Teste de Software

Depuração de Software (*Debugging*)

PUC Minas

Instituto de Ciências Exatas e Informática

Prof. Lesandro Ponciano

Departamento de Engenharia de Software e Sistemas de Informação (DES)

Objetivos da Aula

- Contextualizar a atividade de depuração de software em Teste de Software
- Apresentar dificuldades associadas à atividade de depuração
- Discutir as principais características das estratégias de depuração de software
  - Depuração por Força Bruta
  - Depuração por Indução
  - Depuração por Dedução
  - Depuração por *Backtracking*
  - Depuração por Teste

Depuração

- A função de um caso de teste é revelar defeitos em um software
- Quando um caso de teste cumpre sua função, entra em cena a atividade depuração
- Depuração é uma atividade que consiste de duas fases
  1) Determinar a natureza e o local do defeito
  2) Corrigir o defeito

Depurar é Desagradável

- O ego entra na frente, depurar é mostrar que o programador fez algo incorreto
- É uma atividade desgastante, requer muito esforço mental e geralmente é feita sobre pressão de tempo
- É fácil se perder, o defeito pode estar em qualquer parte do código
- É uma atividade solitária, há pouca informação de auxílio sobre como fazer a depuração com exito

Estratégias de Depuração

1) Depuração por Força Bruta
2) Depuração por Indução
3) Depuração por Dedução
4) Depuração por *Backtracking*
5) Depuração por Teste

Depuração por Força Bruta

- Colocar escritas em arquivo ou na tela
  - Requer mudar o código para inserir as escritas
  - Só funciona para programas bem pequenos

- Monitorar os valores das variáveis com ferramentas automáticas (*break points*)
  - Complexo acompanhar a dinâmica
  - Gera uma quantidade de dados muito grande
  - Tentativa e erro

Depuração por Indução

- Parte de uma situação particular para uma situação geral
- Parte-se dos sintomas de uma falha e vai ampliando para outras situações por meio de hipóteses
  - Ex.: Não funciona para 0 → não funciona para inteiro → não funciona para valor fracionário → não funciona para qualquer número
- Questiona-se *o que*, *onde* e *quando*, enquanto realiza a análise

Depuração por Dedução

- Parte-se do caso geral para o caso específico
  - Define-se uma teoria geral e vai eliminando, refinando até se chegar ao caso específico

- Passos comuns
  1) Enumere todas as possíveis causas ou hipóteses
  2) Colete dados para eliminar possíveis causas ou hipóteses
  3) Refina as hipóteses que permanecerem
  4) Verifique as hipóteses restantes

Depuração por *Backtracking*

- Parte de uma resposta incorreta e, de trás para frente, tenta identificar a causa da falha a partir do fluxo do programa
- Trata-se de uma "execução reversa" do programa
- Pode ser efetivo para programas pequenos, mas não em programas grandes

Depuração por Teste

- Casos de teste para *teste*
  - Visa expor um defeito que ainda não se manifestou como uma falha, ou que não foi detectado previamente
  - Foca na cobertura do programa

- Casos de teste para *depuração*
  - Visa prover informação sobre um defeito suspeito
  - Foca em um caso particular e visa cobrir apenas esse caso

Correção e Teste

- Quando o defeito é encontrado:
  1) o código deve ser corrigido
  2) deve-se criar um caso de teste que pode revelar tal defeito se ele voltar a surgir no futuro
  3) deve-se executar os testes de regressão para garantir que não foram inseridos novos defeitos
  4) conforme o processo, deve-se documentar e reportar o ocorrido

Referências

- Myers, Glenford J. et al (2004) "The Art of Software Testing." 2ed. New York, NY, USA: John Wiley & Sons. . (Capítulo 7)