Técnicas e Estratégias de Teste de Software

Uma estratégia de testes de software descreve a abordagem geral e os objetivos das atividades de teste. Ela deve contemplar os níveis ou fases de teste, os tipos de testes a serem realizados e as técnicas para sua execução. A estratégia de testes de softwares também deve descrever com clareza os critérios para a conclusão dos testes e os critérios de sucesso a serem usados.

Responsáveis: Gerente de Projeto, Engenheiros de Software e Especialistas em Testes.

Importância: A importância das estratégias é evitar tempo desperdiçado, esforço desnecessário, infiltração de erros sem serem descobertos.

Passos: Primeiramente os testes focalizam um único componente ou um pequeno número de componentes relacionados, aplicados para descobrir erros nos dados e na lógica de processamento.

Em seguida testes de alto nível são executados para descobrir erros na satisfação dos requisitos dos clientes. E por fim os erros encontrados devem ser diagnosticados e corrigidos usando o processo de depuração.

Produtos: Especificação de Teste – documenta a abordagem da equipe de software para o teste, definindo o Plano de Testes – descreve uma estratégia global e um procedimento que define passos específicos de teste e os testes que serão conduzidos.

Antes de abordamos cada estratégias de testes é importante veja o resumo que mostra:

Estratégias de Teste de Software para arquitetura de Software Convencionais

O quê	Quem	Como
Teste de Unidade – Código	Desenvolvedores	Focaliza cada componente individualmente, garantindo que ele funcione adequadamente como uma unidade. Usa técnicas de testes que exercitam caminhos específicos na estrutura de controle de um componente, para garantir completa cobertura e máxima detecção de erros. Teste Caixa-Branca.
Teste de Integração – Projeto e Arquitetura de Software	Desenvolvedores	Os componentes devem ser montados ou integrados para formar o pacote de software completo. Cuida dos tópicos associados com os problemas duais de verificação e construção de programas. As técnicas de projeto de casos de teste que enfocam as entradas e saídas são mais prevalentes durante a integração.
Validação – Sw construído x Análise de Requisitos (alto nível)	Analista de Teste	Fornecem garantia final de que o software satisfaz a todos os requisitos funcionais, comportamentais e de desempenho.
Sistema	Analista de Teste	Verifica se todos os elementos (hardware, pessoal, banco de dados) combinam adequadamente e se a função/desempenho global do sistema é alcançada.

Estratégias de Teste de Software para Arquitetura de Software Orientados a Objetos (OO)

Esta estratégia é idêntica a aplicada a arquiteturas convencionais, mas diferente na abordagem.

À medida que as classes são integradas em uma arquitetura OO, uma série de testes de regressão é feita para descobrir erros devidos a comunicação e colaboração entre classes (componentes) e efeitos colaterais causados pela adição de novas classes.

Finalmente o sistema é testado como um todo para garantir que erros nos requisitos sejam descobertos.

Teste de Unidade

O primeiro nível de teste é o Teste Unitário, que envolve assegurar que cada funcionalidade especificada no desenho do componente tenha sido implementada corretamente neste componente.

Características:

_ Testes realizados em uma unidade independente do produto;

_ Estágio mais baixo da escala de teste;

_ Aplicado nos menores componentes de código criados, visando garantir que estes atendem as especificações funcionais e de arquitetura;

_ Geralmente realizado pelo desenvolvedor que construiu a unidade;

_ Utiliza técnicas de Caixa Branca.

Ex.: funções, pedaços de códigos.

Este tipo de teste está limitado à função do componente.

Teste de Integração

Neste nível os testes não são focados em “o quê” os componentes fazem, mas se eles se comunicam conforme especificado no desenho do sistema.

Características:

_ Acontece após os Testes Unitários;

_ Realizados dentro de um ambiente controlado;

_ Geralmente realizado pelo analista de sistema para um módulo ou conjunto de programas;

_ Normalmente são seguidos dois métodos de interação:

_ Abordagem Top-Down

_ Abordagem Bottom-Up

Teste de Regressão

Tem como propósito garantir que os defeitos encontrados foram corrigidos e que as correções ou inserções de novos códigos em determinados locais do software não afetaram outras partes inalteradas do produto. Trata de re-testar o teste.

É necessário ter ferramentas para execução do teste de regreção, isto porque é inviável testar novamente todo o Software.

Reduz "efeitos colaterais". Devem ser executados toda vez que uma mudança importante é feita no Software (integração de novos componentes).

Teste de regressão deve focalizar função de módulo crítico.

Teste Fumaça

Em suma é considerado uma estratégia de integração constante, onde software é reconstruído (com adição de novos componentes) e submetido a teste todos os dias.

Este tipo de teste deve exercitar o sistema de ponta a ponta, sem precisar ser exaustivo, mas deve ser capaz de expor os problemas principais, no entanto deve ser rigoroso.

Teste de Validação

Este tipo de teste começa no fim do teste de Integração, aqui o software está completamente montado e os erros de interface já foram descobertos e corrigidos, focaliza ações visíveis ao usuário e saídas do sistema reconhecidas pelo usuário, não existindo distinção entre o software convencional e o orientado a objetos.

Validação - o software funciona de um modo que pode ser razoavelmente esperado pelo cliente. O teste de validação é feito mediante a Especificação dos Requisitos de Software – documento que descreve todos os atributos do software visíveis aos usuários, este documento contém uma seção chamada Critério de Validação, base para a abordagem do respectivo teste.

A validação do software é conseguida por intermédio de uma série de testes que demonstram conformidade com os requisitos.

Documentação

Plano de Teste – delineia as classes de teste a ser conduzidas.

Procedimento de Teste – define os casos de teste específicos.

Ambos são projetados para garantir que todos os requisitos funcionais sejam satisfeitos, bem como as características comportamentais sejam conseguidas, todos os requisitos de desempenho sejam alcançados, a documentação esteja correta, usabilidade e outros requisitos sejam satisfeitos.

Após cada caso de teste tenha sido executado, uma de duas possíveis condições se realiza:

A característica de função ou desempenho satisfaz a especificação e é aceita ou é descoberto um desvio da especificação e uma lista de deficiência é criada, estes raramente podem ser corrigidos antes da entrega programada.

É impossível para o desenvolvedor de software prever como o cliente usará realmente um programa, o que parece claro para o testador pode ser inteligível para um usuário no campo.

Quando um software é feito sob encomenda, uma série de testes de aceitação é conduzido para permitir ao cliente validar todos os requisitos, este conduzido pelo usuário final, podendo acontecer ao longo de um período de semanas ou meses, descobrindo consequentemente erros cumulativos que poderiam degradar o sistema ao longo do tempo.

Se o software é desenvolvido como um produto a ser usado por vários clientes, não é prático realizar testes formais de aceitação com cada um. Para isto utilizam-se os seguintes testes: alfa e beta.

Testes Alfa – conduzido na instalação do desenvolvedor com os usuários finais. Sendo o software usado em um ambiente natural com o desenvolvedor acompanhando os usuários e registrando erros e problemas de uso, conduzido em um ambiente controlado.

Teste Beta – conduzido nas instalações dos usuários finais, sem a presença do desenvolvedor. O cliente registra todos os problemas reais ou imaginários e os relata ao desenvolvedor responsável em fazer as devidas correções e publicar uma nova versão.

Podemos tratar aqui três dimensões:

Funcionalidade – quais as regras de negócio devem contemplar o Software.

Performance – envolve tempo mínimo de resposta das funções.

Qualidade - envolve atributos de confiança do Software.

Neste contexto aproveitaremos para falar do teste de funcionalidade.

Teste de Funcionalidade

O teste funcional tem por meta verificar se o software executa corretamente suas funções, se a implementação das regras de negócio foi apropriada e se o sistema é capaz de sustentar sua correta execução por um período contínuo de uso.

Baseado nas técnicas de caixa-preta. Analisa as saídas e resultados. Os testes funcionais são baseados nos procedimentos determinados nos casos de teste.

Tem como objetivos: assegurar a funcionalidade do sistema, incluindo entrada de dados, processamento e resposta, verificar se os requisitos dos usuários estão implementados e se atendem os usuários, verificar se o sistema funciona corretamente após um período contínuo de utilização.

Deve ser usado em qualquer sistema devendo ter suas funcionalidades testadas, pode ser usado desde a fase de especificação de requisitos até a fase de operação do sistema.

Teste de Sistema

Os testes de sistema têm foco no funcionamento do sistema como um todo, para validar a exatidão e a perfeição na execução das funções requeridas.

Deve ser executado mediante a especificação do sistema para verificar se estão sendo entregues as funcionalidades que o cliente solicitou.

Características Deste Nível:

- Acontece após todos os testes de integração,

- Realizado dentro de um ambiente controlado (servidor, equipe, máquinas clientes, Sw, browser,

- Realizado pela equipe de testes,

- Envolve testes especializados para validar todos os requisitos funcionais e não-funcionais como: Desempenho, Volume, Documentação e Robustez.

Teste de Recuperação - Teste de Contingência

A meta principal do teste de contingência é verificar se, após uma falha, a sua recuperação é adequadamente executada, garantindo a continuidade dos serviços.

Força ao sw falhar de diversas maneiras e verifica se sua recuperação é executada adequadamente.

Objetivos:

- Manter o backup dos dados;

- Armazenar o backup em local seguro;

- Documentar os procedimentos de recuperação;

- Deixar claro as responsabilidades das pessoas em caso de um desastre.

Quando Usar:

- Sempre que a continuidade dos serviços for essencial para o negócio;

- Perdas devem ser estimadas (quanto vou perder por hora?);

- A quantidade de perda potencial estipula a quantidade de esforço que irá ser empregada.

Teste de Segurança

A principal meta do teste de segurança é garantir que os dados ou funções de um sistema possam ser acessados apenas por atores autorizados a acessá-las.

Todas as formas de ataque de acesso indevido devem ser simuladas.

Verifica se todos os mecanismos de proteção estão embutidos no sistema ou protegerão de fato de acessos indevidos.

O escopo de teste de segurança vai desde verificar se um usuário só tem acesso as funcionalidades que ele realmente deve ter até testar a segurança do sw contra ataques de hackers.

Objetivos:

- Garantir que os dados do sistema estão protegidos adequadamente;

- Assegurar que todos os riscos que envolvem os acessos indevidos foram identificados;

- Analisar as ameaças e vulnerabilidades, tanto físicas como lógicas;

- Assegurar que os mecanismos de controle contra acessos indevidos foram corretamente implementados.

Quando Usar:

- Testes básicos de segurança devem ser aplicados a todos os softwares;

- Quando a informação que circula através do software for de importância fundamental para a organização.

Teste de Estresse

A principal meta do teste de estresse é entender o comportamento do sistema durante condições-limite de execução ou fora da tolerância esperada.

Tipicamente envolve a execução do sistema com baixos recursos de hardware e software, ou a concorrência por estes recursos (retorno do BD, uso da rede).

Objetivos:

- Determinar a que condições-limite de recursos o software é capaz de ser executado;

- Verificar se o sistema é capaz de garantir tempos adequados de resposta sendo executado em condições-limite;

- Verificar se há restrições quanto ao ambiente em que o software vai operar;

- Determinar que volumes de transação, normais e acima dos normais, podem ser processados num período de tempo esperado;

Quando Usar:

- Quando não se sabe quais as condições mínimas de recursos para operacionalização do sistema, sem que haja perdas significativas.

Teste de Desempenho - Teste de Perfomance

O teste de performance, ou de desempenho como também é conhecido, mede e avalia o tempo de resposta, o número de transações e outros requisitos sensíveis ao tempo de resposta do sistema.

Objetivos:

- Determinar o tempo de resposta das transações;

- Verificar se os critérios de desempenho estabelecidos estão sendo atendidos;

- Identificar pontos de gargalo no sistema;

- Verificar o nível de utilização do hardware e software.

Quando Usar:

- Quando se deseja avaliar o tempo de resposta de uma funcionalidade do sistema ou de todo o sistema;

- Devem ser realizados quando ainda há tempo hábil de serem realizadas correções.

Teste de Usabilidade

Este tipo de teste envolve mais a área de desenho, projeto interface usuário, mas que deve ser revisado na fase de teste considerando utilização de manuais, on-line help, agentes e assistentes eletrônicos, etc.

Visa verificar a facilidade que o software possui de ser claramente entendido e facilmente operado pelos usuários. O teste aqui é baseado em check-list com vários itens que o software deve atender para que ele possa ser considerado para boa utilização.

Tendo como objetivos: verificar a facilidade de operação do sistema pelo usuário, verificar a facilidade de entendimento das funções do sistema pelo usuário, através da utilização de manuais, on-line help, agentes e assistentes eletrônicos, etc.

Usado para executar diversas operações do sistema, utilizado a documentação do mesmo.

Um outro teste que não é abordado é o teste de conformidade.

Teste de Conformidade

Este teste verifica se a aplicação foi desenvolvida seguindo os padrões, procedimentos e guias da área de processos.

As metodologias são usadas para aumentar a probabilidade de sucesso do projeto, e portando devem ser testadas.

Tem como objetivos: verificar se as metodologias de desenvolvimento de sistema estão sendo seguidas, garantir as conformidades aos padrões da empresa, avaliar se a documentação do sistema é racional e está completa.

Deve ser usado quando se deseja que os padrões sejam seguidos, quando se deseja determinar o nível de seguimento dos padrões e quando se deseja identificar pontos falhos na metodologia.