Luiz Fernando Geraldo dos Santos - RA: 1110481823051
Isabelly Pinheiro Serra - RA: 1110481823042

1) Descreva a técnica de caixa-preta critério particionamento em classes de equivalência para projetar casos de teste. Como as classes de equivalência são definidas (DELAMARO, cap 2, página 11,12)

Resposta:

O particionamento de equivalência se trata de um teste de caixa preta onde o domínio de entrada ´de um programa é dividido em classes de dados. As classes de equivalência podem ser definidas de acordo com as seguintes diretrizes:

1. Se a condição de entrada especifica um intervalo de valores, por exemplo, valores entre 1 e 12, então teremos uma classe válida e duas inválidas, pois os valores antes de 1 são inválidos e os valores de depois de 12 também, formando assim duas classes de valores inválidos, apenas os valores entre 1 e 12, farão parte da classe de dados válidos
2. Se uma condição de entrada exige um valor específico: uma classe de equivalência válida e duas inválidas são definidas, pois, assim como na primeira diretriz, existe apenas uma classe onde o valor é válido, e no caso, essa classe possui apenas um valor
3. Se uma condição de entrada especifica um membro de um conjunto, uma classe de equivalência válida e uma inválida são definidas. Ou seja, caso a condição de entrada seja haver u valor específico dentro de um conjunto, caso o valor esteja no conjunto, a classe de dados será válida, caso não esteja, a classe será inválida

2) Qual o objetivo do teste de caixa-preta critério análise do valor limite. Descreva algumas diretrizes
que levem à determinação dos dados de entrada (DELAMARO, cap 2, página 14)
Resposta:
Se uma condição de entrada especifica um intervalo entre dois números, por exemplo x e y, os testes devem ser feitos com o valor x, o valor y, com um valor anterior ao valor x, um valor anterior ao valor y, e um valor posterior a x e um valor posterior a y, para assim, alcançar uma maior cobertura dos casos de teste. Essa diretriz complementa o particionamento por equivalência. É importante aplicar o critério análise do valor limite também nas condições de saída.

3) Descreva o objetivo do teste funcional sistemático
. Faça algumas considerações sobre a capacidade de
encontrar defeitos quando se utiliza este critério. (DELAMARO, cap 2, página 15)

Consiste em particionar os domínios de entrada e saída, a fim de estabelecer o uso de dois casos de teste por domínio. Isso faz com que seja minimizado o problema de defeitos coincidentes mascarar falhas. Requer também avaliação do limite de cada partição e da subsequente a ele.
Desta forma fornece algumas diretrizes para a seleção dos casos de teste, aumentando a chance de revelar defeitos que são sensíveis aos dados de entrada. Além disso proporciona maior cobertura do

código está sendo testado.

A divisão em domínios reduz drasticamente os números de entradas de dados para teste, quando escolhemos as classes de equivalência podemos tirar (no mínimo) dos casos de testes de cada uma, o que já é suficiente para nos apresentar qual será o comportamento do programa quando aquele dado for introduzido. Além disso, não é necessário o conhecimento da estrutura interna do programa, apenas a sua especificação.

4) Sommerville descreve alguns dos benefícios em se ut
ilizar uma abordagem de desenvolvimento dirigido po
r testes, descreva quais são esses benefícios (Sommerville, 9ª
. ed., .cap 8, pag. 155)

Somerville descreve este metodo como:

Proporcionando um melhor entendimento do sistema, pois os testes são implementados antes mesmo do código fonte, ou seja, o desenvolvedor já sabe o que o teste espera, logo já compreende o que precisa desenvolver.

Cobertura de código, pois para cada segmento de código deve haver um teste previamente feito, com isso o desenvolvedor já possui a certeza de que todo o código fonte será executado nos testes.

Teste de regressão, com um conjunto de testes sendo desenvolvidos de forma incremental, ao mesmo momento o programa está sendo

desenvolvido. Assim sempre é possível realizar testes de regressão a fim de detectar bugs recorrentes de mudanças no código fonte.

Depuração simplificada, quando um teste falha, fica evidente a localização do problema, então o código recém escrito e verificado e modificado. Desta forma não se faz necessário o uso de ferramentas de depuração.

Documentação, pois os ´próprios testes descreve, o que o código está fazendo, portanto a leitura dos testes torna melhor o entendimento acerca do software.

Redução dos custos dos testes de regressão, fazendo com que os testes já existentes sejam executados de forma rápida e barata.

Testes Funcionais

Teste Caixa-Preta

- Detalhes da implementação não são considerados
- Seu objetivo é determinar se o programa satisfaz aos requisitos funcionais e não-funcionais que foram especificados
- Mais utilizado nos testes de sistema
- Particionamento em classes de equivalência
  - quantidade coerente de casos de teste
  - reduz a quantidade de entradas de dados a um conjunto finito e mais eficiente
- Analise do valor limite
  - Seleciona os elementos que estão mais à extremidade do caso de teste, discrepantes

Teste sistemático

- Particionamento de equivalência + Análise de valor limite
  - Obter alto indice de cobertura de implementação

Desenvolvimento dirigido por testes

- Teste e desenvolvimento caminham juntos, desenvolvimento incremental
  1. Identificar incremento de funcionalidade
  2. Escrita do teste para a funcionalidade
  3. Execução do teste novo sem a funcionalidade (teste deve falhar)
  4. Implementação da funcionalidade e execução do teste (teste deve passar)
  5. Segue para a próxima funcionalidade