# Teste e Qualidade de Software

Gustavo Henrique Massaro

# Objetivos:

- Entender o que é um processo de software e sua importância no desenvolvimento de produtos de qualidade.
- Explorar as principais fases do Ciclo de Vida do Processo de Software.
- Compreender as diferenças entre as abordagens tradicionais (cascata) e ágeis no Ciclo de Vida do Processo de Software.

# Introdução ao Processo de Software

- O que é um Processo de Software
- Um Processo de Software é uma abordagem sistemática e organizada para o desenvolvimento, implantação e manutenção de produtos de software.
- Consiste em um conjunto de atividades, métodos, práticas e ferramentas que são aplicados para produzir software de forma padronizada e eficiente.
- É como um roteiro que guia os profissionais de software ao longo do projeto, desde a concepção até a entrega do produto final.

# Importância do Processo de Software:

- Garantia de Qualidade: Um processo bem definido estabelece padrões que asseguram a qualidade do software produzido.
- Produtividade: Proporciona uma estrutura que torna o desenvolvimento mais eficiente e com menor chance de retrabalho.
- Controle e Gerenciamento: Permite que os gerentes monitorem o progresso do projeto e tomem decisões com base em informações confiáveis.
- Comunicação: Facilita a comunicação entre os membros da equipe, clientes e demais stakeholders, alinhando as expectativas.

# Benefícios do Uso de Processos Bem Definidos:

- Redução de Erros: Um processo estruturado ajuda a evitar erros comuns no desenvolvimento de software.
- Prazos e Custos: Ajuda a cumprir prazos estabelecidos e a manter os custos sob controle.
- Satisfação do Cliente: Contribui para a entrega de produtos que atendam às expectativas dos clientes.
- Melhoria Contínua: Permite a identificação de oportunidades de aprimoramento para projetos futuros.

# Exemplos de Modelos de Processo de Software

- Modelo Cascata (Waterfall): Abordagem sequencial, onde cada fase depende da conclusão da anterior.
- Modelo Incremental: Desenvolvimento em etapas, com entregas parciais e evolutivas.
- Modelo Ágil (Scrum, Kanban): Abordagem iterativa e incremental, adaptável a mudanças.

# A Evolução dos Processos de Software

- Abordagens Tradicionais vs. Abordagens Ágeis.
- Surgimento do Manifesto Ágil e suas diretrizes.

# Conclusão

- O Processo de Software é fundamental para o desenvolvimento de software eficiente e de qualidade. Uma abordagem organizada e padronizada proporciona benefícios significativos, incluindo a garantia de qualidade, produtividade e satisfação do cliente. Nos próximos tópicos, exploraremos as fases do Ciclo de Vida do Processo de Software e as diferenças entre as abordagens tradicionais e ágeis, aprofundando nosso entendimento sobre como desenvolver software de forma mais eficaz.

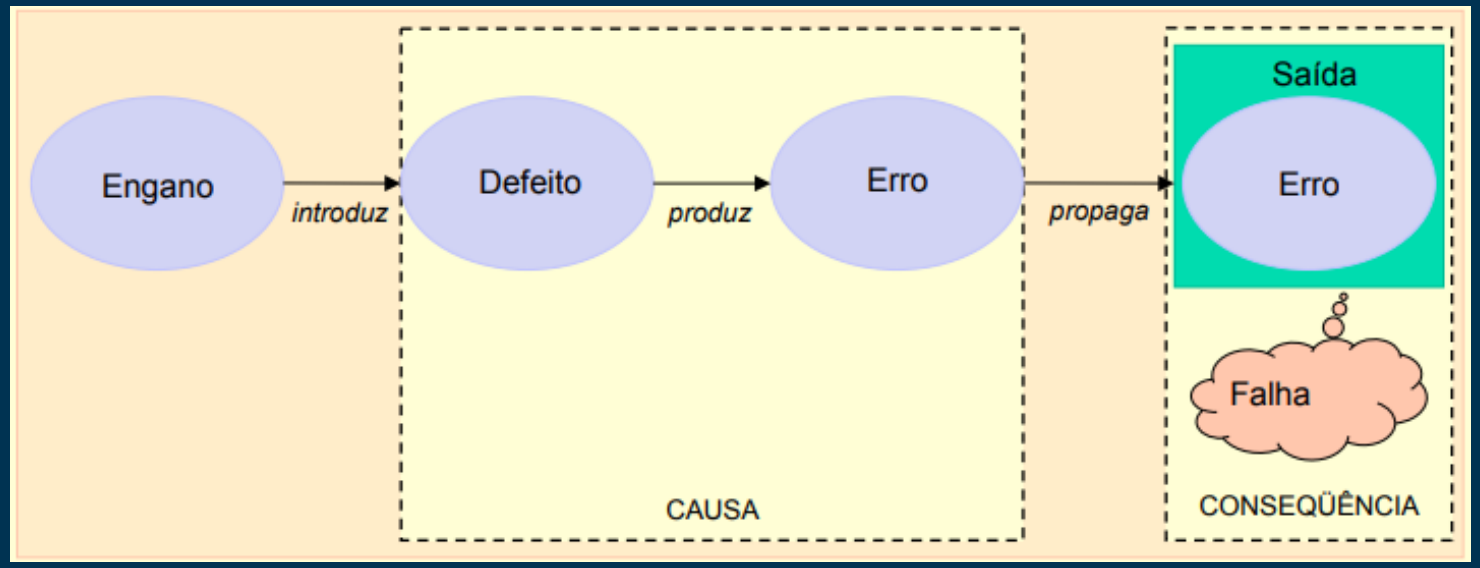
# Definição de Qualidade de Software

- A qualidade de software refere-se à capacidade de um produto de software atender aos requisitos e expectativas dos usuários de forma precisa, confiável e consistente.
- Vai além de um software funcional e envolve também aspectos como desempenho, segurança, usabilidade e manutenibilidade.

# Importância da Qualidade de Software:

- Software de baixa qualidade pode resultar em erros, falhas e insatisfação dos usuários, afetando a credibilidade da empresa ou equipe de desenvolvimento.
- A qualidade do software está diretamente relacionada à produtividade e ao sucesso do projeto, pois problemas não identificados precocemente podem levar a retrabalhos e atrasos.

- Defeito: deficiência mecânica ou algorítmica que, se ativada, pode levar a uma falha
- - Instrução ou comando incorreto
- Erro: item de informação ou estado de execução inconsistente
- Falha: evento notável em que o sistema viola suas especificações



# Desafios do teste

- Todos já testaram algum produto de software... Quais foram os maiores desafios?
- Alguns problemas comuns...
- - Não há tempo suficiente para o teste.
- - Muitas combinações de entrada para serem exercitadas.
- - Não há tempo para o teste exaustivo.
- - Dificuldade em determinar os resultados esperados para cada caso de teste.
- - Requisitos do software inexistentes ou que mudam rapidamente.
- - Não há treinamento no processo de teste.
- - Não há ferramenta de apoio.
- - Gerentes que desconhecem teste ou que não se preocupam com qualidade.

# Impacto de Software com Baixa Qualidade:

- Custo elevado de correções e manutenção.
- Perda de confiança dos clientes e usuários.
- Redução da competitividade no mercado.
- Riscos para a segurança da informação e privacidade dos dados.
- Possibilidade de violações de regulamentos e normas.

# Fatores que Influenciam a Qualidade de Software:

- Processo de desenvolvimento bem definido e estruturado.
- Testes de software abrangentes e eficazes.
- Boas práticas de codificação e design.
- Uso de padrões e normas reconhecidas pela indústria.
- Interação com o usuário durante o ciclo de desenvolvimento

# Modelos de Qualidade de Software:

- ISO/IEC 25010 (SQuaRE): Define características de qualidade e subcaracterísticas relacionadas ao software.
- Modelo CMMI (Capability Maturity Model Integration): Avalia a maturidade do processo de desenvolvimento de software.

# Benefícios da Busca pela Qualidade:

- Aumento da satisfação do cliente e dos usuários finais.
- Redução de custos com suporte e manutenção.
- Melhoria da produtividade da equipe de desenvolvimento.
- Maior competitividade e valorização da marca.
- Aumento da confiabilidade e reputação do produto.

# Conclusão:

- A busca pela qualidade de software é um fator essencial para o sucesso de qualquer projeto de desenvolvimento. Através de práticas adequadas de teste e garantia de qualidade, é possível identificar e corrigir problemas precocemente, proporcionando um produto final mais confiável, seguro e eficiente. Nos próximos tópicos, exploraremos os diferentes tipos de testes de software e as melhores práticas para garantir a qualidade do produto em todas as etapas do ciclo de desenvolvimento.

# Fundamentos dos Testes de Software

- O que são Testes de Software?
- Os testes de software são atividades sistemáticas realizadas durante o processo de desenvolvimento para verificar se o software atende aos requisitos e está livre de defeitos.
- Consiste em executar o software com o propósito de encontrar falhas e garantir que ele funcione conforme o esperado.

# Objetivos e Benefícios dos Testes de Software:

- Identificar defeitos e erros no software antes de sua liberação, garantindo um produto mais confiável.
- Verificar se o software está em conformidade com os requisitos e especificações definidos.
- Assegurar que o software funcione corretamente em diferentes cenários e ambientes.
- Validar a usabilidade, desempenho, segurança e outras características do software.
- Reduzir custos de correção de defeitos, uma vez que identificá-los no início do ciclo de desenvolvimento é mais econômico.
- Aumentar a satisfação do cliente e a reputação da empresa, fornecendo produtos de alta qualidade.

# Vantagens da Abordagem de Teste Contínuo:

- Teste contínuo envolve a realização de testes em todas as etapas do ciclo de desenvolvimento, desde o início do projeto até a implantação.
- Permite a detecção precoce de problemas, tornando mais fácil e barato corrigi-los.
- Facilita a integração contínua do software, garantindo que as novas funcionalidades não causem regressões em funcionalidades existentes.
- Ajuda a construir um ambiente de desenvolvimento confiável, que promove a cultura da qualidade em toda a equipe.

# Abordagens de Teste de Software:

- Teste Manual: Realizado por testadores humanos, que executam o software de forma interativa e exploratória.
- Teste Automatizado: Utiliza ferramentas de automação para executar casos de teste predefinidos, agilizando o processo e permitindo execuções repetitivas.
- Testes Funcionais: Concentram-se em verificar se as funcionalidades do software estão de acordo com os requisitos.
- Testes Não Funcionais: Avaliam aspectos como desempenho, segurança, usabilidade e compatibilidade.

# Ciclo de Vida dos Testes de Software:

- Planejamento de Testes: Definição dos objetivos, escopo, recursos e cronograma dos testes.
- Projeto de Casos de Teste: Elaboração dos casos de teste com base nos requisitos e cenários de uso.
- Execução dos Testes: Realização dos testes e registro dos resultados.
- Rastreamento e Correção de Defeitos: Identificação e registro de defeitos, acompanhamento de sua correção e verificação de resolução.
- Relatório e Encerramento: Documentação dos resultados dos testes e encerramento da atividade de teste.

# Conclusão:

- Os fundamentos dos testes de software são essenciais para garantir a qualidade e confiabilidade do produto final. Por meio de práticas adequadas de teste, é possível reduzir riscos, aumentar a eficiência do desenvolvimento e proporcionar uma experiência positiva aos usuários. Nos próximos tópicos, exploraremos os diferentes tipos de testes de software e suas aplicações para tornar o software ainda mais robusto e seguro.

# Tipos de testes

- Existem vários tipos de testes de software, cada um com um propósito específico e foco em diferentes aspectos do software. Abaixo estão alguns dos principais tipos de testes de software:

# Testes Unitários

- Objetivo: Verificar a funcionalidade correta de unidades individuais do código (por exemplo, funções, métodos ou classes).
- Escopo: Concentra-se em testar componentes isoladamente.
- Ferramentas comuns: JUnit (para Java), NUnit (para .NET), pytest (para Python).

# Testes de Integração:

- Objetivo: Verificar a integração correta entre diferentes módulos ou componentes do software.
- Escopo: Validação das interfaces e interações entre as unidades integradas.
- Pode envolver testes de integração vertical (de cima para baixo) ou horizontal (de esquerda para direita).

# Testes Funcionais:

- Objetivo: Verificar se as funcionalidades do software estão de acordo com os requisitos e especificações.
- Escopo: Testa as funções do software para garantir que produzam os resultados esperados.

# Testes Não Funcionais:

- Objetivo: Avaliar atributos não funcionais do software, como desempenho, segurança, usabilidade e escalabilidade.
- Escopo: Verificação das características do software além das funcionalidades.

# Testes de Regressão:

- Objetivo: Verificar se as alterações no código não introduziram defeitos em funcionalidades já testadas e funcionando anteriormente.
- Escopo: Execução de casos de teste que garantem que as funcionalidades existentes não foram afetadas pelas alterações.

# Testes de Aceitação do Usuário (UAT - User Acceptance Testing):

- Objetivo: Verificar se o software atende aos requisitos e expectativas dos usuários finais.
- Escopo: Realizado pelos usuários finais ou representantes dos clientes.

# Testes de Usabilidade:

- Objetivo: Avaliar a facilidade de uso e a experiência do usuário com o software.
- Escopo: Garantir que o software seja intuitivo e atenda às necessidades dos usuários.

# Testes de Desempenho:

- Objetivo: Avaliar a performance e a capacidade de resposta do software sob diferentes condições de carga.
- Escopo: Verifica se o software atende aos requisitos de desempenho especificados.

# Testes de Segurança:

- Objetivo: Identificar vulnerabilidades e garantir a segurança do software contra ameaças externas.
- Escopo: Verificação de possíveis falhas de segurança que possam comprometer a integridade e a confidencialidade dos dados.

# Obrigado