**Compliance & Quality Assurance**

**Prof. M.Sc. Felipe Desiglo Ferrare**  
proffelipe.ferrare@fiap.com.br

**Introdução aos Testes de Software**

**Aula 2**

**Cronograma**

* Recap da última aula
* Conceitos
* Terminologia
* Psicologia do Teste
* Regra 10 de Myers
* Modelo V
* Níveis de Teste
* Pirâmide de Testes

**Nos Últimos Capítulos (Última Aula)**

* **O Que é Qualidade**
  + Depende do contexto (expectativa)
  + É de responsabilidade de todos
  + Testes servem para mitigar riscos
  + Qualidade de processo VS qualidade de produto
  + Funcional VS Não Funcional
  + Certificações (ISTQB, ASQ)

**Conceitos Básicos**

**Terminologia**

* **Teste Estático:** Atividade de teste que contempla apenas a revisão (manual ou automática) de artefatos de teste ou do próprio código, porém sem executar o software.
  + **Tipos de testes estáticos:**
    - Revisão
    - Acompanhamento (Walkthrough)
    - Inspeção
  + **Exemplos de produtos testados estaticamente:**
    - Requisitos e Casos de Uso
    - Arquitetura & Design
    - Código fonte
    - Manual do usuário
* **Teste Dinâmico:** Atividade de teste que envolve a execução do software, fornecendo entradas e avaliando as saídas e o comportamento apresentado.
  + **Testes dinâmicos podem ser:**
    - Funcionais
    - Não-funcionais
* **Caso de Teste:** Conjunto de pré-condições, procedimentos e resultados esperados usado pelo testador para determinar se o sistema satisfaz o requisito ou funciona corretamente.
* **Suíte de Teste:** Conjunto de casos de teste, organizados em uma ordem lógica, que devem ser executados em uma dada atividade de teste.
* **Plano de Teste:** Documento que descreve o escopo, a abordagem, recursos necessários e cronograma previstos para as atividades de teste do software. Também indica as funções que serão testadas, quem executará as tarefas, o ambiente a ser usado, motivações das escolhas, riscos identificados e planos de contingência.
* **Erro:** Também chamado de engano (mistake), é uma ação humana que produz um resultado incorreto.
* **Defeito:** Também chamado de falta ou bug, é uma imperfeição ou deficiência em um produto de trabalho (código ou outro) causada por um erro.
* **Falha:** Evento causado por um defeito no qual um sistema, ou parte dele, não executa uma função conforme os requisitos estabelecidos.

**Defeitos, Causa-Raiz e Efeitos**

* **Causas-raiz de um defeito:** São as primeiras ações ou condições que contribuíram para o surgimento desse defeito.
* **Análise de causa-raiz (Root-cause Analysis – RCA):** Atividade de investigar profundamente o defeito para identificar suas causas-raízes, de modo que possam ser implementadas ações de melhoria que evitem que tais erros e defeitos voltem a se repetir no futuro.
* **Efeitos:** Consequências das falhas, como reclamações de clientes, perda de receita ou reputação, etc.

**Psicologia do Teste**

* **Desenvolvedor:** Objetivo de construir
* **QA:** Objetivo de manter a qualidade

**Objetivos podem ser conflitantes**

* Críticas podem ser vistas como pessoais
* Não perceber que a qualidade faz parte do processo de entrega

**Os 7 Princípios do Teste de Software**

1. **O teste mostra a presença de defeitos e não a sua ausência:** O teste reduz a probabilidade de defeitos não descobertos permanecerem no software, mas, mesmo se nenhum defeito for encontrado, o teste não é uma prova de correção.
2. **Testes exaustivos são impossíveis:** Testar tudo (todas as combinações de entradas e pré-condições) não é viável, exceto em casos triviais. Em vez de tentar testar exaustivamente, a análise de risco, as técnicas de teste e as prioridades devem ser usadas para concentrar os esforços de teste.
3. **O teste inicial economiza tempo e dinheiro:** Para encontrar antecipadamente os defeitos, as atividades de teste estático e dinâmico devem iniciar o mais cedo possível no ciclo de vida de desenvolvimento de software. O teste inicial é por vezes referido como shift-left. O teste no início do ciclo de vida de desenvolvimento de software ajuda a reduzir ou eliminar alterações dispendiosas.
4. **Defeitos se agrupam:** Um pequeno número de módulos geralmente contém a maioria dos defeitos descobertos durante o teste de pré-lançamento ou é responsável pela maioria das falhas operacionais. Agrupamento de defeitos previstos e os agrupamentos de defeitos observados reais em teste ou produção, são uma entrada importante em uma análise de risco usada para focar o esforço de teste.
5. **Cuidado com o paradoxo do pesticida:** Se os mesmos testes forem repetidos várias vezes, esses testes não encontrarão novos defeitos. Para detectar novos defeitos, os testes existentes e os dados de teste podem precisar ser alterados e novos testes precisam ser gravados. Em alguns casos, como o teste de regressão automatizado, o paradoxo do pesticida tem um resultado benéfico, que é o número relativamente baixo de defeitos de regressão.
6. **O teste depende do contexto:** O teste é feito de forma diferente em diferentes contextos. Por exemplo, o software de controle industrial de segurança crítica é testado de forma diferente de um aplicativo móvel de comércio eletrônico. Como outro exemplo, o teste em um projeto ágil é feito de forma diferente do que o teste em um projeto de ciclo de vida sequencial.
7. **A ausência de erros é uma ilusão:** Algumas organizações esperam que os testadores possam executar todos os testes possíveis e encontrar todos os defeitos possíveis, mas os princípios 2 e 1, respectivamente, nos dizem que isso é impossível. Além disso, é uma ilusão esperar que apenas encontrar e corrigir muitos defeitos assegure o sucesso de um sistema. Testar exaustivamente todos os requisitos especificados e corrigir todos os defeitos encontrados ainda pode produzir um sistema difícil de usar, que não atenda às necessidades e expectativas dos usuários ou que seja inferior em comparação com outros sistemas concorrentes.

**A Regra 10 de Myers**

* **Prevenir defeitos**
* **Corrigir eles o quanto antes**

**Modelo V**

* **Validação:** “Estamos planejando o processo para atingir nosso objetivo?”
* **Verificação:** “Estamos construindo o produto certo?”

**Níveis de Teste**

* **Testes Unitários:** Consiste em escrever testes automatizados para testar pequenas unidades de código, geralmente funções ou métodos individuais, para garantir que eles funcionem como esperado. Testes unitários são geralmente escritos por desenvolvedores e executados durante o processo de desenvolvimento de software.
* **Testes de Integração:** Se concentram em validar as interações entre componentes (unidades de código) ou sistemas, garantindo que eles trabalhem em conjunto corretamente e que suas interações não introduzam nenhum defeito ou comportamento inesperado.
* **Testes de Sistema:** Se concentram no comportamento e nas capacidades de todo um sistema ou produto, geralmente considerando as execuções das tarefas de ponta a ponta do sistema e os comportamentos não-funcionais exibidos ao executar tais tarefas.
* **Testes de Aceitação:** Relacionados às necessidades do usuário, requisitos e processos de negócios, executados para determinar se um sistema satisfaz ou não os critérios de aceitação e para permitir que o usuário determine se aceita ou não a entrega.

**Pirâmide de Testes**

**Referências**

* **ISTQB CTFL Syllabus v4.0:** [Syllabus ISTQB](https://bcr.bstqb.org.br/docs/syllabus_ctfl_4.0br.pdf)
* [BSTQB](https://bstqb.org.br/)
* [ISTQB](https://www.istqb.org/)
* [ASQ](https://asq.org/)
* [Software Testing Fundamentals](https://softwaretestingfundamentals.com/)
* The Art of Software Testing - 3ª Edição

**Obrigado!**

Espero que este material seja útil para seus estudos!