Ciclo de vida de Software

A fase de testes: A importância do V&V

O que é a fase de testes?
A fase de testes é o processo de validação. O objetivo é garantir que o software construído funcione como esperado, atendendo aos requisitos e sem falhas. A detecção de bugs e problemas é crucial antes de o software ser entregue ao usuário final.

V&V
A Verificação e Validação (V&V) são processos fundamentais no desenvolvimento de software, especialmente durante a fase de testes. A **verificação** busca garantir que o sistema foi construído corretamente de acordo com as especificações e requisitos definidos, analisando se cada etapa do desenvolvimento está em conformidade com o que foi planejado. Já a **validação** tem como objetivo assegurar que o produto final atende às necessidades reais do usuário e cumpre sua função no contexto para o qual foi projetado.

A importância de V&V está no fato de que ambos atuam como uma barreira contra falhas e defeitos que poderiam comprometer a qualidade do software. Enquanto a verificação evita erros de implementação e inconsistências técnicas, a validação garante que o software entregue tenha valor prático, resolvendo os problemas para os quais foi criado. Juntas, essas atividades reduzem riscos, aumentam a confiabilidade do sistema e evitam retrabalhos caros em fases avançadas do projeto.

Tarefas e trabalhos executados nos testes:

1. Planejamento de Testes: Mesmo que um plano inicial seja feito na fase de projeto, aqui ele é detalhado e executado. A equipe define os cenários de teste, as ferramentas a serem usadas e os critérios para a aprovação do software.
2. Execução de Testes: Os testadores executam os testes planejados, registrando os resultados e os erros encontrados em um sistema de gerenciamento de bugs.
3. Relatório de Bugs: Qualquer desvio do comportamento esperado é documentado com o máximo de detalhes possível, incluindo os passos para reproduzir o problema.
4. Regressão: Após a correção de um bug, a equipe de testes executa novamente os testes que passaram anteriormente para garantir que a correção não tenha introduzido novos erros ou afetado outras partes do sistema.

Tipos de testes executados:

- Testes de Unidade: Testam as menores partes do código (funções, métodos, classes) de forma isolada para garantir que funcionem corretamente. Geralmente, são escritos pelos próprios desenvolvedores.
- Testes de Integração: Verificam se os diferentes módulos e componentes do sistema se comunicam e funcionam juntos como esperado.
- Testes de Sistema: Avaliam o sistema completo para garantir que ele atenda a todos os requisitos funcionais e não-funcionais.
- Testes de Aceitação do Usuário (UAT): O cliente ou usuário final testa o software em um ambiente de simulação real. Se o software atender às expectativas do negócio, ele é "aceito" para a próxima fase.

Stakeholders envolvidos:

- Testadores (Analistas de QA): Profissionais especializados em encontrar bugs e garantir a qualidade do software.
- Equipe de Desenvolvimento: Responsável por corrigir os bugs encontrados.
- Analista de Sistemas/Negócios: Auxilia no entendimento do comportamento esperado do software e na definição dos cenários de teste.
- Cliente/Usuário Final: Participa ativamente dos Testes de Aceitação, validando se o produto final está de acordo com o que foi solicitado.A sinergia entre a implementação e os testes é o que garante a construção de um software robusto e de alta qualidade. Ambas as fases são essenciais para transformar um plano em um produto funcional e confiável.

Técnicas e Métodos Utilizados

- Test-Driven Development (TDD): escrever testes antes do código.
- Behavior-Driven Development (BDD): testes baseados em comportamento esperado.
- Automação de Testes: Selenium, Cypress, JUnit, pytest.
- Ferramentas de Qualidade: SonarQube, ESLint, PMD.

Saídas Esperadas

- Casos de teste documentados.
- Relatórios de execução de testes.
- Registro e status de defeitos corrigidos.
- Evidências de validação (logs, capturas, métricas).
- Software pronto para implantação ou homologação.

Ciclo de vida de Software

A fase de entrega e manutenção: do lançamento à vida útil do software

A fase de **entrega e manutenção** marca o ponto culminante do ciclo de desenvolvimento de software e o início de sua vida útil no mundo real. A entrega é o momento de colocar o software em operação, e a manutenção é o suporte contínuo necessário para garantir que ele continue funcionando bem e atendendo às necessidades dos usuários ao longo do tempo.

O que é a fase de entrega (deploy)?

A **entrega**, ou *deploy*, é o processo de transferir o software de um ambiente de desenvolvimento ou teste para um ambiente de produção, onde os usuários finais podem acessá-lo. É o momento de lançar o produto, seja para um grupo de usuários específicos ou para o público em geral. A entrega pode ser um evento único ou uma série de lançamentos contínuos, dependendo da metodologia de desenvolvimento.
Tarefas e trabalhos executados na entrega:

1. **Preparação do Ambiente de Produção:** A equipe configura os servidores, bancos de dados e redes que o software irá utilizar. O ambiente de produção deve ser seguro, escalável e otimizado para a performance do sistema.
2. **Instalação do Software:** O código-fonte, bibliotecas e todos os arquivos necessários são instalados no ambiente de produção. Isso pode ser feito manualmente ou, de forma mais comum, com o uso de ferramentas de automação.
3. **Testes Pós-Entrega:** Após o software estar em produção, a equipe realiza testes finais para garantir que tudo está funcionando corretamente. É uma verificação de sanidade para confirmar que a migração não causou nenhum problema.
4. **Treinamento e Documentação:** A equipe prepara manuais de uso e documentação técnica para os usuários e para a equipe de suporte. O treinamento para usuários e administradores é realizado para que todos saibam como operar o novo sistema.

Stakeholders envolvidos:

- **Gerente de Projeto:** Coordena o lançamento e garante que todos os planos de comunicação e treinamento estejam em andamento.

- **Equipe de Desenvolvimento:** Apoia a transição e a resolução de quaisquer problemas que surjam durante a instalação.
- **Equipe de Operações (DevOps):** Especialistas na automação e gerenciamento de ambientes de produção. Eles são cruciais para um *deploy* suave e eficiente.
- **Cliente/Usuário Final:** Começa a usar o software em seu ambiente real, fornecendo o primeiro feedback sobre a experiência.

O que é a fase de manutenção?

A **manutenção** começa no momento em que o software é entregue. É a fase mais longa do ciclo de vida, pois o software precisa ser suportado e adaptado continuamente para permanecer relevante e funcional. O trabalho de manutenção não é apenas consertar o que quebra, mas também melhorar o que já existe.
Tarefas e trabalhos executados na manutenção:

1. **Manutenção Corretiva:** A equipe corrige bugs e falhas que não foram detectados nas fases anteriores. É a solução de problemas para garantir que o software funcione como o esperado.
2. **Manutenção Adaptativa:** O software é modificado para se adaptar a um novo ambiente, como uma nova versão do sistema operacional, um novo navegador ou um novo hardware. O objetivo é garantir a compatibilidade e a funcionalidade contínua.
3. **Manutenção Evolutiva:** Novas funcionalidades e melhorias são adicionadas ao software para atender a novas necessidades do negócio ou solicitações dos usuários. Esta é a forma mais comum de manutenção e é essencial para a evolução do produto.
4. **Manutenção Preditiva:** A equipe monitora o sistema para identificar e resolver problemas potenciais antes que eles se tornem sérios. Isso inclui o monitoramento de desempenho, logs de erros e segurança.

Stakeholders envolvidos:

- **Equipe de Suporte e Atendimento ao Cliente:** Coleta feedback e relatos de bugs dos usuários e os repassa para a equipe de desenvolvimento.
- **Equipe de Desenvolvimento:** Aloca tempo para corrigir bugs, desenvolver novas funcionalidades e adaptar o software às mudanças do ambiente.
- **Gerente de Produto/Negócio:** Prioriza os novos recursos e as melhorias com base no feedback dos usuários e nas metas da empresa.
- **Usuário Final:** Contribui ativamente com feedback e solicitações que impulsionam a evolução do software.

A entrega e a manutenção são fases cruciais que asseguram o valor do software a longo prazo. Um *deploy* bem-sucedido garante que o software chegue ao usuário, e uma manutenção eficaz garante que ele continue a servir seus propósitos e a evoluir com o tempo.

Técnicas e Métodos Utilizados

Entrega

- **CI/CD (Integração Contínua e Entrega Contínua).**
- **Infraestrutura como código (IaC):** Ansible, Terraform, Docker, Kubernetes.
- **Testes em Produção:** canary releases, feature toggles.

Manutenção

- **Monitoramento e Observabilidade:** Prometheus, Grafana, ELK Stack.
- **Controle de versão e gestão de mudanças:** Git, ITIL Change Management.
- **Gestão de bugs e backlog:** Jira, Trello, Azure DevOps.
- **Ciclos de atualização contínua (DevOps/Agile).**

Importância das Fases

- **Entrega**: garante que o investimento feito se traduza em valor real para o usuário. Uma entrega mal planejada pode gerar rejeição do sistema, perda de confiança e riscos de falhas críticas.
- **Manutenção**: representa a **maior parte do custo total de um software** (chegando a 60–80% do ciclo de vida). Sem manutenção adequada, o sistema rapidamente se torna obsoleto, inseguro ou inutilizável.

Resumo:

- **Entrega (Deploy):** levar o sistema pronto ao ambiente real, garantindo migração de dados, treinamento e aceitação do cliente.
- **Manutenção:** manter o software vivo, corrigindo, adaptando e evoluindo conforme as necessidades.