DCC / ICEx / UFMG

Testes de Software

Eduardo Figueiredo

http://www.dcc.ufmg.br/~figueiredo

Testes de Software

- Teste de software buscam por erros ou anomalias em requisitos funcionais e não funcionais
- Classificação de testes pelo objetivo
  - **Teste de Validação**: mostrar que um programa faz o que é proposto a fazer
  - **Teste de Defeito**: descobrir os defeitos do programa antes do uso

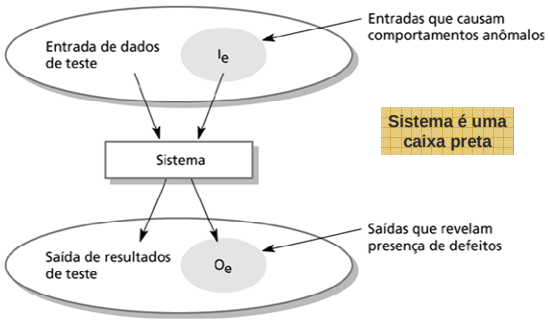
Teste de Validação

- Pretende mostrar que o software atende aos seus requisitos
  - Faz o que o cliente deseja
- Um teste bem sucedido mostra que o requisito foi implementado
- Refletem o uso esperado do software

Teste de Defeito

- Destinado a revelar defeitos no sistema
- Um teste de defeitos bem sucedido é aquele que revela defeitos no sistema
- Os casos de teste podem ser obscuros
  - Não precisam refletir exatamente como o sistema é normalmente usado

Modelo de Entrada e Saída

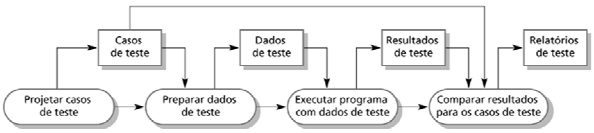


Modelo de Entrada e Saída

- Dado que:
  - O conjunto de entradas I gera um conjunto de saídas O
  - Algumas entradas erradas I_e geram saídas com defeitos O_e
- Testes de Defeito buscam encontrar as entradas em I_e que revelam saídas em O_e
- Testes de Validação envolvem entradas corretas I (não incluem entradas em I_e)

Processo de Teste

- Exemplo de processo de teste baseado em planos
  - As atividades são planejadas antes de serem executadas



Casos e Dados de Teste

- Casos de teste
  - Declarações do que será testado
  - Especificações das entradas para o teste
  - Especificações das saídas esperadas do sistema
- Dados de teste
  - Entradas criadas para o sistema
  - Eles podem ser gerados automaticamente

Resultados e Relatório

- Resultados de teste
  - Saídas que somente podem ser previstas por pessoas que conhecem o domínio de negócio do sistema
- Relatório de teste
  - Pode ser feito de forma manual, seguindo um formulário específico
  - Pode ser automatizado comparando os resultados esperados às saídas dos testes

Testes de Desenvolvimento

Teste de Desenvolvimento

- O sistema é testado durante o desenvolvimento para descobrir defeitos
  - Os próprios programadores são responsáveis pelos testes
- Classificação dos testes de desenvolvimento
  - Teste Unitário
  - Teste de Componente (integração)
  - Teste de Sistema

Teste Unitário

- Objetivo é garantir que uma unidade ou classe funciona
  - Testa unidades individuais de programa de forma independente
- Geralmente é de responsabilidade do próprio desenvolvedor da unidade
  - Os testes são derivados da experiência do desenvolvedor

Teste de Classe (OO)

- O teste completo de uma classe de objetos requer
  - Teste de todas as operações associadas com um objeto
  - Atribuir e obter valores a todos os atributos de objeto
  - Exercício do objeto em todos os estados possíveis
- A herança dificulta o teste de classe

Automatização de Testes

- Sempre que possível, os testes unitários devem ser automatizados
- Um teste automatizado têm três partes
  - Configuração: inicia o sistema com o caso de teste e dados de entrada
  - Chamada: chama o objeto a ser testado
  - Afirmação: compara o resultado da chamada ao resultado esperado

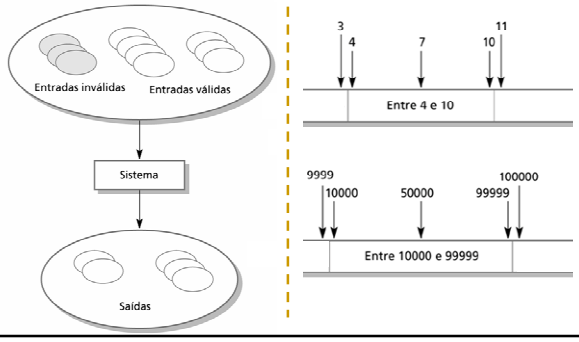
Escolha do Caso de Teste

- Teste de software é caro
  - Portanto, é importante a escolha de casos de testes efetivos
- Estratégias para escolha dos testes
  - Teste de partições
  - Teste estrutural

Teste de Partições

- Dados de entrada e resultados de saída podem ser particionados
  - O programa se comporta de maneira semelhante para cada partição
- Exemplos de partições
  - Números positivos / negativos
  - Itens de um mesmo menu
- Casos de teste devem ser escolhidos para exercitar cada partição

Exemplos de Partições



Diretrizes do Teste de Partições

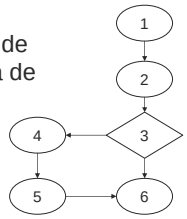
- Testar o software com sequências de tamanhos extremos
  - Sequência de comprimento zero
  - Sequência com um único valor
  - Sequência com o tamanho máximo
- Usar sequências de tamanhos diferentes em testes diferentes
- Derivar testes para o primeiro, o médio e o último elementos da sequência

Teste Estrutural

- Frequentemente chamado de teste caixa-branca
- A escolha de casos de teste ocorre de acordo com a estrutura do programa
  - O conhecimento do programa é usado para identificar casos de teste
- O objetivo é exercitar todas as declarações do programa

Teste Estrutural de Caminho

- O objetivo é assegurar que cada caminho do programa é executado pelo menos uma vez
  - Ponto de partida do teste de caminho é um fluxograma de programa



1 – 2 – 3 – 4 – 5 – 6
1 – 2 – 3 – 6

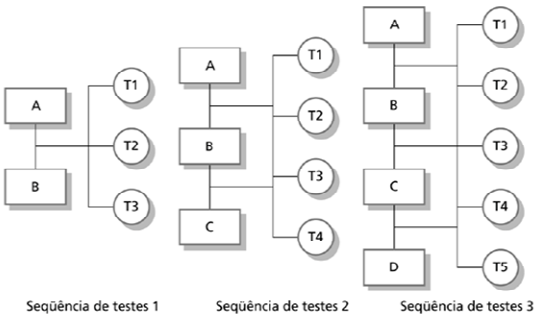
Teste de Componente

- O objetivo é garantir que dois ou mais componentes funcionam juntos
  - Testa grupos de componentes integrados para criar um sistema ou um subsistema
  - Os testes se concentram nas interfaces de comunicação entre componentes
- A geralmente responsabilidade é de uma equipe independente de teste

Tipos de Testes

- Integração *top-down*
  - Desenvolver o esqueleto do sistema e preenchê-lo com componentes
- Integração *bottom-up*
  - Integrar componentes de infra-estrutura e, depois, adicionar componentes funcionais

Integração Incremental



Teste de Sistema

- O sistema é testado como um todo
  - Testa uma release de sistema que será entregue ao cliente
- O teste de sistema verifica
  - se os componentes são compatíveis
  - se eles interagem corretamente
  - se transferem os dados certos no momento certo,etc.

Diretrizes para Teste de Sistema

- É impossível fazer testes exaustivos em um sistema
- Algumas diretrizes
  - Todas as funções acessadas por menus devem ser testadas
  - As combinações de funções dos mesmos menus devem ser testadas
  - As funções devem ser testadas com entradas corretas e incorretas

Teste de Aceitação

Teste de Aceitação

- Também chamado de teste de release
- Uma versão particular do software é testada para uso fora do ambiente de desenvolvimento
- O teste é caixa-preta
  - Nenhum conhecimento interno da estrutura do software é necessária

Diretrizes do Teste Caixa-Preta

- Escolher entradas que forcem o sistema a gerar as mensagens de erro
- Projetar entradas que causem estouro de pilha
- Forçar a geração de saídas inválidas
- Forçar resultados de cálculo a serem muito grandes ou muito pequenos

Teste de Sistema x Aceitação

- Objetivos diferentes
  - Testes de sistema focam na descoberta de defeitos
  - Testes de aceitação buscar verificar se o sistema atente aos requisitos do cliente
- No teste de aceitação, o software é testado por uma equipe separada
  - Não é pela equipe de desenvolvimento

Testes baseados em Requisitos

- Casos de uso podem ser uma base para derivar os testes de um sistema
  - Ajudam a identificar as operações a serem testadas
  - Ajudam no projeto dos casos de teste
- Os requisitos devem ser testáveis
  - As entradas e saídas podem ser identificadas no Diagrama de Sequência

Testes de Desempenho e Estresse

- Testes de desempenho
  1. Planejamento de uma série de testes
  2. A carga é constantemente aumentada
  3. Verifica o limite em que desempenho do sistema se torne inaceitável
- Testes de estresse forçam o sistema além de sua carga máxima de projeto

Teste de Usuário

- Neste tipo de teste, o usuário ou cliente é quem fornece os dados de teste
- Tipos de teste de usuário
  - Teste Alfa: usuários trabalham com a equipe de desenvolvimento
  - Teste Beta: uma versão do software é disponibilizada para os usuários finais
  - Aceitação: testa final para implantar o sistema em ambiente de produção

Bibliografia

- Ian Sommerville. **Engenharia de Software**, 9ª Edição. Pearson Education, 2011.
  - Cap. 8 Testes de Software