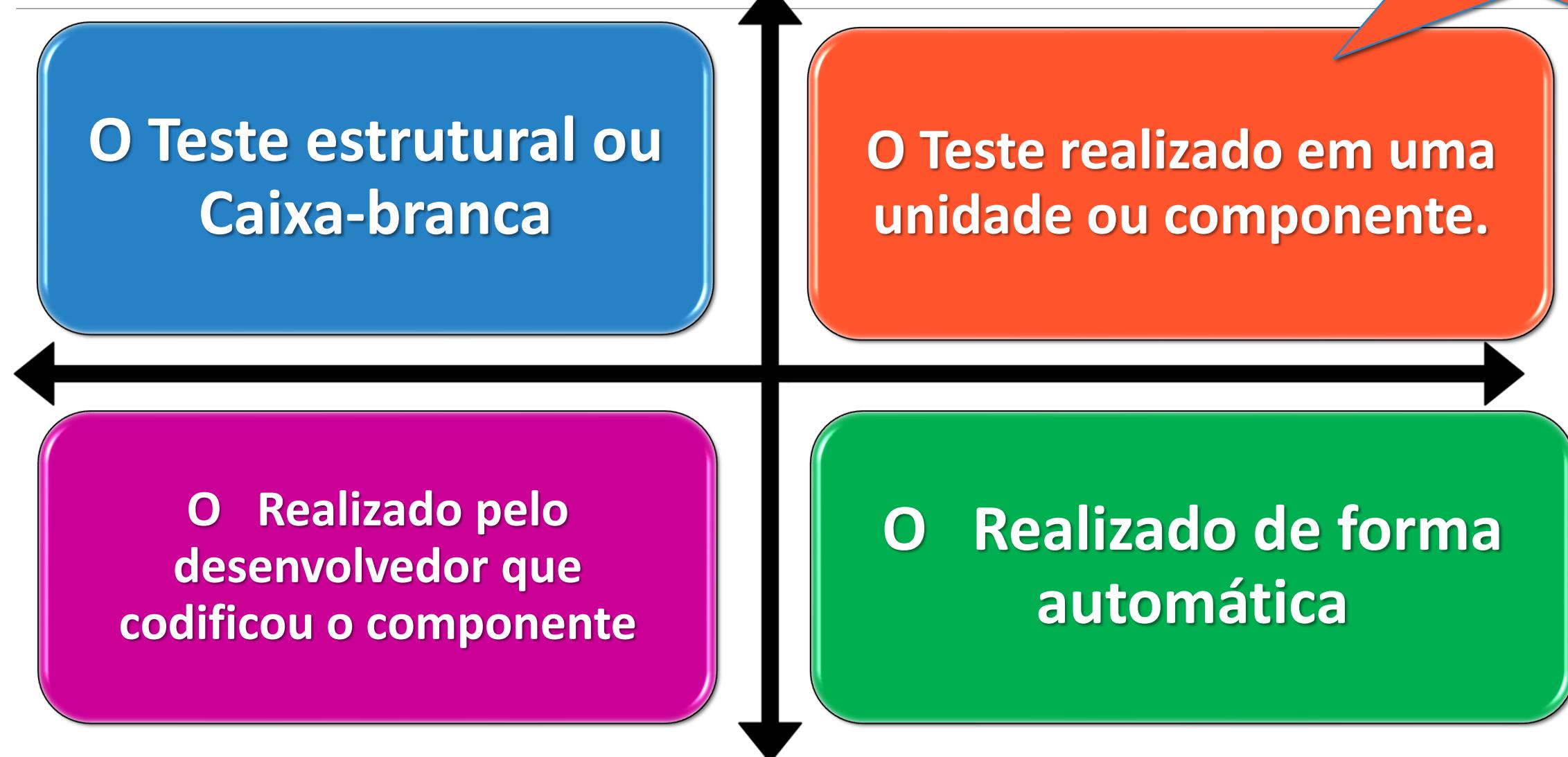


Tipos e Níveis dos Testes de Software

Professor Alann Perini

TESTES DE UNIDADE



Ex.: Teste para uma classe ou métodos do sistema.

Níveis de Teste de Software

❖ Exemplo de Teste de Unidade

The screenshot shows the Eclipse IDE interface with the JUnit perspective active. On the left, the Package Explorer view displays a tree of test classes under the package `test`. A red arrow points to the `testDivisao` method in the `test.CalculadoraTest` class, which failed with an `java.lang.ArithmetricException: / by zero`. The center of the screen shows the code editor with `CalculadoraTest.java` and `Calculadora.java`. The `testDivisao` method is annotated with `@Test` and contains assertions for addition and subtraction. The status bar at the bottom indicates the test was finished after 0.069 seconds with 4 runs, 1 error, and 0 failures. A red callout bubble on the right side of the code editor states: "Ferramenta usada para testes de java".

Package Explorer Hierarchy JUnit

Finished after 0,069 seconds

Runs: 4/4 Errors: 1 Failures: 0

`test.CalculadoraTest [Runner: JUnit 4] (0,018 s)`

- `testAdicao (0,003 s)`
- `testSubtracao (0,002 s)`
- `testMultiplicacao (0,003 s)`
- `testDivisao (0,010 s)`

`Failure Trace`

`java.lang.ArithmetricException: / by zero`

`at code.Calculadora.divisao(Calculadora.java:22)`

`at test.CalculadoraTest.testDivisao(CalculadoraTest.java:40)`

JUnit

Calculadora.java CalculadoraTest.java

```
package test;

import junit.framework.Assert;
import org.junit.Before;
import org.junit.Test;
import code.Calculadora;

public class CalculadoraTest {

    private Calculadora calculadora;

    @Before
    public void setUp() throws Exception {
        calculadora = new Calculadora();
    }

    @Test
    public void testAdicao() {
        Assert.assertEquals(1, calculadora.adicao(1, 0));
        Assert.assertEquals(-11, calculadora.adicao(-1, -10));
    }
}
```

Ferramenta usada para testes de java

TESTES DE INTEGRAÇÃO

TESTE ESTRUTURAL OU CAIXA-BRANCA

- ✓ Tem a finalidade de verificar se ao juntar vários componentes do sistema, eles se comunicam corretamente.
- ✓ A interface entre as unidades é testada.

TESTES DE INTEGRAÇÃO

Realizado pelos desenvolvedores ou analistas de sistema para testar um módulo do sistema.

Utiliza ‘Stubs’ para simular módulos que ainda não foram implementados, mas que se comunicam com o módulo a ser testado.

Realizado de forma automática.

TESTES DE SISTEMAS

TESTE FUNCIONAL OU CAIXA-PRETA.

- ✓ Tem por objetivo verificar se o sistema está em conformidade com a especificação de requisitos.
- ✓ Realizado pelo **testador**, o qual tem acesso apenas a interface do sistema.

TESTES DE SISTEMAS

O testador não faz parte da equipe de desenvolvimento.

Os testes geralmente são baseados em **roteiros de teste** criados a partir da **especificação**.

Pode ser realizado de forma **manual ou automática**.

Existem várias ferramentas, como: Selenium, Watir, Badboy, etc...

TESTES DE ACEITAÇÃO

Teste funcional ou Caixa-preta

Tem por objetivo verificar se o sistema está em conformidade com os requisitos esperados pelo cliente.

Realizado pelo **cliente** ou pelo **testador**, desde que este possua um **checklist** feito pelo cliente do que é esperado que haja no sistema.

Realizado no **ambiente de homologação**

TESTES DE ACEITAÇÃO

O sistema é utilizado para capacitação dos usuários de forma que eles validem todos os requisitos do sistema.

- ✓ Realizado de forma **manual** ou **automática**.
- ✓ **Exemplo:** Existe a ferramenta EasyAccept.

- ✓ Teste realizado pelo usuário pode ser:
 - ✓ **Teste Alfa:** em um ambiente de homologação.
 - ✓ **Teste Beta:** em um ambiente de produção.

Testes Exploratórios

TESTE FUNCIONAL OU CAIXA-PRETA

- ✓ Realizado por **testadores com experiência**.
- ✓ Quando não há muita documentação sobre o sistema.
- ✓ Realizado de forma **manual**.
- ✓ Os defeitos encontrados são reportados à medida que eles ocorrem.

Regressão



Testes de Regressão

TESTE FUNCIONAL OU ESTRUTURAL CAIXA-CINZA

- ✓ À medida que uma nova versão do sistema é liberada, novos bugs podem ser incluídos no sistema.
- ✓ Tem a finalidade de **realizar novamente testes** em um sistema já testado.
- ✓ Realizado pelo testador.
- ✓ Pode ser realizado de forma **manual ou automática**.

Testes de Cobertura

Teste funcional ou estrutural Caixa-cinza

- ✓ **Estrutural:** Tem a finalidade de identificar se os testes realizados no sistema abrangem pelo menos 95% do código produzido.
- ✓ **Funcional:** Os roteiros de teste abrangem 100% das funcionalidades do sistema, ou seja, possui pelo menos 1 caso de teste para cada regra de negócio.

Testes Funcionais x Testes de Unidade

FUNCIONAIS

TESTE DE CAIXA-PRETA

Manual ou Automático

Testador diferente do desenvolvedor faz os testes

Testador acessa o sistema via interface do usuário

Testes verificam se o sistema **faz** o que deve fazer e **não faz** o que não deve fazer

UNITÁRIO

TESTE DE CAIXA-BRANCA

Automático

Desenvolvedor faz os testes

Testador tem acesso ao código

Testes verificam a corretude de cada unidade (método, classe) de forma independente

A EQUIPE DE TESTES



A EQUIPE DE TESTES

GERENTE DE TESTES

Lidera a equipe de teste.

Faz a comunicação entre a equipe de teste e de desenvolvimento.

Planeja os testes, define estratégias, entre outros.



A EQUIPE DE TESTES

ARQUITETO DE TESTES

Conhece os requisitos do sistema.

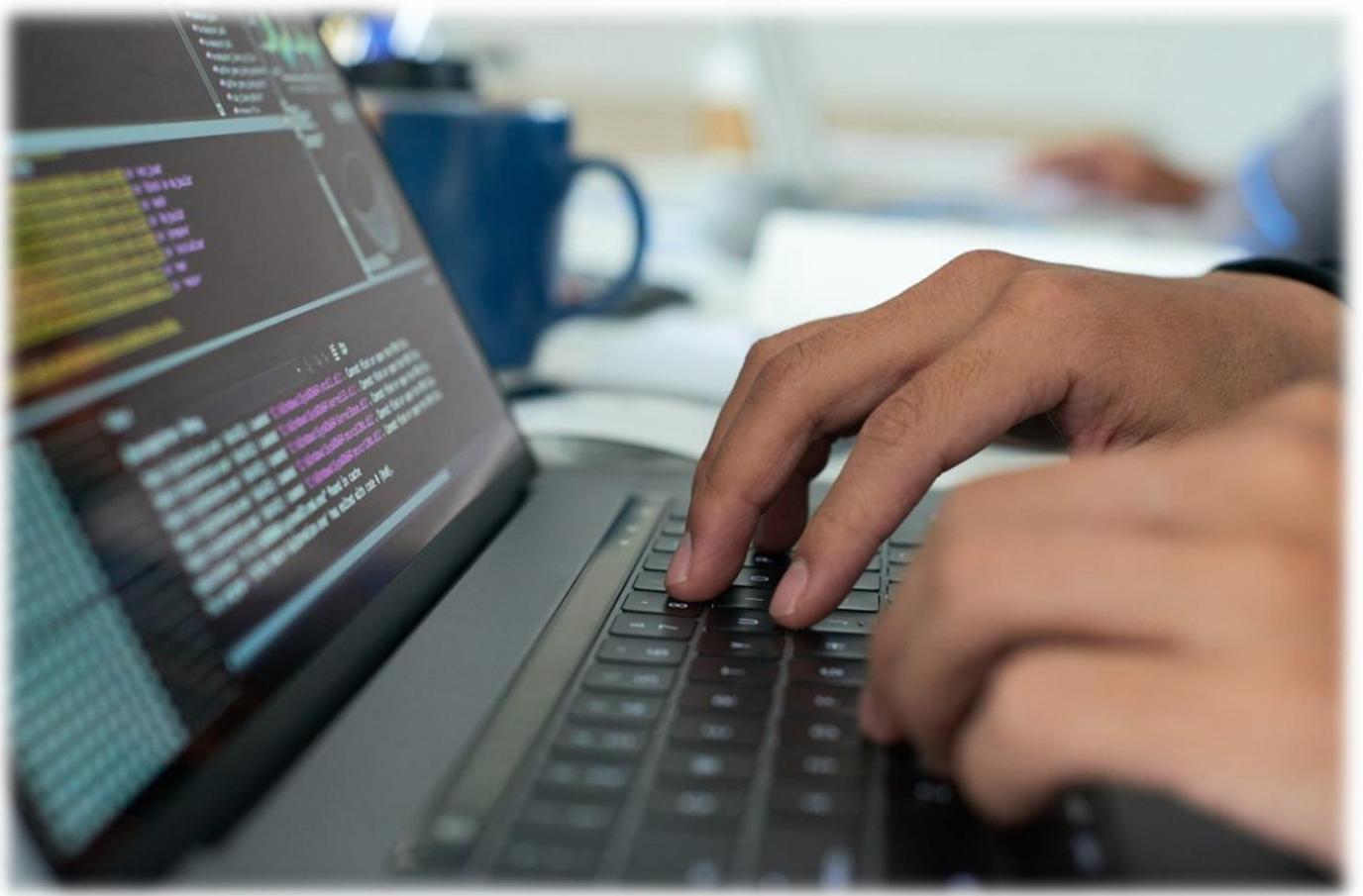
Elabora os roteiros de teste.



A EQUIPE DE TESTES

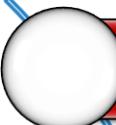
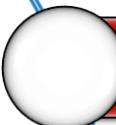
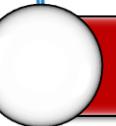
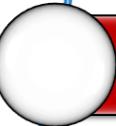
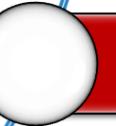
O TESTADOR

- ✓ É criativo ao executar os testes.
- ✓ Tem noções de programação.
- ✓ É objetivo ao descrever um erro.
- ✓ É perfeccionista.



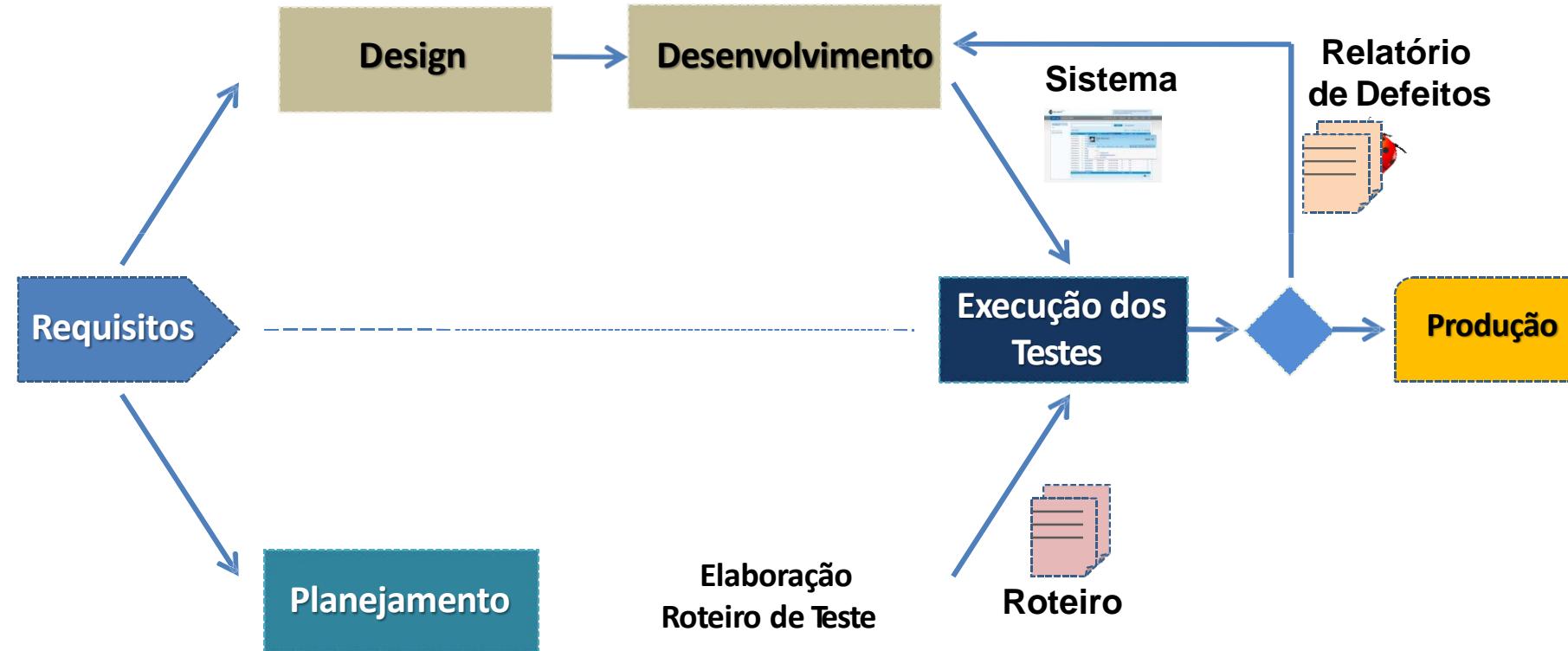
O TESTADOR



-  Não deve testar seu próprio programa.
-  Não deve duvidar que um erro existe.
-  Deve ter cuidado para não reportar falsos bugs.
-  O testador não é inimigo do desenvolvedor.
-  O testador deve saber se comunicar com o desenvolvedor.
-  Os bugs descritos por ele devem ser baseados em fatos.
-  Um bom testador é aquele que encontra muitos bugs!

Processo de Testes x Processo de Desenvolvimento

Processo de Desenvolvimento



Processo de Testes

Quando usar ferramentas de teste de software?

Quando há apenas 1 testador para o projeto

- Deve realizar apenas testes manuais.
- Não há necessidade de ferramentas para criar os roteiros de teste.
- Ferramenta apenas para reportar os defeitos. (**Redmine**)

Quando há uma equipe de teste

- Ferramenta para gerenciar a equipe. (**Redmine**)
- Ferramenta para criar os roteiros de teste. (**TestLink**)
- Ferramenta para reportar e gerenciar os defeitos. (**Redmine**)
- Automatizar os testes para auxiliar nos testes de regressão. (**Selenium**)

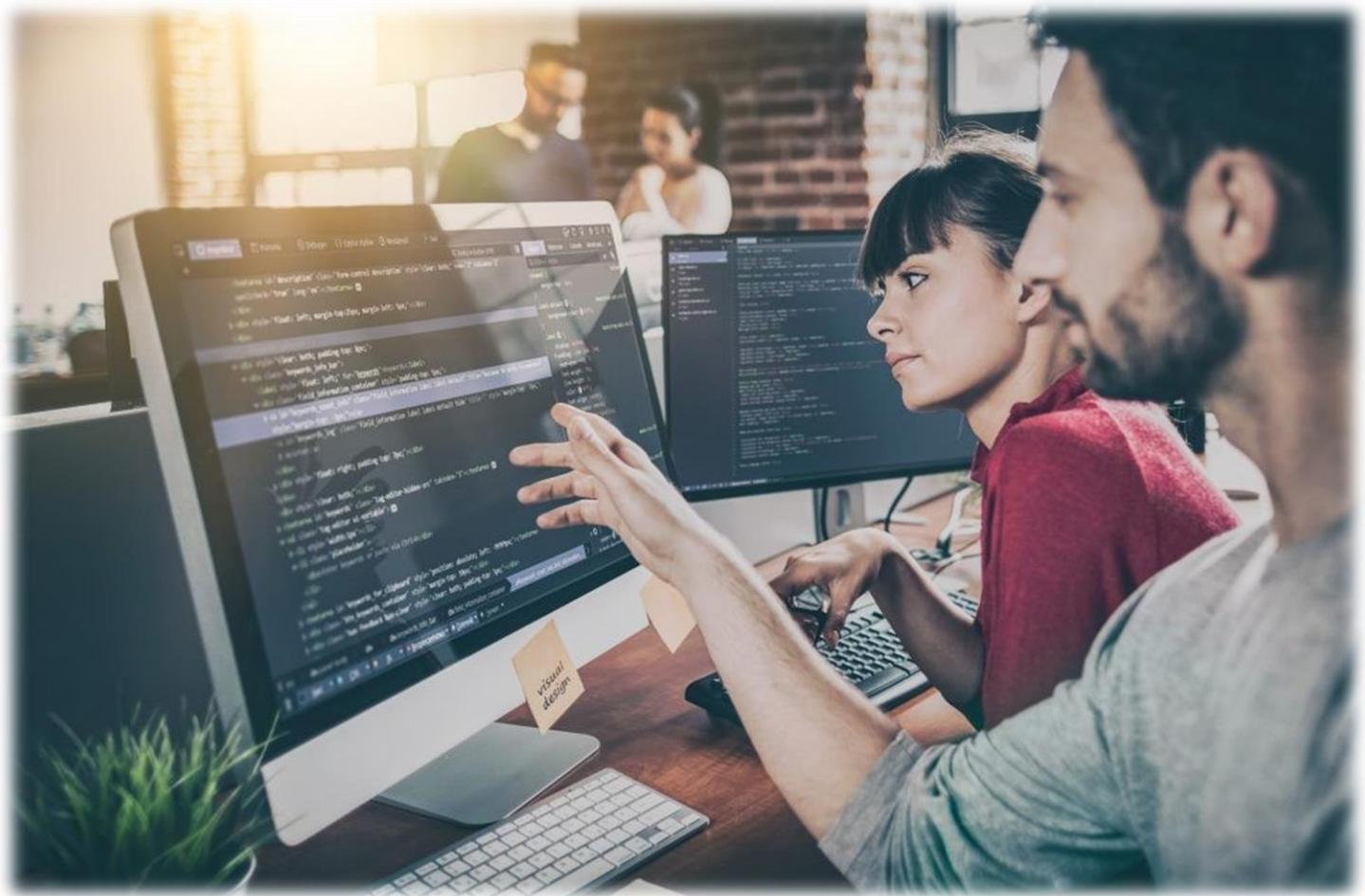
Quando os testes devem ser automatizados?



- ✓ Frequência de execução dos testes.
- ✓ As funcionalidades são testadas mais de uma vez.
- ✓ Baixo esforço para automatizar (equipe experiente).
- ✓ Ferramentas de automação relevantes para a realidade do projeto.
- ✓ Dificuldade de executar os testes manualmente.

IMPLANTANDO TESTES DE SOFTWARE

- ✓ Montar uma equipe de teste.
- ✓ O tamanho da equipe depende da quantidade de projetos a serem testados.
- ✓ Qualificar a equipe de testes com treinamentos.



IMPLANTANDO TESTES DE SOFTWARE

Quando a equipe de teste é pequena e precisa testar 2 projetos. Então, deve-se planejar a entrega de cada projeto em datas diferentes.



IMPLANTANDO TESTES DE SOFTWARE

Deve-se sempre iniciar com testes manuais.

Existem ferramentas que nos ajudam a gerenciar defeitos e a gerenciar testes em geral, veja exemplos abaixo:

Redmine

The screenshot shows the Redmine homepage with a dark blue header containing navigation links: Visão geral, Download, Atividade, Planejamento, Tarefas. Below the header, there's a section titled "Redmine" with a brief introduction and a "Features" section listing various project management features like multiple projects support, flexible access control, and issue tracking.

TestLink

TestLink 1.8.5 : caroline [admin]

The screenshot shows the TestLink interface with a title bar "TestLink 1.8.5 : caroline [admin]" and a menu bar with links like Início, Especificação, Executar Testes, Resultados, and Usuários. The main area displays a hierarchical tree of a test plan titled "Plano de Teste : UC001 - Cad Categoria de Despesa [Baseline : UC001-CadCategDesp - Exec.001]". The tree structure includes categories like "SIPAC-Orçamento / UC001 - Cad Categoria de Despesa", "Aba Cadastros", "Tela Dados da Categoria de Despesa", and "Layout da Página", with numerous detailed test cases listed under each category.

IMPLANTANDO TESTES DE SOFTWARE

Sistemas devem possuir uma **especificação básica** necessária para criar os roteiros de teste e um protótipo de cada tela.

Os **protótipos** de tela devem ser validados pelo cliente antes do início do desenvolvimento.

Especificação

UC01. Consulta do local de prova

1. Descrição
Um candidato inscrito no vestibular deverá poder consultar o local de prova na data pré-determinada.

2. Atores
Candidatos inscritos para o vestibular

3. Pré-condições
1. Computador com acesso a internet.
2. Sistema disponível.
3. Usuário ter conseguido efetuar o login com sucesso.
4. O período para consulta do local de prova esteja vigente.

4. Fluxo de Eventos
4.1. Fluxo Principal
1. O usuário se loga no sistema.
2. O usuário visualiza a página principal do sistema com os links para os locais de prova dos concursos que ele se inscreveu. [A.01] [E.01]
3. O usuário clica no link do concurso desejado a obter a informação.
4. O usuário será redirecionado para uma página que mostra as informações do concurso selecionado, do candidato e do local de prova (Consultar wireframe referente a local de prova) e uma mensagem informando o que é permitido levar no dia da prova [E.02] [E.03]
5. O usuário visualizará um botão no qual ele poderá imprimir a página. [E.04]

Protótipo

The wireframe shows a user profile page for the COPERVE (Comissão Permanente do Concurso Vestibular) system, developed by the UFPB (Universidade Federal da Paraíba). The page includes the COPERVE logo, the UFPB logo, and navigation links for Início, Perfil, Concursos, and Sair. The main content area displays personal information (Nome da Pessoa, identidade - uf, Logradouro, Número - Bairro, Cidade - UF), a placeholder for an image, and a 'Mensagens ao Usuário' section. Below this, a 'Dados Pessoais' section lists fields for Nome da Pessoa, CPF, Identidade, Data de Nascimento, E-mail, Telefone 1, Telefone 2, and Telefone 3. An 'Endereço' section lists fields for Logradouro, Número, Complemento, Bairro, Cidade, and Estado. A 'Você está aqui' breadcrumb trail shows the path: Coperve > Página Pessoal > Perfil. The footer contains links for 2010 - Núcleo de Tecnologia da Informação, NTI, Contato, Ajude-nos a melhorar, and Mapa do Site.



CódigoFonte



CódigoFonte

SÍNTESE FINAL

Software com testes melhoram a credibilidade do setor de informática

O Usuário mais satisfeito

Desenvolvedor perderá menos tempo resolvendo bugs de sistemas em produção, enquanto está alocado em outro projeto

O Desenvolvedor mais satisfeito e motivado

Sistema só deve ser colocado em produção após aprovação da equipe de testes.

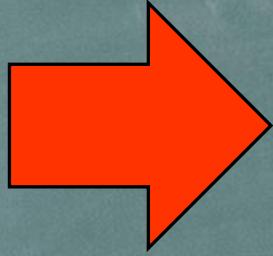
SÍNTESE FINAL

A equipe de teste é parte da equipe de desenvolvimento.

Cronogramas devem levar em consideração os testes.

Processo de Teste deve ser integrado ao processo de desenvolvimento

Testar reduz riscos do negócio



Faça uma pesquisa em “TRIO” relatando 3 FALHAS DE SOFTWARE QUE CAUSARAM CATÁSTROFES e foram motivo de morte ou prejuízo considerável.

- 
- Apresente exemplos, comente o motivo da falha e o que aconteceu após a falha.
 - Mostre vídeos ou imagens sobre a falha e o problema que a mesma causou.
 - Sugira com base no estudo realizado em sala, que tipo de teste a equipe recomendaria para evitar este tipo de problema
 - Cada grupo deverá apresentar falhas diferentes umas das outras.

Identificando Falhas de Software

PONTOS A SEREM AVALIADOS NA APRESENTAÇÃO:

Os slides, vídeos e imagens devem obrigatoriamente mostrar o motivo da falha e os resultados;

Os textos e o layout da apresentação deverão estar visíveis para todos os alunos (nada de textos minúsculos);

O grupo que está apresentando deve manter os alunos focados.