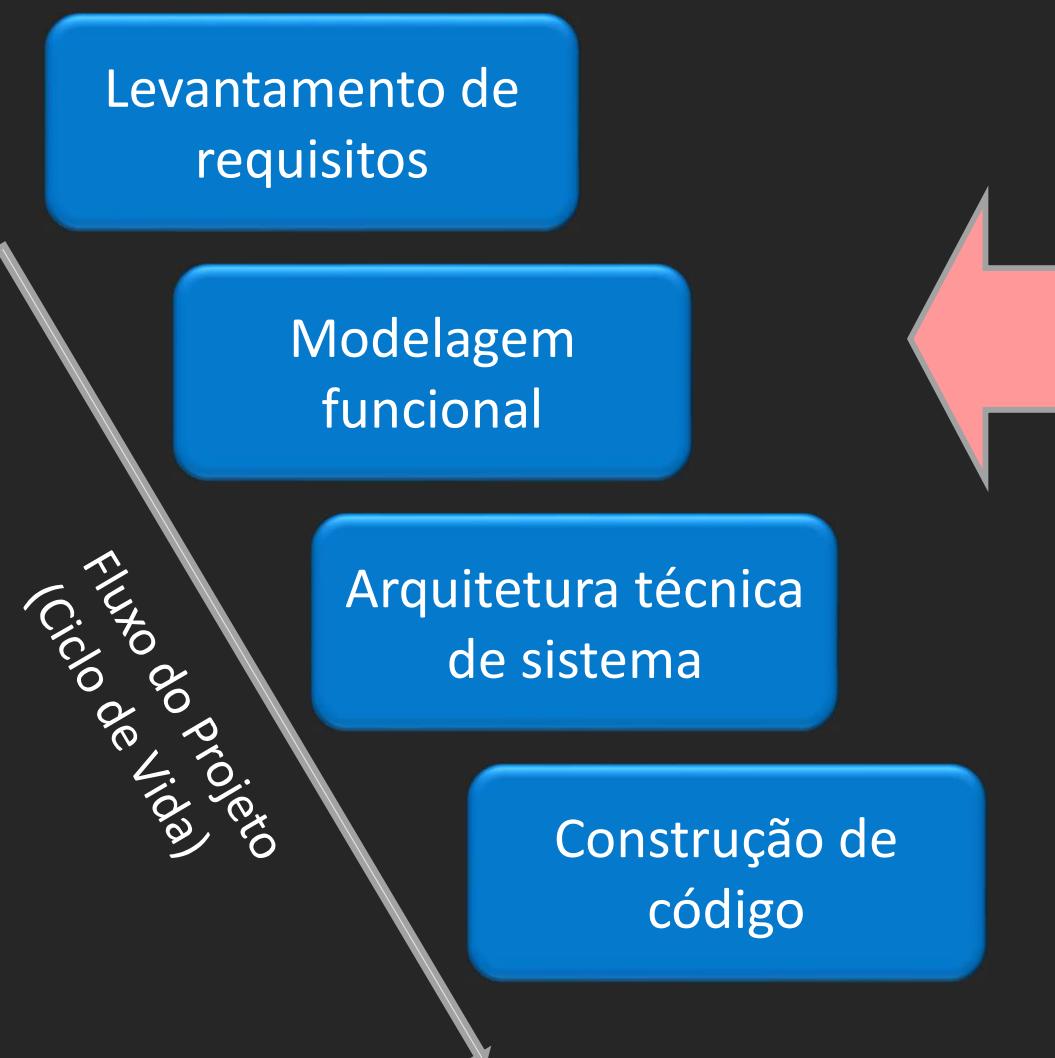
## NÍVEIS DE TESTE

Na produção de software, temos os níveis de testes:



## NÍVEIS DE TESTE

O modelo V de testes define esses níveis:

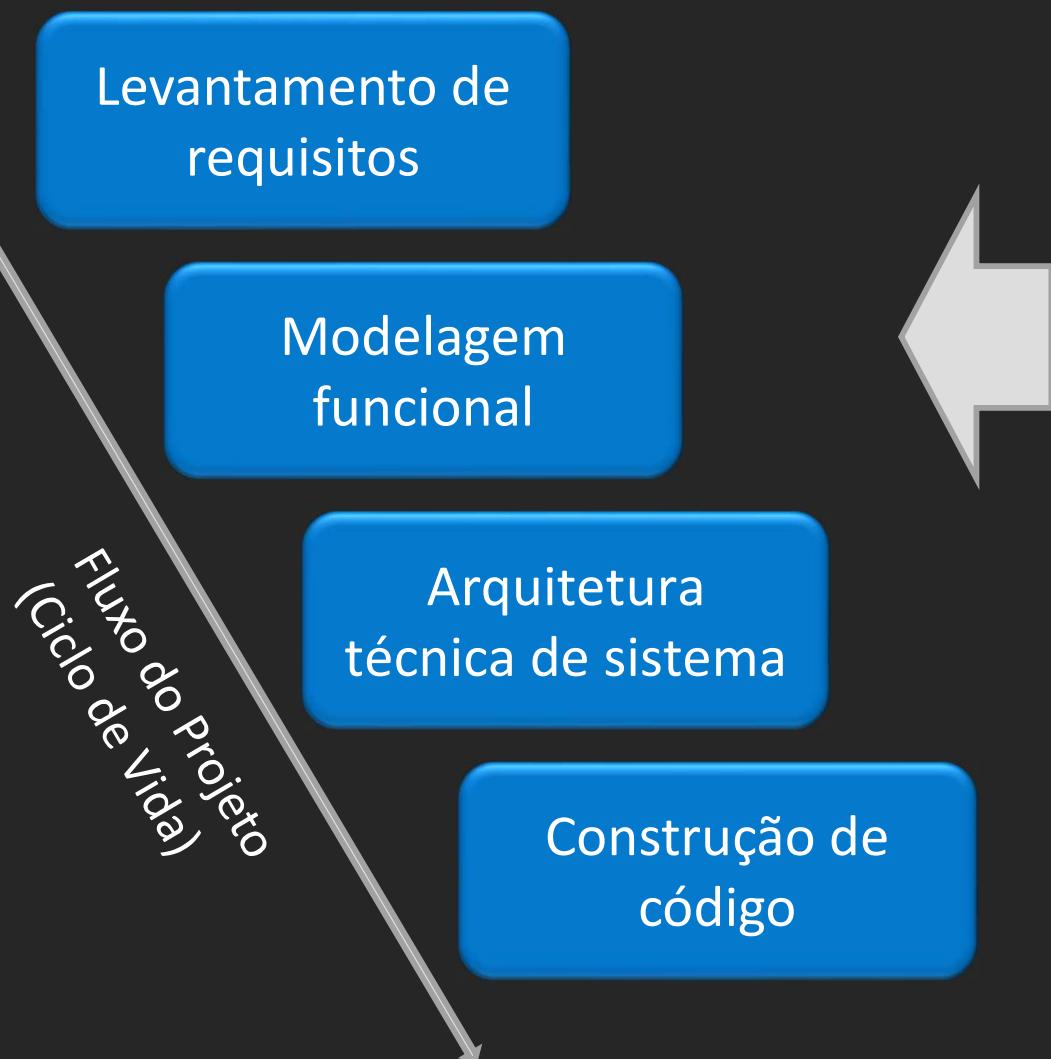


Durante as etapas produtivas, podemos **VERIFICAR** o cumprimento dos processos de trabalho definidos para a fábrica de software e gerenciamento de projeto do software.

Também podemos validar o conteúdo dos materiais produzidos, como textos e diagramas de especificação e estruturação e documentação interna de código de aplicação, atestando sua qualidade.

## NÍVEIS DE TESTE

O modelo V de testes define esses níveis:



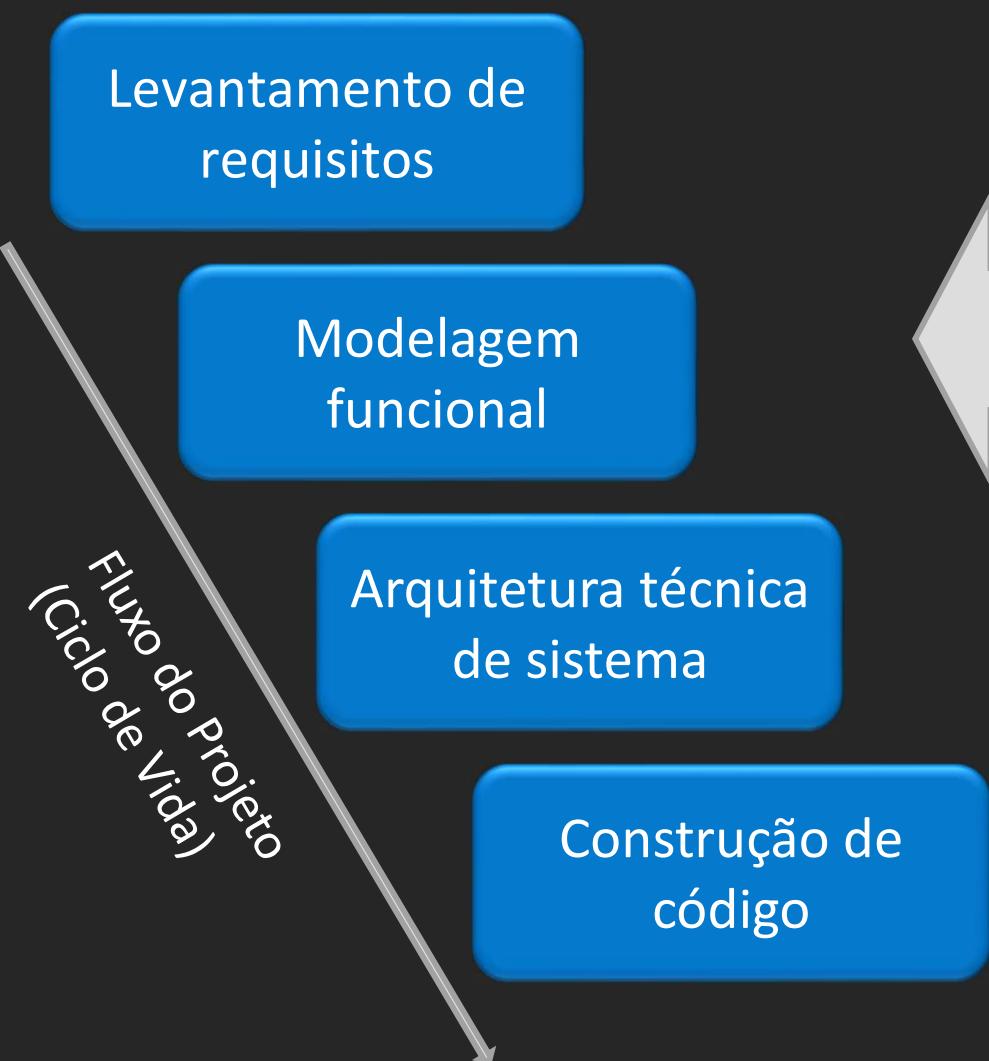
Podemos **verificar** se um Termo de Abertura de projeto foi feito, seguindo um modelo padrão de estrutura.

Podemos **verificar** se um código fonte foi escrito reutilizando um componente já construído que foi determinado pela arquitetura.

Podemos **verificar** se todos os métodos de uma Classe foram detalhados em Algoritmos

## NÍVEIS DE TESTE

O modelo V de testes define esses níveis:



Podemos **validar** se o Algoritmo de um Método de Classe de Objeto do sistema contém a lógica funcional correta.

Podemos **validar** se o Diagrama de Casos de Uso representa semanticamente todo o escopo da Lista Funcional representada no Backlog de Produto.

## NÍVEIS DE TESTE

O modelo V de testes define esses níveis:

Podemos **verificar** se a lista de testes que planejamos foi aplicada.

Podemos **verificar** se o status de cada teste aplicado (sucesso ou insucesso) está sendo apontado junto com os dados de data e hora e responsável pela aplicação do teste.

Teste de Aceitação / Homologação pelo usuário

Teste de Sistema completo

Teste de Integração entre componentes

Teste Unitário de código

Fluxo do TESTE  
(Ciclo de TESTE)

## NÍVEIS DE TESTE

O modelo V de testes define esses níveis:

Podemos **validar** se as chamadas de programas e passagens de parâmetros estão corretas.

Podemos **validar** se o resultado de um cálculo feito pelo software foi correto.

Podemos **validar** se o desempenho está adequado.

Podemos **validar** se o usuário consegue aprender o software sem pedir ajuda.

Teste de Aceitação / Homologação pelo usuário

Teste de Sistema completo

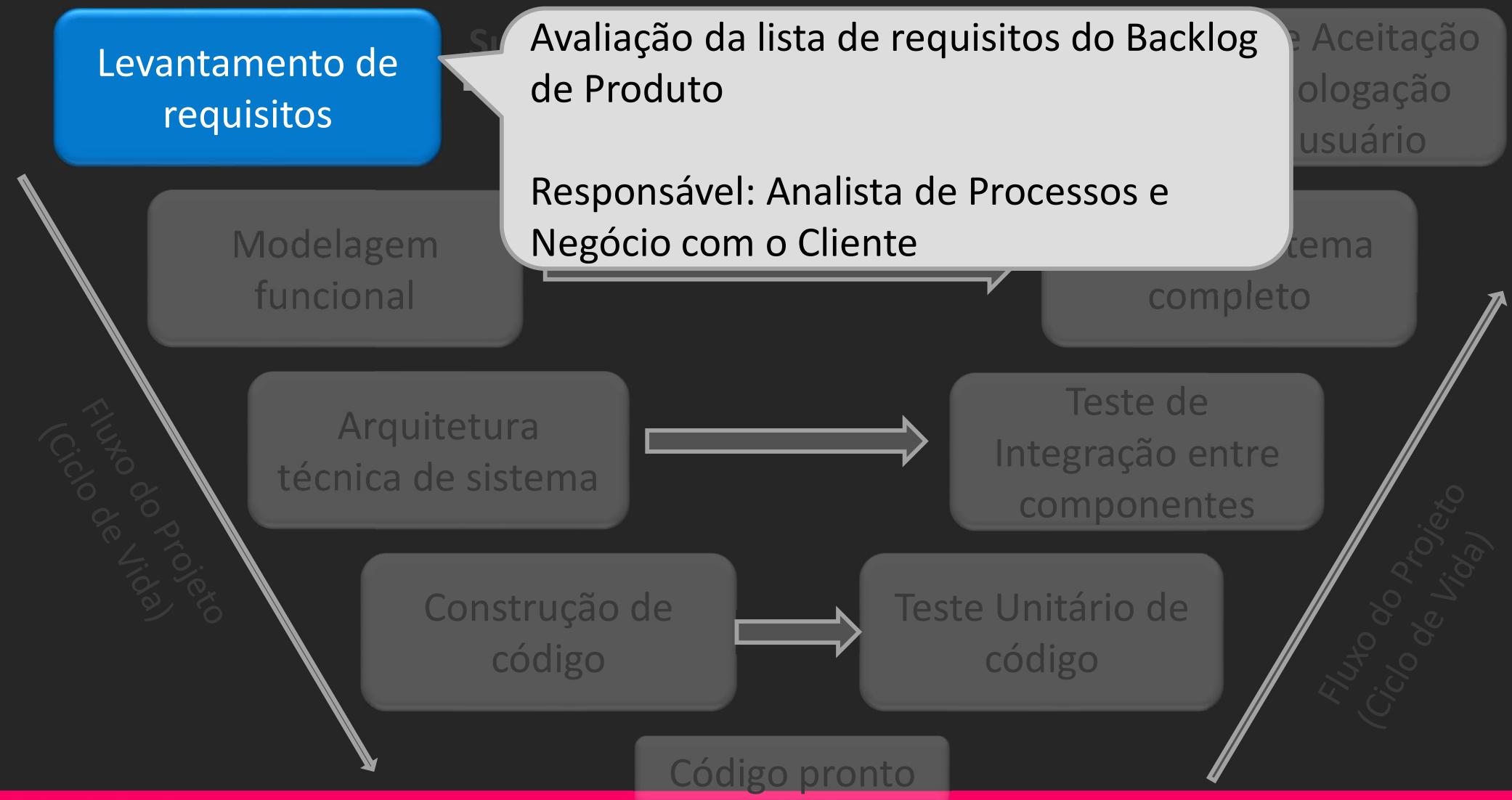
Teste de Integração entre componentes

Teste Unitário de código

Fluxo do TESTE  
(Ciclo de TESTE)

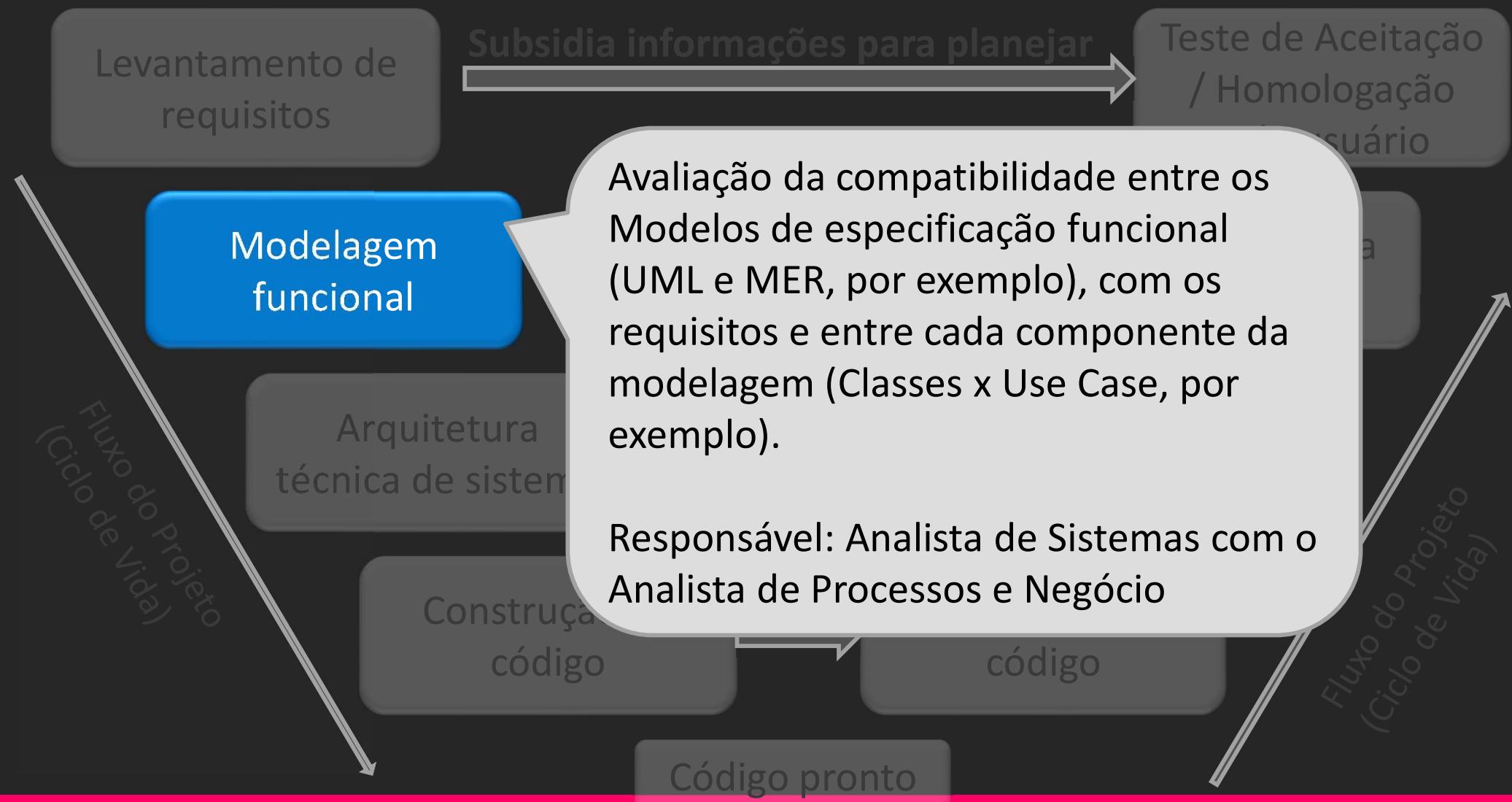
## NÍVEIS DE TESTE

Resumidamente avaliamos:



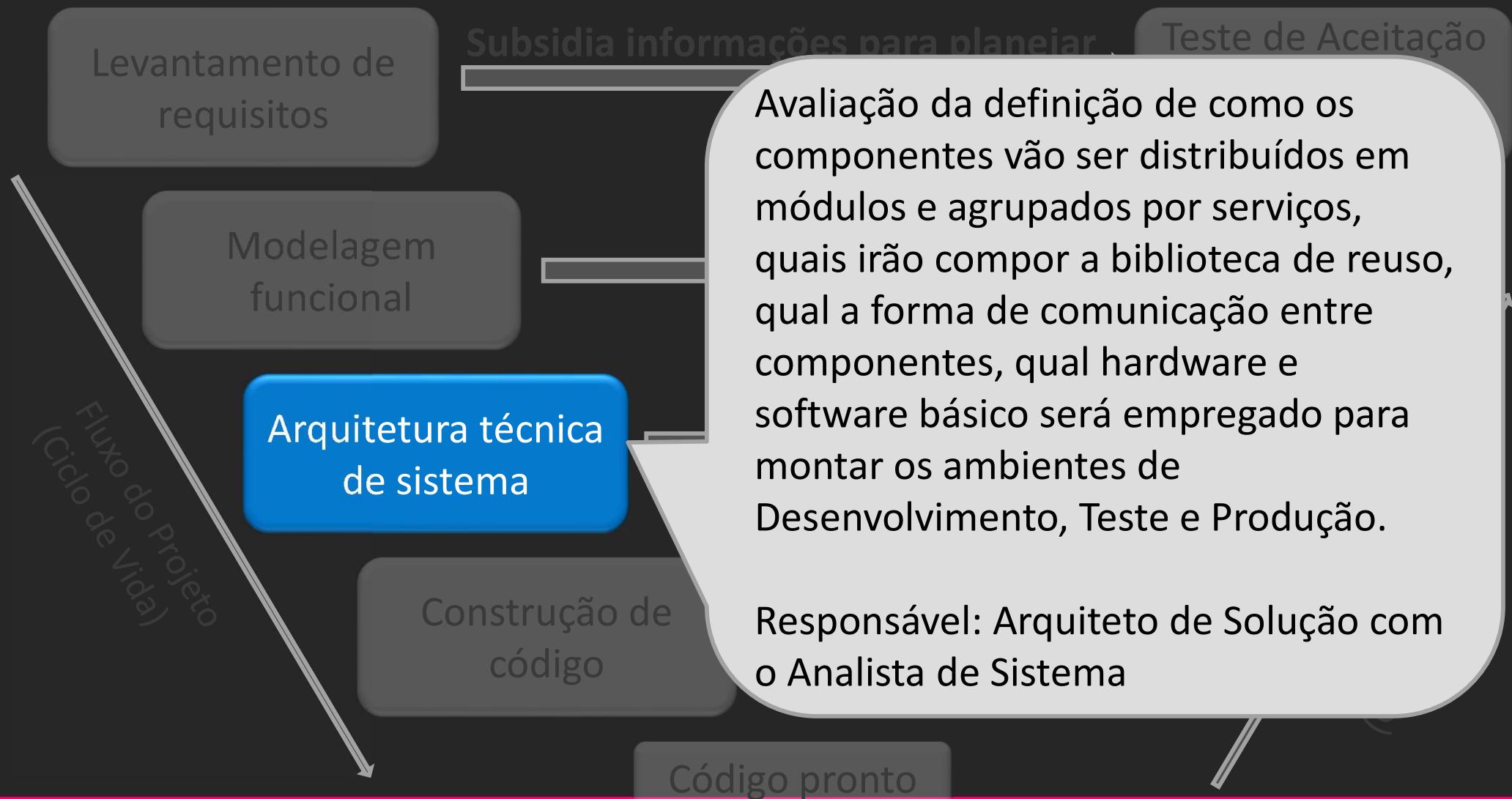
## NÍVEIS DE TESTE

Resumidamente avaliamos:



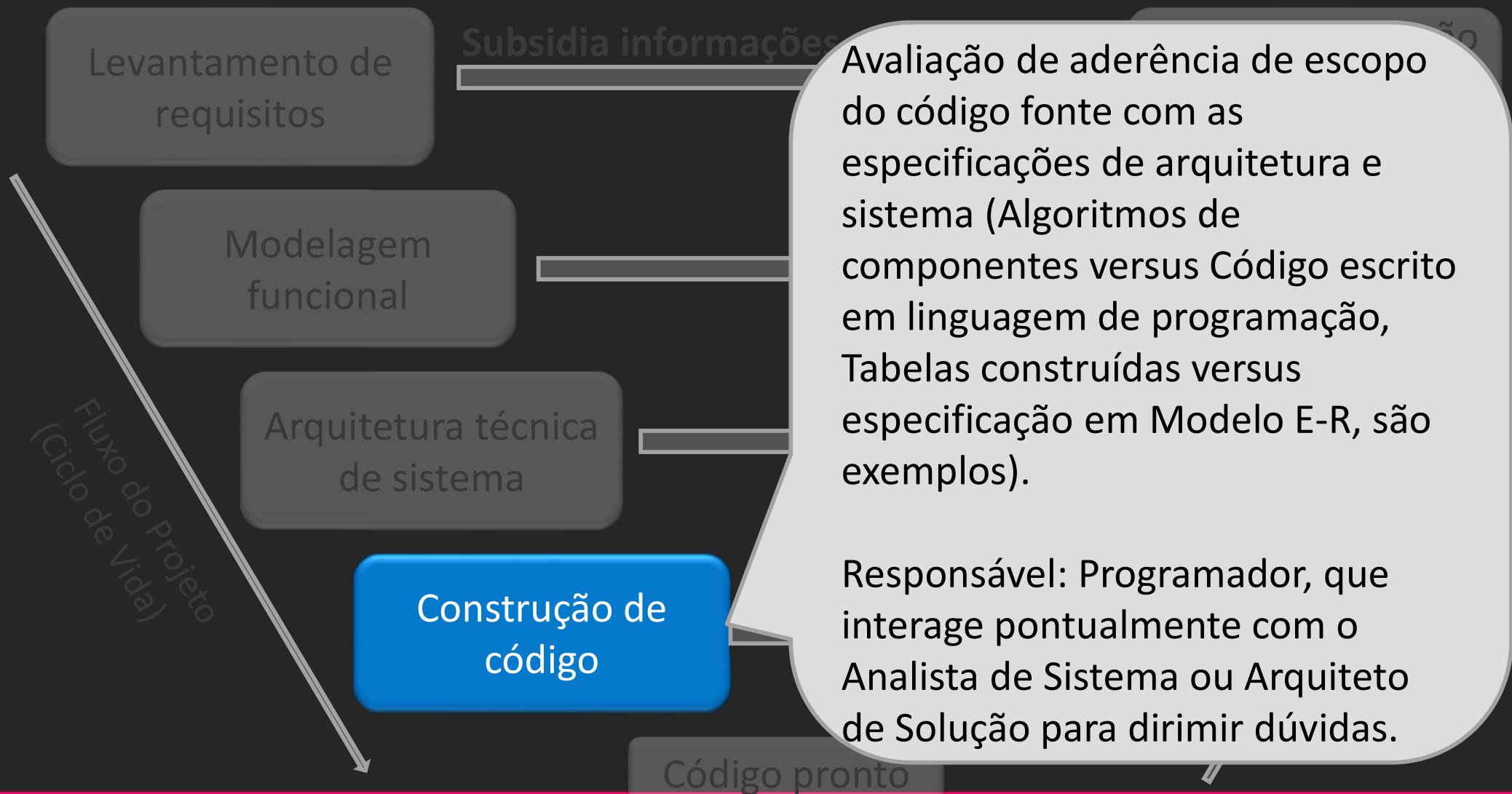
## NÍVEIS DE TESTE

Resumidamente avaliamos:



## NÍVEIS DE TESTE

Resumidamente avaliamos:



## NÍVEIS DE TESTE

Resumidamente avaliamos:

Levantamento de  
requisitos

Subsidiaria informações para planejar

Teste de Aceitação / Homologação pelo usuário

Avaliação do código fonte, seus desvios lógicos, seus cálculos, formatação de saída e validações de entradas de dados.

Responsável: Programador, que interage pontualmente com o Analista de Sistema ou Arquiteto de Solução para dirimir dúvidas.

Teste de Sistema completo

Teste de Integração entre componentes

Teste Unitário de código

(Ciclo)

Fluxo do Projeto (Ciclo de Vida)

## NÍVEIS DE TESTE

Resumidamente avaliamos:

Aplica um plano e um roteiro de casos de teste de comunicação entre componentes do sistema de aplicação, confirmando a passagem adequada de parâmetros, protocolos de comunicação, produção de arquivos de entrada e saída de dados.

Responsável: Especialistas em testes que alinharam quais avaliações fazer com os Analistas de Sistema e Arquitetos de Solução. Podem ou não contar com o apoio do Programador

Código pronto



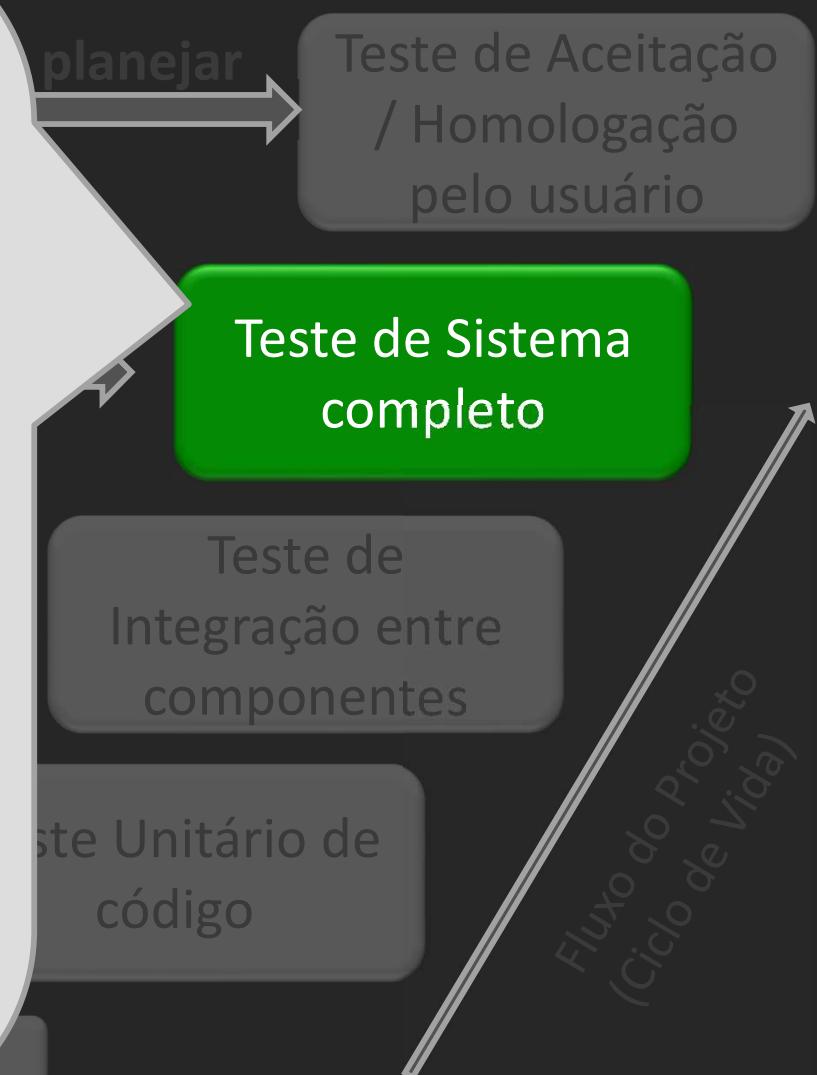
## NÍVEIS DE TESTE

Resumidamente avaliamos:

Avaliação do funcionamento do sistema como um todo, com todos os seus programas de aplicação, simulando as rotinas de negócio.

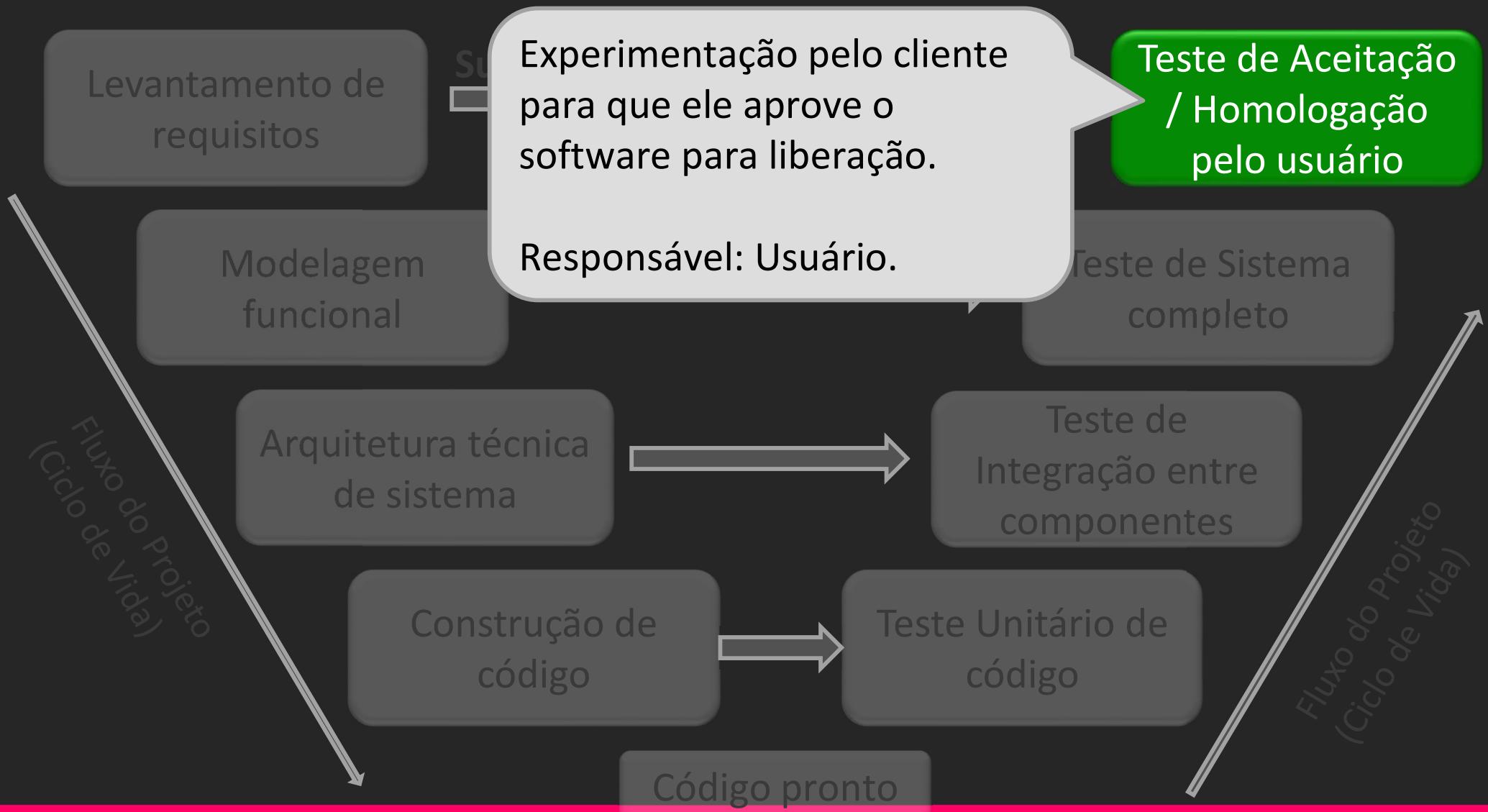
Os testes são direcionados aplicando um plano de testes que inclui roteiros e casos de testes com controle de dados de entrada e saída previstos, tomando como base os Casos de Uso e Cenários de Uso.

Responsável: Especialistas em testes que alinharam quais avaliações fazer com os Analistas de Sistema e têm acesso aos documentos de especificação funcional e não funcional e modelos sistêmicos.



## NÍVEIS DE TESTE

Resumidamente avaliamos:



## NÍVEIS DE TESTE

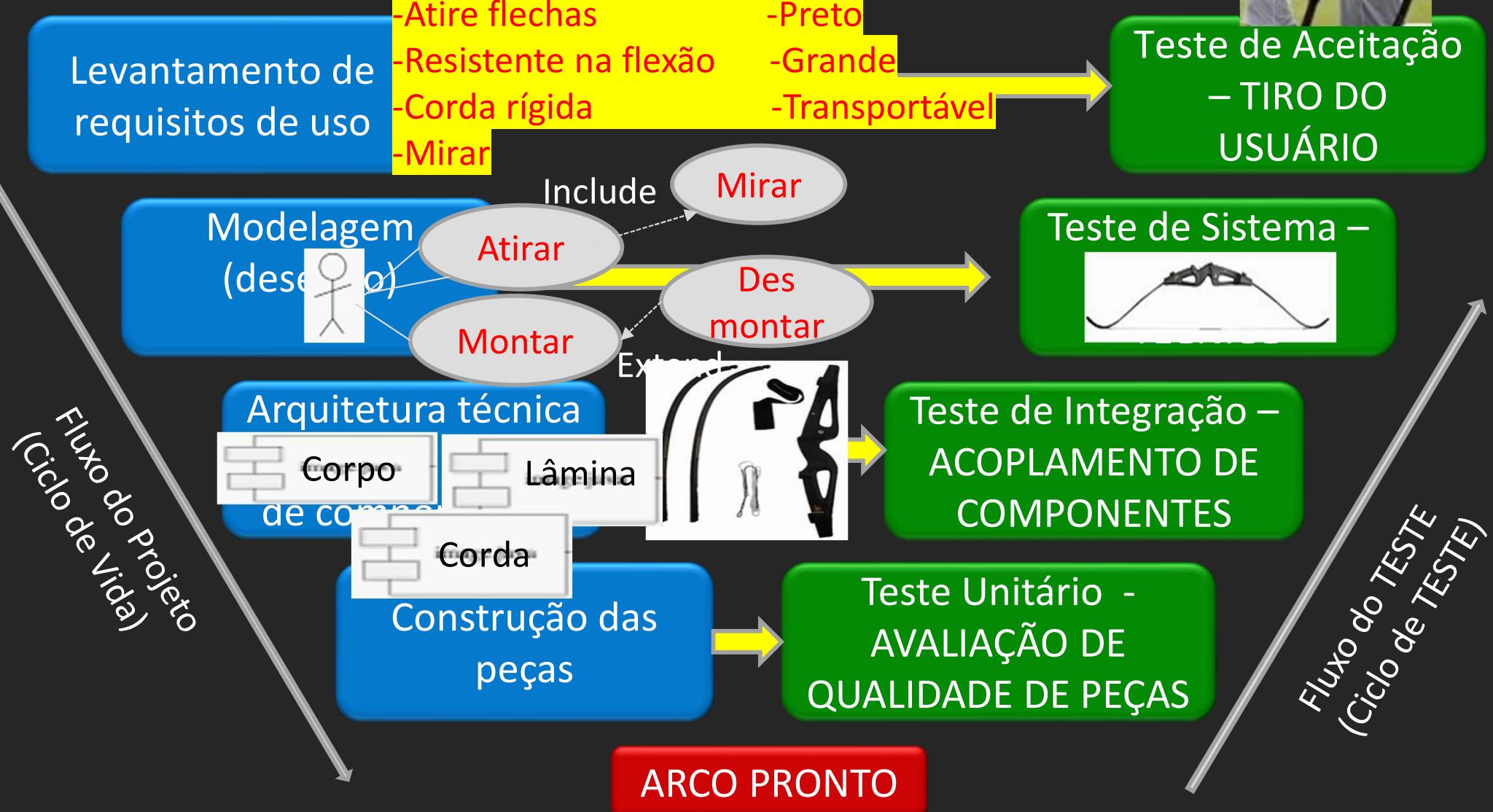
Brincando com níveis de teste... Exemplo lúdico



Teste de BOW and ARROW

## NÍVEIS DE TESTE

O modelo V de teste define os seguintes níveis:



## TIPOS DE TESTE

Tipos de teste **definem a ênfase**, o propósito do teste.:

- Avaliar o funcionamento do software (Teste Funcional), olhando para o serviço que o software deve prestar, a informação que deve gerar, guardar, processar, distribuir, proteger – **Teste do Tipo FUNCIONAL**.
- Avaliar atributos da qualidade dos software não relacionados com o seu funcionamento como a estética, facilidade de uso, velocidade de resposta - **Teste do Tipo NÃO FUNCIONAL**.
- Avaliar desenhos técnicos, projeto de engenharia e arquitetura do software – **Teste Estrutural**.
- **Teste de Mudanças**

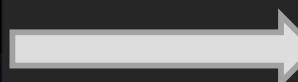
## TIPOS DE TESTE

Quais testes funcionais, não funcionais e estruturais você consegue identificar para o carro de F1?



## TIPOS DE TESTE

Quais testes funcionais, não funcionais e estruturais você consegue identificar para o carro de F1?



### Funcionais:

- Andar para frente
- Fazer curva



### Não Funcionais:

- Espaço no cockpit
- Visualização de instrumentos



### Estruturais:

- Correto acoplamento de peças
- Existência de folgas dentro de limites de tolerância da engenharia
- Conectividade elétrica e eletrônica

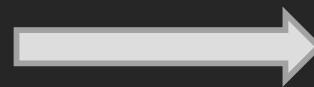


## TIPOS DE TESTE

Em software.



Software para Planilhar Cálculos



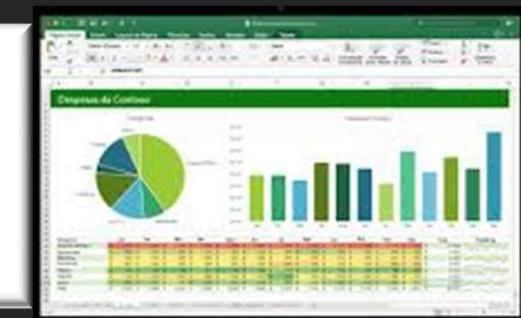
### Funcionais:

- Permitir tabulação de dados
- Fazer soma
- Calcular percentual
- Traçar gráfico de Pizza



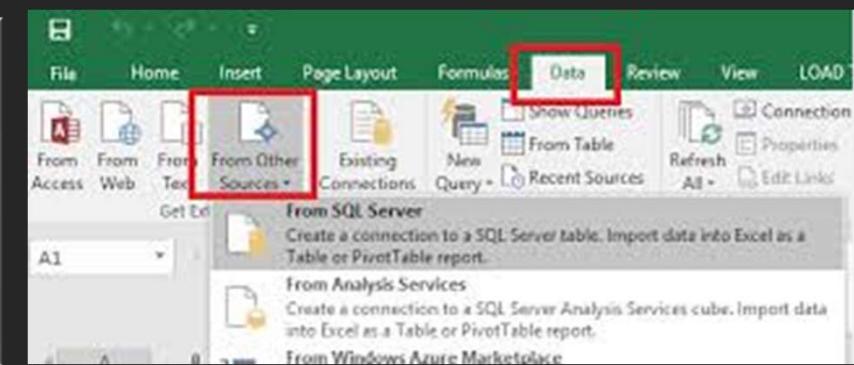
### Não Funcionais:

- Beleza dos gráficos
- Facilidade para digitar fórmulas



### Estruturais:

- Conectividade OLE com PowerPoint e MS-Word
- Protocolo de integração com Bancos de Dados MS-SQL



## TIPOS DE TESTE

Brincando com tipos de teste... Exemplo lúdico



Teste de BOW and ARROW

## TIPOS DE TESTE

Teste de **MUDANÇA**.

Se você alterou um programa entre 1.205 programas fonte desenvolvidos, o que você testaria?

Só a sua aplicação?

Todo o sistema com os demais programas?



## TIPOS DE TESTE

Teste de **MUDANÇA**.

Os testes sobre mudanças são tratados de uma forma especial, pois:

- 1º) Não precisamos testar todo o sistema novamente quando mudamos um componente – precisamos testar apenas os componentes relacionados, observando as matrizes de integridade referencial.
- 2º) Podemos **reaproveitar testes já elaborados** anteriormente para reavaliar programas que não foram modificados mas serão re-testados, focando esforços em construir novos casos de testes somente para aquilo que é novo.

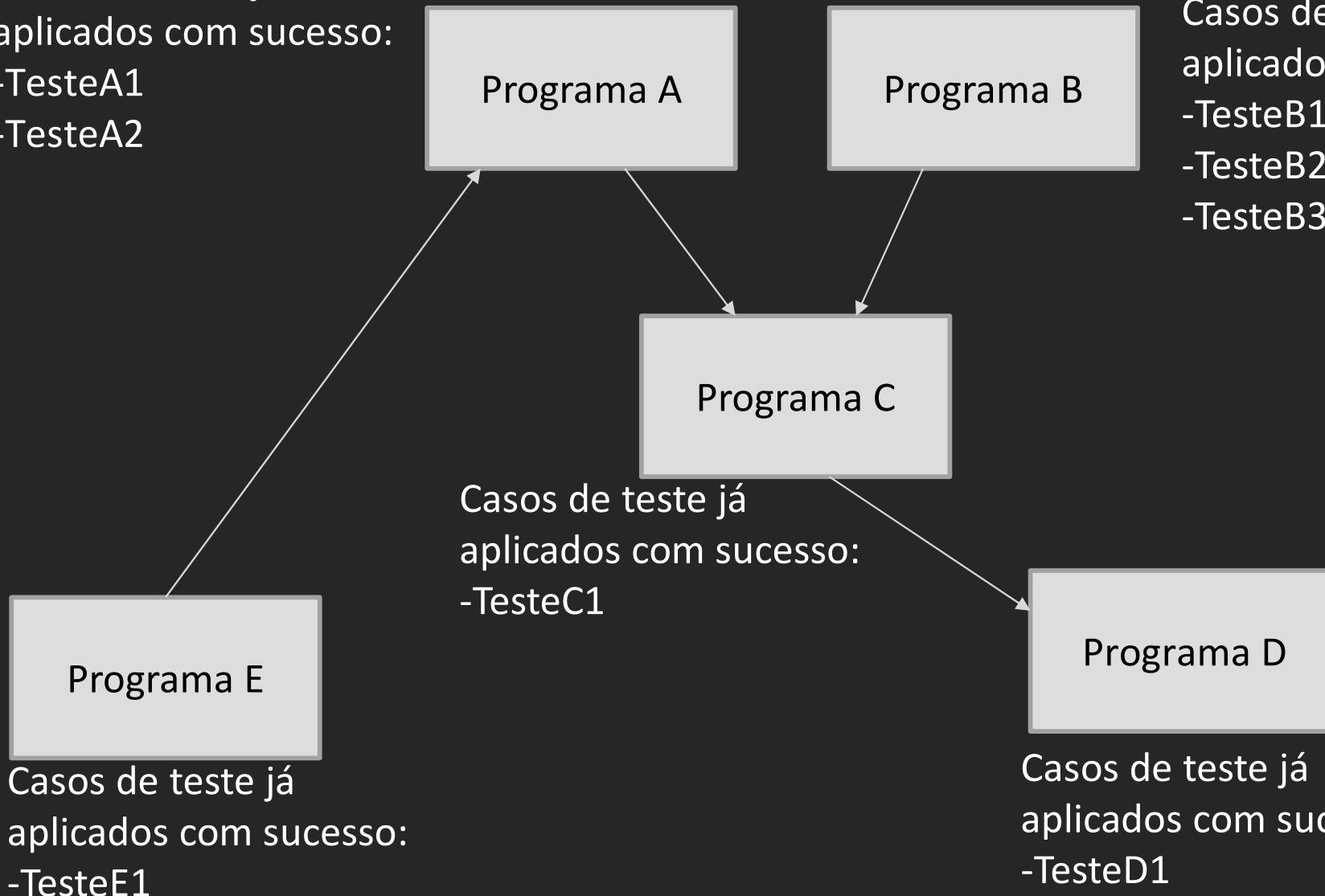
## TIPOS DE TESTE

Os testes de **MUDANÇA** têm dois subtipos:

- **Teste de Confirmação (Re-Teste)**: avalia se o **componente modificado** está eficaz, eficiente e efetivo.
- **Teste de Recessão**: avalia se **os componentes não modificados** mas que operam em conjunto com aquele que foi alterado, continuam operando com eficácia, eficiência e efetividade. São testes repetitivos que verificam que, a cada nova versão do sistema, partes que já estavam funcionando corretamente, continuam funcionando e não estão sendo afetadas pelas mudanças. A mecanização desses testes deve ser visada, uma vez que são repetitivos.

## SISTEMA JÁ HOMOLOGADO

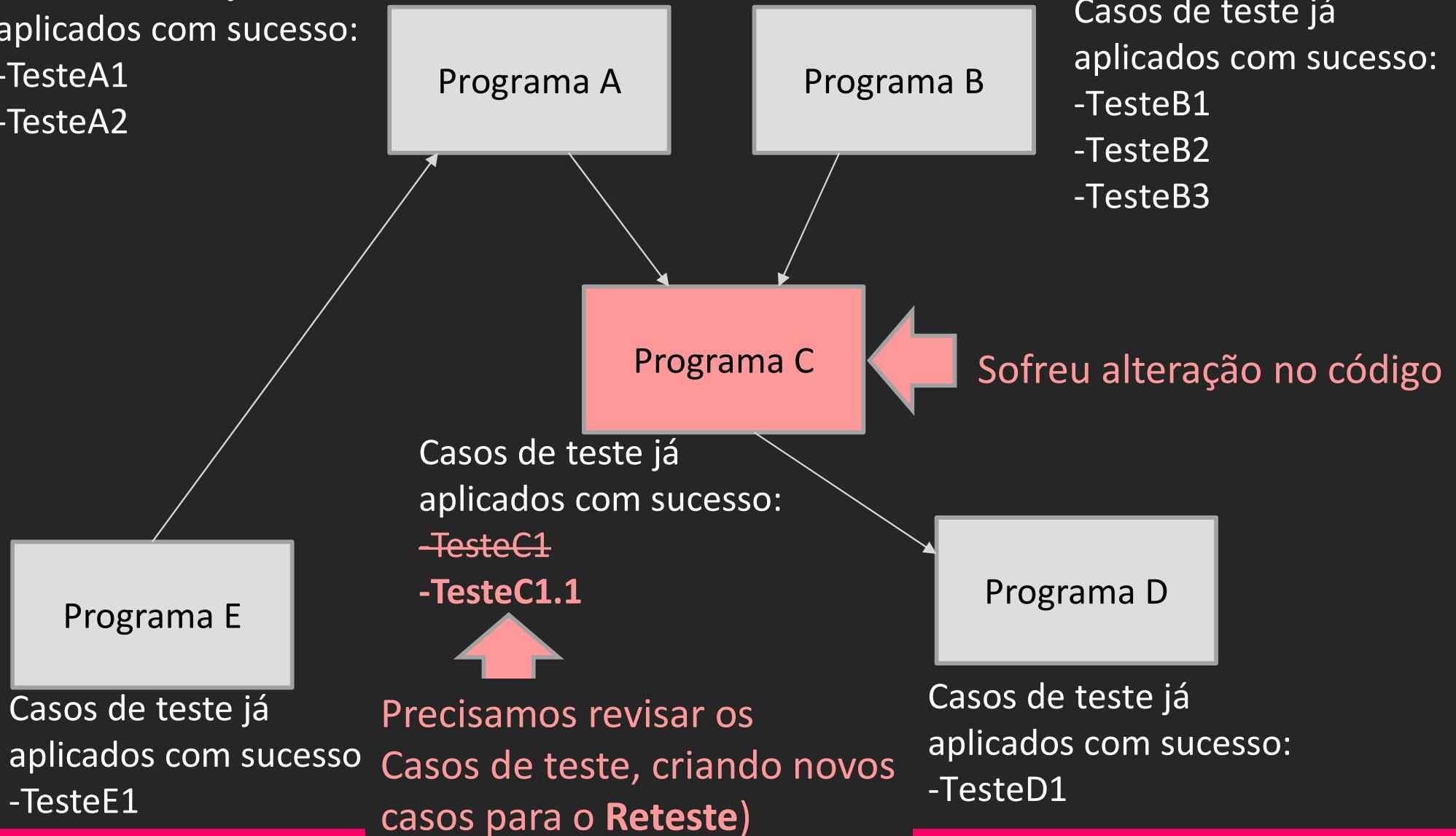
Casos de teste já aplicados com sucesso:  
-TesteA1  
-TesteA2



SISTEMA JÁ HOMOLOGADO sofre modificação

Casos de teste já aplicados com sucesso:  
-TesteA1  
-TesteA2

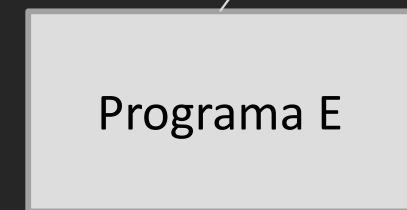
Casos de teste já aplicados com sucesso:  
-TesteB1  
-TesteB2  
-TesteB3



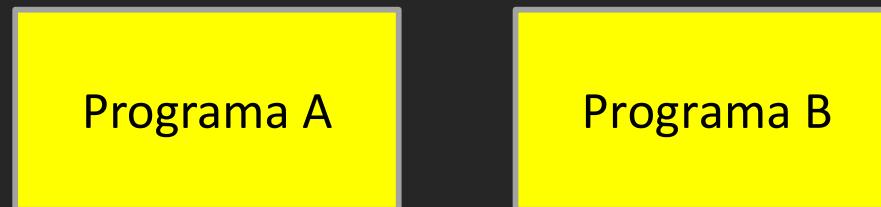
## SISTEMA JÁ HOMOLOGADO – procedimento de Reteste e Regressão

Fazemos a regressão aos programas que consomem o serviço da aplicação modificada e reaplicamos os Casos de teste já aplicados com sucesso, sem mudança:

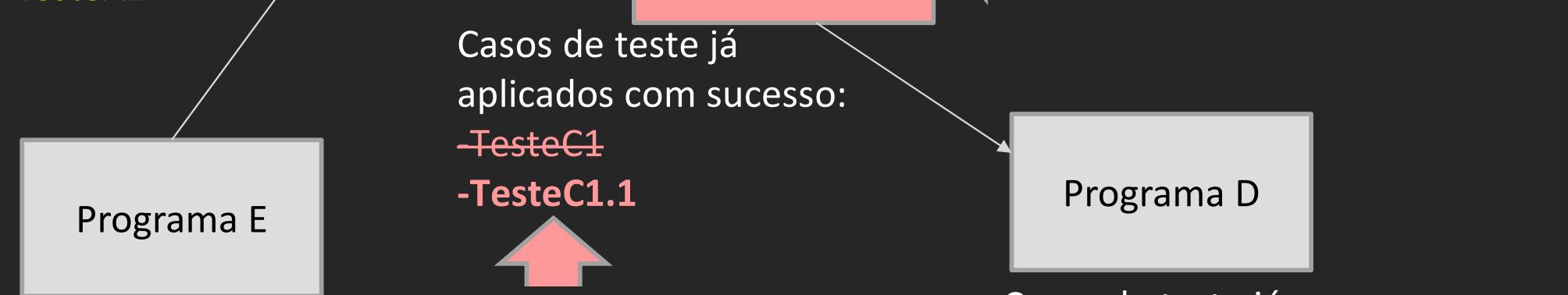
- TesteA1
- TesteA2



Casos de teste já aplicados com sucesso  
-TesteE1



Casos de teste já aplicados com sucesso, são reapplados nos testes de Regressão:  
-TesteB1  
-TesteB2  
-TesteB3



Precisamos revisar os Casos de teste, criando novos casos para o Reteste)

PERCEBA QUE O TESTE DE REGRESSÃO TEM UMA RELAÇÃO DIRETA COM O MAPA DE INTEGRIDADE REFERENCIAL DOS COMPONENTES DE PROJETO (discutido no material de aula sobre Gestão integrada de projeto).

Se não existir um software como o Azure + GIT que auxilie na identificação dos “work itens” relacionados, é possível construir uma matriz de relacionamento de requisitos e componentes de software que aponte quais são os componentes impactados quando um deles for alterado:

	Prog.01	Prog.02	Prog.03	Prog.04
Prog.01	X		X	
Prog.02		X	X	X
Prog.03			X	
Prog.04				X

Nesse exemplo de matriz, perceba que se o programa Prog.01 for alterado, o Prog.03 e o Prog.04 são impactados, porque eles CHAMAM o Prog.01. Considerando isso, Prog.03 e Prog.04 sofrem testes de regressão, se o Prog.01 tiver mudança e re-teste.

## TÉCNICAS DE TESTE

As **técnicas de teste** direcionam como fazer para realizar os testes, o modus operandi recomendado.

São elas:

- Caixa Preta: Usamos o item avaliado sem entrar no mérito da sua construção interna. Fornecemos Inputs e esperamos Outputs. Nesta técnica de teste, não importa a estrutura interna do software.
- Caixa Branca: Abrimos o item avaliado e exploramos os seus detalhes internos. Acompanhamos passo a passo da execução ou uso do item avaliado. Estamos interessados em conhecer a lógica/conteúdo interno.

## TÉCNICAS DE TESTE

As **técnicas de teste** direcionam como fazer para realizar os testes, o modus operandi recomendado.

São elas:

- Caixa Preta: NÃO PRECISAMOS CONHECER A ESTRUTURA INTERNA DA APLICAÇÃO PARA TESTÁ-LA! BASTA SABER QUE SAÍDA SE ESPERA DE CADA ENTRADA DE DADOS.
- Caixa Branca: PRECISAMOS CONHECER A ESTRUTURA INTERNA (LÓGICA) DA APLICAÇÃO PARA ELABORARMOS TESTES QUE AVALIEM CADA TRECHO DA LÓGICA.

## TÉCNICAS DE TESTE

Brincando com técnicas de teste...



Teste de NERF

## TÉCNICAS DE TESTE

Indique técnicas e teste para o carro de F1?



## TÉCNICAS DE TESTE

Indique técnicas e teste para o carro de F1?



### Caixa preta:

- Pisar no acelerador e calcular o tempo/espaço de aceleração
- Pisar no freio e calcular o tempo/espaço de frenagem
- Virar o volante e ver se o carro segue o curso apontado

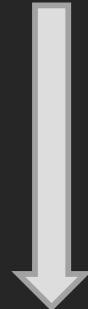
### Caixa branca:

- Abrir o motor após certa quilometragem de uso e verificar desgastes de peças
- Capturar sinais de movimento de componentes sensores da suspensão e dos pneus
- “Debugar” o software de controle de monitoração da suspensão

Para o caso de software...



Software para Planilhar Cálculos



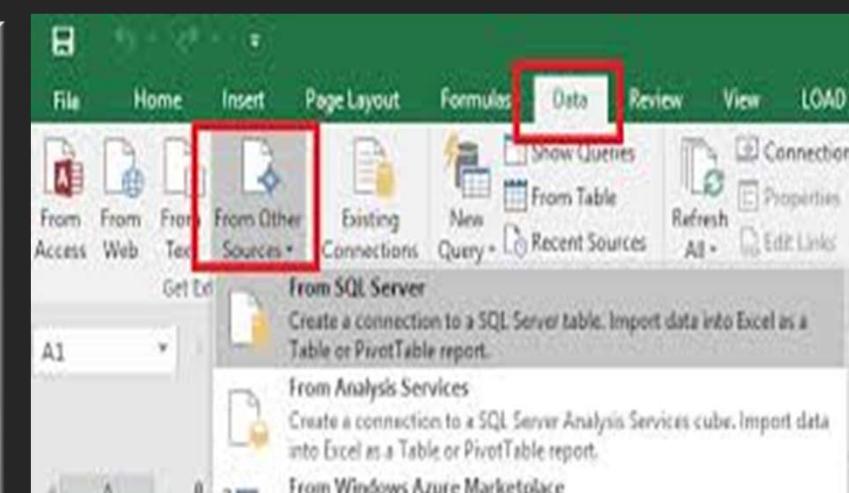
### Caixa Preta:

- Preencher as células da planilha
- Aplicar fórmulas
- Verificar resultados de cálculos
- Gerar gráficos
- Confirmar o alinhamento dos resultados com as entradas e equações



### Caixa Branca

- Avaliar o código fonte do programa de planilha de cálculo
- Observar parâmetros que entram e saem de rotinas
- Exibir e observar a instanciação de variáveis



# Síntese da Estratégia de Testes

## NÍVEIS DE TESTE

Teste de Aceitação / Homologação pelo usuário

Teste de Sistema completo

Teste de Integração entre componentes

Teste Unitário de código

## TIPOS DE TESTE

Funcional

Não Funcional

Regressão

## TÉCNICAS DE TESTE

Caixa Preta

Caixa Branca

## AUTOMAÇÃO DE TESTES

Os testes podem ser automatizados.

A automação envolve a **criação** de Engines que **executam** o nosso software de aplicação **sem intervenção humana**, facilitando a reaplicação perfeita do caso de teste eleborado.

A automação é especialmente interessante em caso de testes que teremos que **aplicar e reaplicar** diversas vezes.

Com a automação, **agiliza-se** o processo de execução e conferênciça do sucesso de testes e **reduz-se** o consumo de tempo do time de desenvolvimento ou equipes focadas em testes para executar a tarefa de avaliação do software.

## AUTOMAÇÃO DE TESTES



**ESCUTE O PODCAST NO CANAL DO PROFESSOR  
AUTOMAÇÃO EM TESTE DE SOFTWARE**

<https://youtu.be/MnwLCOy9A3g>

## ESTRATÉGIAS DE TESTE EM MÉTODOS ÁGEIS

Os níveis, técnicas e tipos de testes devem ser aplicados mesmo em projetos ágeis!

Se estiver usando o SCRUM, por exemplo:

- O Product Owner define e aplica testes de aceitação;
- O Time de desenvolvimento define e aplica os testes de Sistema e Integração;
- O Desenvolvedor define e aplica os testes unitários.

## ORGANIZAÇÃO PARA REALIZAR TESTES



**ESCUTE O PODCAST NO CANAL DO PROFESSOR  
EQUIPES DE TESTE**

<https://youtu.be/iZd4D7P5A6U>

## CASOS, ROTEIROS E PLANOS DE TESTE

A Estratégia de Teste depende da elaboração de um conjunto de documentos que explicam os testes a serem executados para atender os Níveis de avaliação, estabelecendo metas de prazos e responsabilidades.

## CASOS, ROTEIROS E PLANOS DE TESTE

A Estratégia de Teste incluirá:

- Casos de teste: explicam passo a passo o que deve ser simulado (interações de usuário ou software de teste com a aplicação), quais entradas de dados necessárias, quais as saídas esperadas. **Aplicam um Tipo e uma Técnica de teste.**
- Roteiros de teste: demonstram a sequência em que os Casos de teste devem ser aplicados, buscando otimização do processo de testar. Formam pacotes de testes a serem aplicados, ficando cada pacote a cargo de um responsável. **Cobrem um determinado Nível de teste.**
- Planos de teste: explicam o cronograma de aplicação dos Roteiros de Testes, associando responsáveis por cada Roteiro.

## CASOS, ROTEIROS E PLANOS DE TESTE

Exemplo de documentação de Estratégia de Teste:

<b>Plano de testes - Projeto XPTO - Sprint 1</b>	
<b>Avaliação do entregável - E1</b>	
<b>Roteirio de testes - R1</b>	
Aplicar Caso de teste T1	
Aplicar Caso de teste T3	
Aplicar Caso de teste T9	
Aplicar Caso de teste T2	
<b>Roteirio de testes - R2</b>	
Aplicar Caso de teste T15	
Aplicar Caso de teste T7	
Aplicar Caso de teste T8	
Aplicar Caso de teste T4	
<b>Avaliação do entregável - E2</b>	
<b>Roteirio de testes - R3</b>	
Aplicar Caso de teste T20	
Aplicar Caso de teste T16	

## CASOS DE TESTE

Os casos de testes que devem ser construídos têm que cobrir dois objetivos:

- **Teste POSITIVO:** avalia se o software funciona, se forem seguidos os passos corretos de utilização.
- **Teste NEGATIVO:** avalia o quanto o software tolera e trata as tentativas de uso indevido que deveriam ocasionar falhas.

## CASOS DE TESTE

Para não se perder na execução de testes...

O **objetivo** é esclarecer quais são os testes mais importantes de serem aplicados sobre o software, respondendo:

- O que testar;
- Como testar;
- Quantos testes realizar;
- Como registrar os testes e resultados para permitir auditoria da qualidade.

Vamos percorrer os Níveis, Tipos e Técnicas de testes, criando uma estratégia!

## COMO PLANEJAR UM CASO DE TESTE

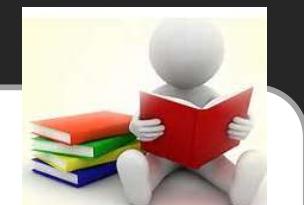
Se você tem pouco tempo para testar o seu software, quais são os testes mínimos que você deve planejar?



## COMO PLANEJAR UM CASO DE TESTE

Alguns dos Métodos mais populares de planejar Casos de Testes são:

- Teste Exploratório – para o nível de Homologação
- Teste com base em Caso de Uso – para o nível de SISTEMA
- Controle de fluxos de chamadas de funções externas – para INTEGRAÇÃO
- Avaliação de Complexidade Ciclomática – para o nível de TESTE UNITÁRIO
- Teste de Enlace – para o nível de TESTE UNITÁRIO
- Avaliação de Limites – para o nível de TESTE UNITÁRIO
- Avaliação de Condição de Equivalência – para o nível de TESTE UNITÁRIO



C vai lhe ensinar essas técnicas a seguir

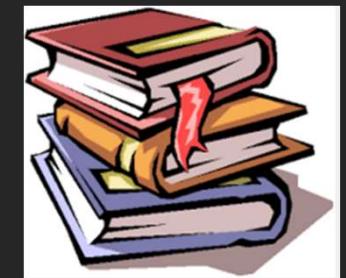


D Ú V I D A S

## Referência bibliográficas

### BIBLIOGRAFIA :

- MOLINARI, Leonardo. Testes de Software – Produzindo Sistemas Melhores e Mais Confiáveis, 4a. Edição. Editora Erica, 2013.
- MOLINARI, Leonardo. Inovação e Automação de Testes de Software, 1ª edição. Érica, 2010.
- CMMi V3. SEI - Software Engineering Institute., USA, 2007. Disponível na biblioteca online da Carnegie Mellon University.
- Reis, Luís Filipe Souza. ISO 9000/Auditorias de sistemas da qualidade. Editora: Érica, 1995.



## TESTE DE SOFTWARE

**Continua na próxima aula...**

PROFESSOR:

**RENATO JARDIM PARDUCCI**

**PROFRENATO.PARDUCCI@FIAP.COM.BR**