Engenharia de Software II

Aula 11

http://www.ic.uff.br/~bianca/engsoft2/

Ementa

- Processos de desenvolvimento de software
- **Estratégias e técnicas de teste de software** (Caps. 13 e 14 do Pressman)
  - **Depuração** (seção 13.7)
- Métricas para software
- Gestão de projetos de software: conceitos, métricas, estimativas, cronogramação, gestão de risco, gestão de qualidade e gestão de modificações
- Reengenharia e engenharia reversa

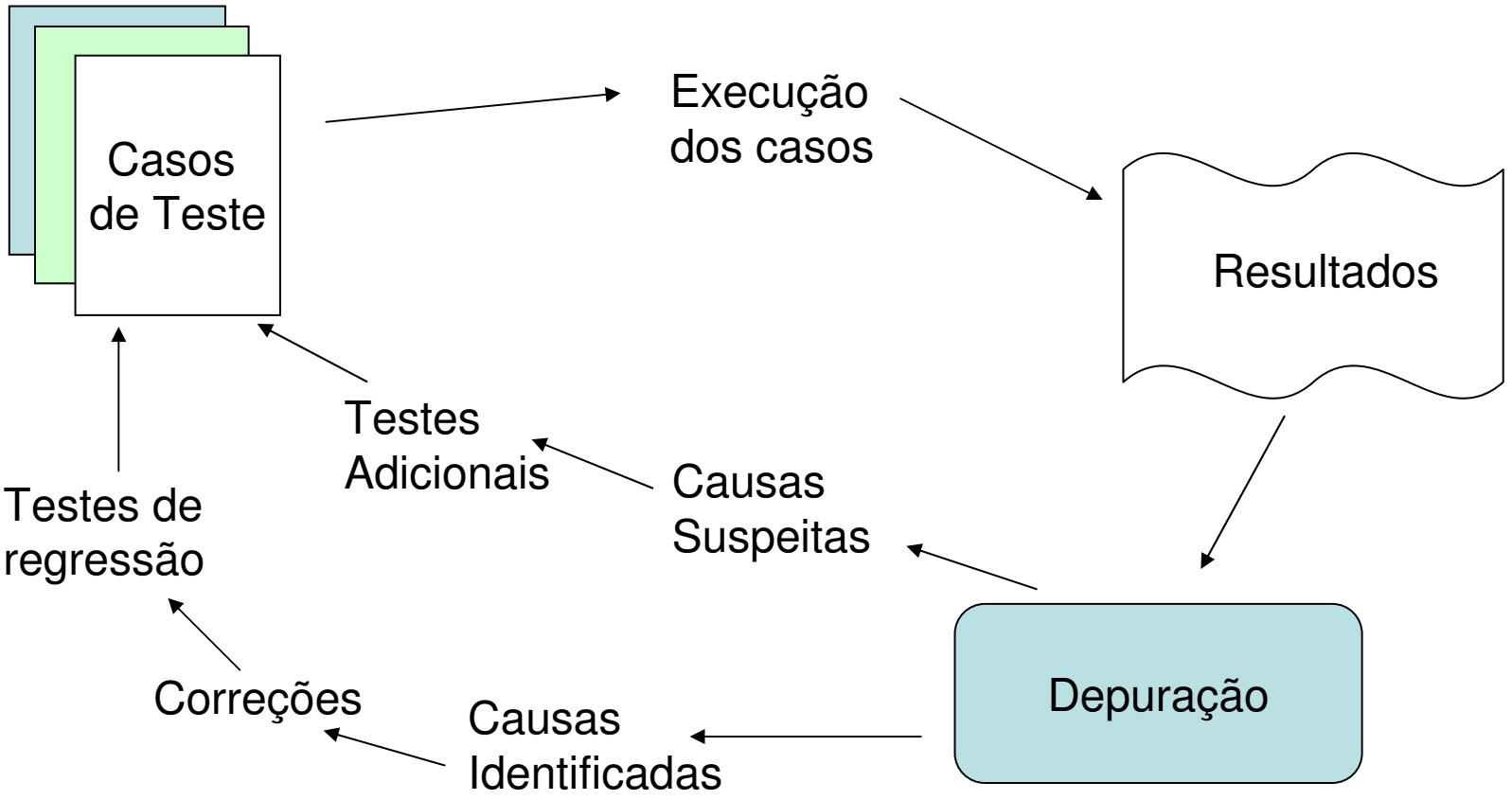
Depuração ("*debugging*")

- Ocorre como conseqüência do teste bem-sucedido.
  - Quando um caso de teste descobre um erro, a depuração é feita para se descobrir a causa do erro e corrigí-lo.
- Ao contrário do teste, é uma arte.
  - A manifestação externa do erro e a causa interna do erro podem não ter uma relação óbvia.
  - É necessário formular e testar hipóteses.

Exemplo Simples de Teste de Hipótese

- Problema: um abajur da casa não funciona.
- Hipóteses:
  1. A casa está sem luz.
     - Para testar essa hipótese, ligo outros aparelhos conectados a eletricidade.
  2. A tomada não funciona.
     - Para testar essa hipótese, mudo o abajur de tomada.
  3. A lâmpada não funciona.
     - Para testar essa hipótese, troco a lâmpada do abajur.
  4. Se todas as hipóteses foram eliminadas, concluo que o problema é com o abajur.

O Processo de Depuração



Por que a depuração é difícil?

- O sintoma e a causa podem estar espacialmente separados.
- O sintoma pode desaparecer temporariamente quando outro erro é corrigido.
- O sintoma pode, na verdade, ser causado por condições externas ao programa.
  - Por exemplo, imprecisões de arredondamento.
- Pode ser difícil reproduzir exatamente as condições de entrada que causam o erro.
- O sintoma pode ser intermitente.
  - Por exemplo, se ele depende de condições de hardware ou de outro programa.

Considerações Psicológicas

- Grandes variações na habilidade de depuração têm sido relatadas sobre programadores com a mesma educação e experiência.
- A depuração é uma das partes mais frustrantes da programação.
  - A elevada ansiedade e a má vontade em aceitar a possibilidade de erros aumentam a dificuldade da tarefa.
- Apesar de ser difícil aprender a depurar, seguir um processo de depuração e utilizar ferramentas ajuda muito.

TRAFFIC: um processo de depuração de sete passos

1. **T**rack: arquive o sintoma num banco de dados.
2. **R**eproduce: reproduza o sintoma.
3. **A**utomate: automatize e simplifique o caso de teste.
4. **F**ind: encontre possíveis causas para o sintoma.
5. **F**ocus: focalize nas causas mais prováveis.
6. **I**solate: isole a conexão de causa para sintoma.
7. **C**orrect: corrija o erro.

1) Arquive o sintoma

- Crie um relatório de erro.
  - Deve relatar como o erro pode ser reproduzido.
- Uma ferramenta de relatório de erros deve ser utilizada.
  - Exemplos:
    - BugZilla
    - IssueTracker
    - SourceForge
    - TRAC

2) Reproduza o sintoma

- Não é tão fácil quanto parece.
- Em alguns casos é difícil controlar a entrada porque ela pode depender de:
  - Usuários
  - Comunicações com outros computadores
  - Tempo
  - Números aleatórios
  - Descritores de arquivos existentes nos sistemas
  - Escalonamento de processos
- Ferramentas para facilitar a reprodução de sintomas:
  - Winrunner, Android, revirt, checkpoint.org

3) Simplifique o caso de teste

- Encontre a entrada mais simples tal que o erro ainda aconteça.
- Encontre pequenas mudanças na entrada para as quais o erro deixe de acontecer.

4) Encontre possíveis causas

- Volte atrás no programa.
  - Exemplo: Se `print(x)` mostra um valor incorreto, encontre de onde vem o valor de `x`.
    - Provavelmente haverão várias fontes, que servirão como várias hipóteses a ser testadas.
- Usar ferramentas ("*debuggers*") ajuda muito:
  - É possível observar o estado das variáveis e executar o programa passo a passo para encontrar possíveis causas.
- Se depois de um tempo limite uma causa possível não é encontrada, deve-se pedir ajuda de outro desenvolvedor.

5) Focalize nas causas mais prováveis

- Ache a hipótese mais provável para a causa do erro.
  - Intuição e entendimento do programa ajudam.
- O próximo passo é validar ou rejeitar essa hipótese.

6) Isole a conexão causa-sintoma.

- Reconstrua como o programa muda de estado da possível causa para o sintoma observado.
- Caso essa não seja a causa, escolha outra hipótese e repita o processo.

7) Corrija o erro

- Corrija o erro e teste o programa novamente.
- Deve-se ter cuidado para não introduzir outros erros.
- Deve-se também pensar no que poderia ter sido feito para que o erro não ocorresse.
  - O processo de desenvolvimento também deve ser corrigido.