



O QUE SÃO TESTES?

Paulyne Jucá (paulyne@ufc.br)

RELAÇÃO DE TESTES COM REQUISITOS

- Requisito funcional: O filme cadastrado deve pertencer a uma categoria previamente determinada.
 - Problema?
- Requisito não-funcional (desempenho): Ao registrar um item sendo vendido, a descrição e preço devem aparecer em tempo aceitável.
 - Problema?



RELAÇÃO DE TESTES COM REQUISITOS

- Requisito: O filme cadastrado deve pertencer a uma categoria previamente determinada.
 - Problema: quais são as categorias possíveis?
 - Solução: O cliente fornece o conjunto de categorias durante a especificação. Durante a implementação o desenvolvedor já faz obriga que o filme tenha uma categoria (não podendo cadastrar com valor nulo) e testa em código se o valor é um dos valores previamente cadastrados.
- Requisito não-funcional (desempenho): Ao registrar um item sendo vendido, a descrição e preço devem aparecer em tempo aceitável.
 - Problema: O que é um tempo aceitável.
 - Solução: Reescrever o requisito determinando esse tempo aceitável. Planejar a arquitetura e desenvolver o sistema de forma a atender esse requisito.



VERIFICAÇÃO X VALIDAÇÃO

○ Validação

- Estamos construindo o software certo?
 - Quando estiver pronto é esse o software que nosso cliente/usuário espera?

○ Verificação

- Estamos construindo o software da maneira correta?
 - O software vai funcionar corretamente?



VERIFICAÇÃO ESTÁTICA E DINÂMICA

Inspeções de software - preocupadas com a análise estática das representações do sistema para descobrir problemas (verificação estática))

pode ser complementadas por alguma análise automática do código fonte de um sistema ou dos documentos associados.

Teste de software - preocupado com a execução e observação do comportamento do produto (verificação dinâmica).

O sistema é executado com dados de teste e o seu comportamento operacional é observado.

INSPEÇÕES X TESTES

- Inspeções e testes são técnicas de verificação complementares e ambas devem ser utilizadas no processo de V&V.
- Inspeções podem verificar a conformidade com uma especificação mas não a conformidade com os reais requisitos do usuário.
- Inspeções não podem verificar as características não funcionais tais como desempenho, usabilidade, etc.



INSPEÇÕES DE CÓDIGO

- Revisão cuidadosa, linha por linha, do código fonte do programa.
- Objetivo é a DETECÇÃO de defeitos (não correção)
- Defeitos podem ser erros lógicos, anomalias no código que podem indicar uma condição errônea (por ex. uma variável não inicializada) ou a não conformidade com padrões organizacionais.



INSPEÇÃO DE CÓDIGO

```
public void Eco (String frase) {  
    System.out.println(frase.length());  
}
```



INSPEÇÃO DE CÓDIGO

```
public void Eco (String frase) {  
    System.out.println(frase.length());  
}
```

E se frase for nula? Na inspeção a equipe de desenvolvimento pode identificar esse defeito e arrumar o código.



TESTE

- De software:

- Um conjunto de atividades e técnicas relacionadas à procura de erros em um programa.

- De um componente:

- “A set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement” [IEEE, do178b]



OBJETIVOS DO TESTE

- Teste de defeito (dirty/negative test)
 - mostrar a presença de erros
- Teste de validação ou conformidade (clean/positive test)
 - mostrar que o software opera corretamente

O número de ações válidas é finito, porém o número de ações inválidas é normalmente infinito



ABORDAGENS DE TESTE

Caixa branca (estrutural)

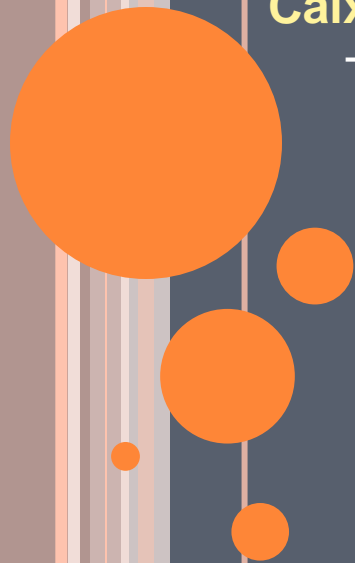
Realizado a partir do conhecimento de detalhes da implementação.

Caixa preta (funcional)

Os testes são planejados a partir de uma especificação abstrata. A implementação é desconhecida.

Caixa cinza (híbrido)

Testes de caixa preta com conhecimento limitado sobre a implementação.



TESTES X DEPURAÇÃO

- Teste e depuração de um programa são atividades complementares
- Os testes verificam a existência de defeitos (**falhas**) em um software
- Através da depuração é possível localizar e remover **faltas**.



FALHA X FALTA

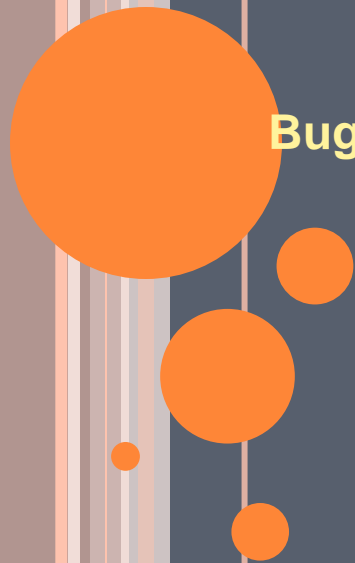
Falta ou defeito (fault)

Código incorreto ou ausente que, quando executado, pode resultar em uma falha.

Falha ou erro (failure)

Manifestação de um problema através de uma saída incorreta ou o término anormal do programa, por exemplo.

Bug: termo geral para faltas ou falhas



LOCALIZE A FALHA E A FALTA

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

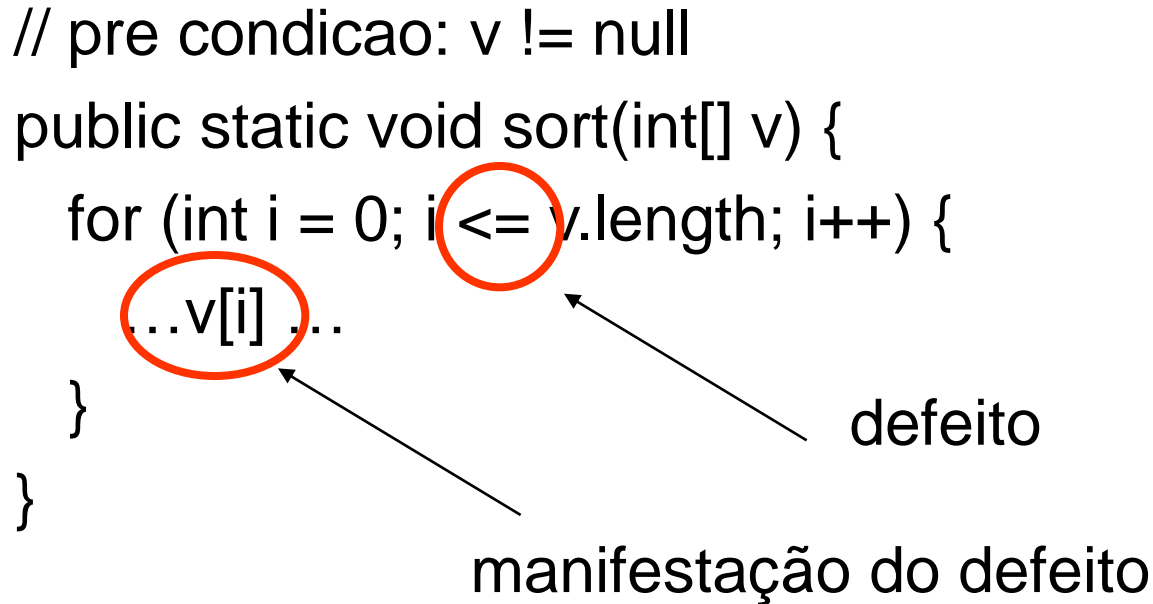


LOCALIZE A FALHA E A FALTA

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

defeito

manifestação do defeito



SUÍTE DE TESTE

- Conjunto de testes de um sistema
- Ex: suíte de testes dos middlewares de celular e TV digital



TESTE DE REGRESSÃO

- Testes feitos em partes do sistema que já foram testados anteriormente para garantir que eles ainda funcionam e que não foram impactados por mudanças realizadas em outros pontos do sistema.
- Pode ser feita regularmente através de uma suíte de testes
- Testes que são refeitos constantemente são candidatos à automação.



OUTROS CONCEITOS

- Domínio de entrada: conjunto de todos os valores de entrada possíveis para o programa
- Dado de teste: um elemento do domínio de entrada usado em uma execução do teste do programa
- Caso de Teste: Especifica o que se quer testar: estado inicial, condições e entradas de teste e resultados esperados. Resultados esperados incluem: mensagens geradas pela implementação, exceções, valores retornados, e estado resultado da implementação e seu ambiente.



OUTROS CONCEITOS

- Conjunto de teste ou conjunto de casos de teste (bateria de testes) : conjunto dos casos de teste usados durante o teste de um programa.
- Teste exploratório: teste realizado sem um caso de teste definido.
- Omissão: funcionalidade requisitada e não presente no software.
- Surpresa: funcionalidade não requisitada e presente no software. Pode ser bom ou ruim



OBJETIVOS DA V&V

- Estabelecer a confiança de que o software é “adequado a seu propósito”.
- Isso NÃO significa que o programa tenha que ser livre de defeitos.
- Ao invés disso, significa que o sistema deve ser suficientemente bom para o uso pretendido. O tipo de uso irá determinar o grau de confiança que será necessário.
 - Esse nível de confiança é acordado com o cliente antes da implementação e confirmado na fase de entrega do software. (exemplo de SLA)



CONFIANÇA DE V&V

- Depende do propósito do sistema, as expectativas do usuário e o ambiente de mercado.
- Função do software
 - O nível de confiança depende do tipo de sistema e o quanto é importante para a organização.
- Expectativas do usuário
 - Usuários podem ter poucas expectativas de certos tipos de software
- Ambiente de mercado
 - Colocar um produto no mercado pode ser mais importante do que encontrar todos os defeitos no programa.

