Testes automatizados

08/08/2025

Na Aula de hoje

- Definições: Teste de aceitação, Teste Alfa, Teste beta e Teste de regressão.
- Teste ágil

Recapitulando Dívidas

- Testes que podem ter resultados difíceis de prever: Web Scrapping, uso de APIs públicas ou dados externos; sorteios aleatórios; simulações;
- Exemplos de drivers e stubs:

Recapitulando Dívidas

Exemplo de stup:

```
// Função a ser testada
function registrarUsuario(usuario, servicoEmail) {
  // Salva o usuário no banco
  salvarNoBanco(usuario);
  // Envia e-mail de boas-vindas
  servicoEmail.enviarEmail(usuario.email, "Bem-vindo!");
}
```

```
const stubServicoEmail = {
  enviarEmail: (email, mensagem) => {
    console.log(`Stub: e-mail para ${email} simulado.`);
  }
};

// Teste
registrarUsuario(
    { nome: "João", email: "joao@email.com" },
    stubServicoEmail
);
```

Recapitulando Dívidas

Exemplo de Driver: Vamos dizer que você tem um sensor que precisa ser testado, mas ele ainda não esta disponível.

```
class SensorListener {
  constructor(limite) {
    this.limite = limite;
  }

  onData(temperatura) {
    if (temperatura > this.limite) {
      console.log("Alerta! Temperatura muito alta.");
    }
  }
}
```

Recapitulando Dívidas

Você pode simular / mockar o comportamento do sensor para realizar os testes

```
function simularSensor(sensorListener) {
  const leituras = [25, 28, 32, 29];
  leituras.forEach(temp => {
    console.log(`Simulando leitura: ${temp}°C`);
    sensorListener.onData(temp);
  });
}
//...

const listener = new SensorListener(30);
simularSensor(listener);
```

Tipos de Teste

A técnica de teste é definida pelo tipo de informação utilizada para realizar o teste.

Técnica funcional - os testes são baseados exclusivamente na especificação de requisitos do programa. Nenhum conhecimento de como o programa está implementado é requerido.

Técnica estrutural- os testes são baseados na estrutura interna do programa, ou seja, na implementação do mesmo.

Técnica baseada em defeito - os testes são baseados em informações histórica sobre defeitos cometidos frequentemente durante o processo de desenvolvimento de software.

Entradas

Geralmente identificadas como dados fornecidos via teclado para o programa executar.

Entretanto, os dados de entrada podem ser fornecidos por outros meios, tais como:

- Dados oriundos de outro sistema que servem de entrada para o programa em teste;
- Dados fornecidos por outro dispositivo; Dados lidos de arquivos ou banco de dados;
- O estado do sistema quando os dados são recebidos; O ambiente no qual o programa está executando.

Saídas

As saídas também podem ser produzidas de diferentes formas.
A mais comum é aquele apresentada na tela do computador.
Além dessa, as saídas podem ser enviadas para:

- Outro sistema interagindo com o programa em teste; Dados enviados para um dispositivo externo;
- Dados escritos em arquivos ou banco de dados;
- Estado do sistema ou o ambiente de execução podem ser alterados durante a execução do Programa.

Testes de Sistema

Apropriado para testar requisitos não funcionais do software:

- Velocidade

- Confiabilidade
- Segurança

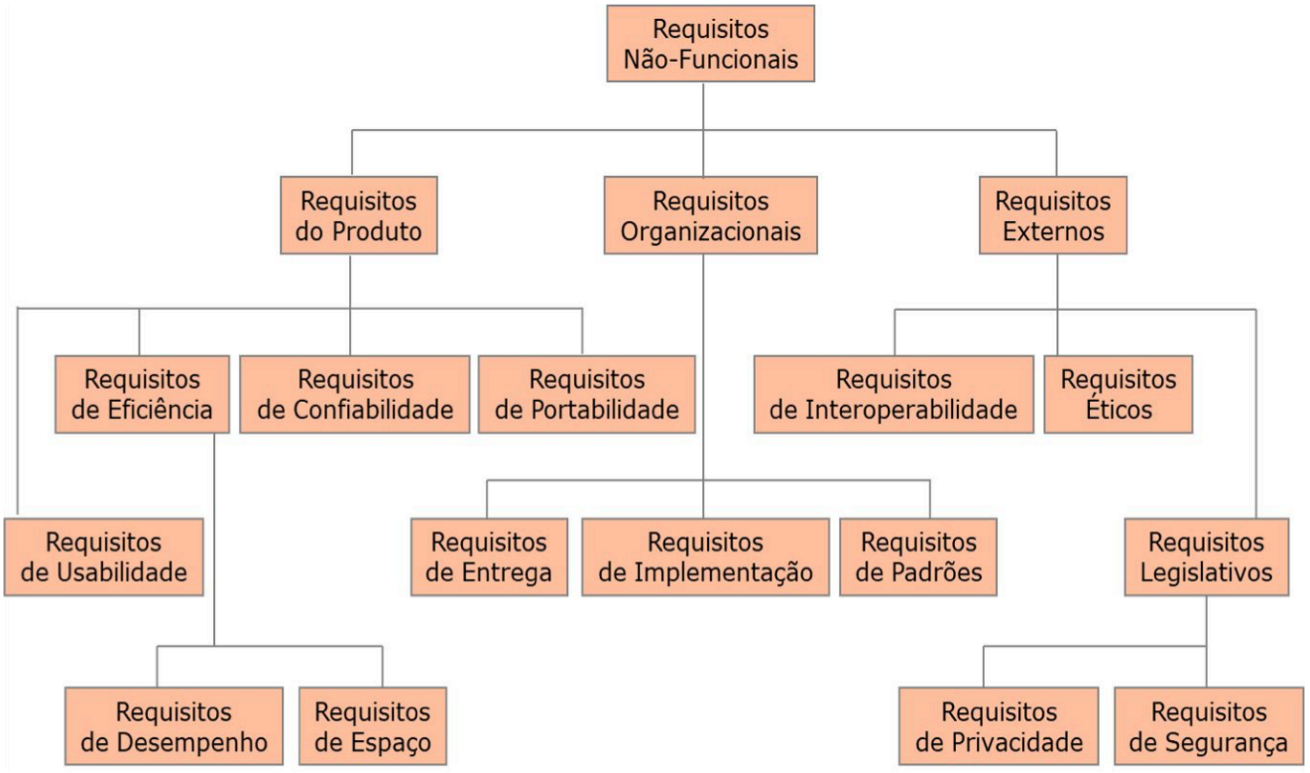
Testes de Sistema

Cada técnica de teste possui um conjunto de critérios de teste.

Os critérios sistematizam a forma como requisitos de teste devem ser produzidos a partir da fonte de informação disponível (especificação de requisitos, código fonte, histórico de defeitos, dentre outras).

- Os requisitos de teste são utilizados para: Gerar casos de teste.
  - Avaliar a qualidade de um conjunto de teste existente.

Requisitos não funcionais



Testes de aceitação

- Objetivo é verificar se o programa desenvolvido atende as exigências do usuário

---

Teste Alfa

- Realizado pelo usuário no ambiente do desenvolvedor;
- O desenvolvedor tem controle sobre o ambiente de teste;
- Monitora as ações do usuário registrando falhas e problemas de uso.

Teste Beta

- Realizado pelo usuário no seu ambiente;
- O desenvolvedor não tem controle sobre o ambiente utilizado nos testes;
- diferencial é a grande diversidade de ambientes e configurações de software e hardware.

Teste de Regressão

- Mesmo após liberado o software precisa ser testado
- A cada manutenção é preciso verificar se o software mantém suas características
- Se nenhum efeito colateral foi introduzido
- Técnicas de teste de regressão reutilizam subconjuntos do conjunto de teste existente

---

Testes Ágeis

Baseado nas metodologias ágeis, a ideia é fazer entregas frequentes de versões do software durante o desenvolvimento para testes para pelos clientes, garantindo um software mais correto

Focar nas funcionalidades mais importantes e de alto risco primeiro. Depois expandir o escopo dos testes para cobrir outras áreas. Como resultado, eles entregam recursos valiosos aos clientes em pouco tempo

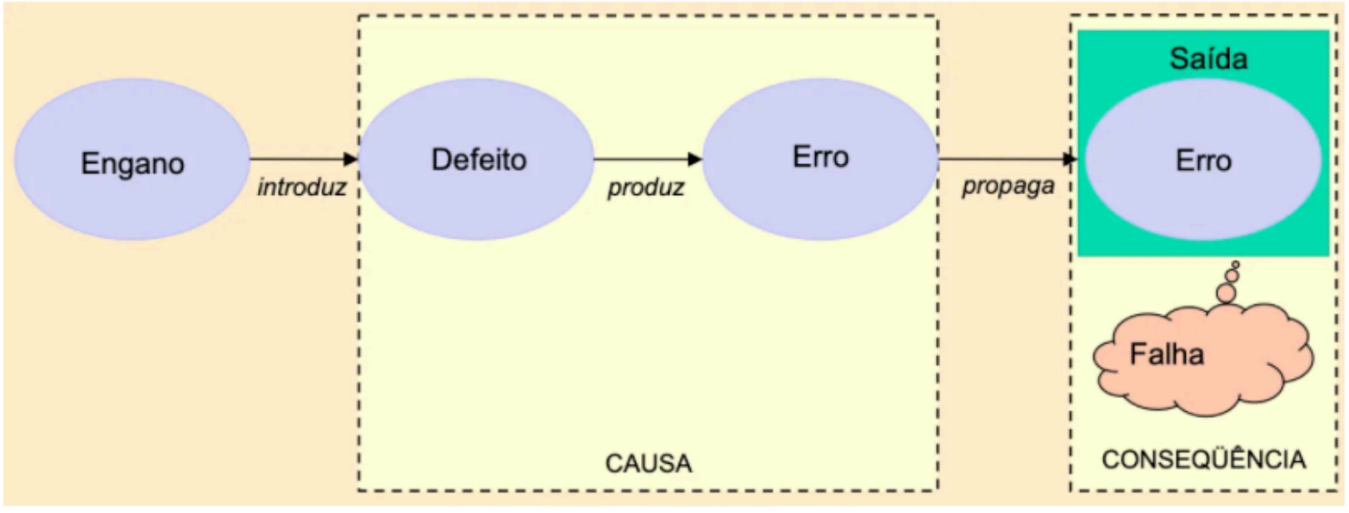
- Testes em CI/CD
- Dividir o desenvolvimento de software em pequenas iterações, concentrando-se na entrega das funcionalidades mais críticas primeiro

Fonte: Miro - O que é Agile Testing

Defeito, Erro e Falha

- **Defeito**: uma incorreção estática no código.
- **Erro**: um estado interno incorreto, que é consequência da execução de um defeito.
- **Falha**: comportamento externo incorreto, com respeito aos requisitos ou qualquer outra descrição do comportamento esperado.

Defeito, Erro e Falha



Para uma falha ocorrer (RIP)

- O ponto do programa que contém um defeito deve ser executado (alcançabilidade).
- Após a execução deste ponto, o estado da execução deve ser incorreto (infecção).
- O estado infectado deve se propagar de modo a produzir uma saída incorreta (propagação).

RIP (Reachability, Infection, Propagation).

> **Edsger W. Dijkstra**
>
> Testes de software mostram a presença de bugs, mas não a sua ausência.

Atividade

Em uma universidade, os alunos recebem conceito A se tiverem nota maior ou igual a 90. A seguinte função foi usada, em um dos sistemas da universidade para implementar esse requisito:

```
function isConceitoA(nota: number) -> boolean {
    if(nota > 90){
        return true
    }
    return false;
}
```

(a) Essa implementação possui um bug ? Se sim, quando esse bug resulta em falha?

(b) Suponha que a função seja testada com as notas 95 e 85. Qual a cobertura de comandos do teste? E a de branches?