PUCRS
Pontifícia Universidade Católica do Rio Grande do Sul

ESCOLA POLITÉCNICA

Reutilização
Conceitos

46504-04 - Construção de Software

Prof. Msc. Eduardo Arruda
eduardo.arruda@pucrs.br

Material original elaborado pelo Prof. Dr. Marcelo Yamaguti

Referências

- Estude para aprofundamento no conteúdo:
  - IEEE. **Guide to the Software Engineering Body of Knowledge**. SWEBOK. Version 3. IEEE Computer Society. 2014. – Chapter 3.
  - SOMMERVILLE, I. **Engenharia de software**. 9ª ed. São Paulo: Pearson Brasil, 2011. – Capítulo 16.
  - PFLEEGER, S. L. **Engenharia de Software**: teoria e prática. 2ª ed. São Paulo: Prentice Hall, 2004. – Capítulo 12.
  - PRESSMAN, Roger S. **Software Engineering**: A Practitioner's Approach. McGraw-Hill, 2000.
  - WERNER,C; BRAGA,R. **Desenvolvimento Baseado em Componentes**. Minicurso desenvolvido no XIV Simpósio Brasileiro de Engenharia de Software, 4 a 6 de outubro de 2000, João Pessoa – PB.

Tecnologias de reúso - introdução

- Engenharia de Software: busca pela melhor produtividade e qualidade em software.
- Produtividade pode ser aumentada pela reutilização de software.
- Encontrar elementos que possam ser reusados a partir da decomposição de sistemas: especificações, subsistemas, módulos, subprogramas, sub-rotinas, classes, etc.

Implementação

- Reúso:
  - Pode ocorrer em vários níveis:
    - reúso de especificação.
    - reúso de código (bibliotecas, API – *Application Programming Interface*).
    - reúso de componentes e *frameworks* (conjunto de objetos).
    - reúso de sistemas (COTS – *Commercial-Off-The-Shelf*, software de prateleira).
  - Pode ter dois focos:
    - Implementar **com** reúso: visa a reutilização de software existente.
    - Implementar **para** reúso: visa a criação de software que possa ser reutilizado.

Reúso de software

- Reúso de experiência:
  - **Padrões** - soluções que podem ser aproveitadas para situações atuais.
- Reúso de projeto:
  - ***Frameworks*** – conjunto de classes concretas e abstratas são reutilizadas para aplicações de domínios distintos (exemplo: MVC)
  - **Linhas de produtos** – conjunto de aplicações com arquitetura comum e componentes compartilhados.
- Reúso de código:
  - **Bibliotecas** (APIs) específicas de linguagens (exemplos: C++, Java)
  - **Componentes**

Reúso em OO

- Destaque do reúso de Projeto e Experiência
  - soluções são propostas num nível de abstração mais elevado.
  - apoio a etapas mais iniciais do desenvolvimento de um software.
  - melhoria no ganho com o reúso.

Benefícios do reúso (Sommerville)

- Maior confiabilidade:
  - Componentes reutilizados têm maior confiabilidade do que componentes novos.
- Redução dos riscos de processo
  - Componentes reutilizáveis facilitam a estimativa de tempo de um projeto.
- Uso efetivo de especialistas
  - Especialistas podem desenvolver componentes reutilizáveis pra vários projetos, em vez de realizar o mesmo trabalho em projetos diferentes.
- Conformidade com padrões
  - Padrões de desenvolvimento (por exemplo: padrões de interface com o usuário) podem ser implementados como um conjunto de componentes-padrão.
- Desenvolvimento acelerado
  - O uso de componentes acelera a produção pois diminui o tempo de desenvolvimento e validação.

Reúso de código

- Problemas do reúso restrito apenas a código:
  - reúso de código muito dependente da linguagem de programação.
  - reúso somente de código tende a deixar as etapas de anteriores (análise e projeto) e posteriores (testes) do ciclo de vida desprovidas de apoio ao reúso.
  - reúso somente de código pode comprometer ganhos de produtividade e qualidade.

Resistência ao reúso

- Poucas empresas e organizações têm uma ideia clara da forma como seria seu plano de reúso de software.
- Embora haja ferramentas ou componentes que fornecem apoio ao reúso de software (repositórios de componentes para reúso), muitos de seus desenvolvedores não utilizam estes recursos.
- Existe, de maneira geral, pouco treinamento para ajudar engenheiros e gerentes de software para entendimento e aplicação da abordagem de reúso.
- Muitas empresas continuam a utilizar metodologias de desenvolvimento de software que não privilegiam reutilização.

Limitações

- Limitações com o reúso:
  - **Aumento nos custos de manutenção**: caso os códigos-fonte dos componentes não estejam disponíveis a adaptação dos mesmos pode tornar-se onerosa a cada mudança do sistema.
  - **Falta de ferramentas de apoio**: há poucas ferramentas CASE com suporte a reúso e pode ser inviável integrá-las a bibliotecas de componentes.
  - **Síndrome do "não-foi-inventado-aqui"**: muitos desenvolvedores preferem desenvolver seu próprios componentes do que reutilizar componentes prontos (falta de confiança).
  - **Manutenção de biblioteca de componentes**: garantir que os componentes sejam reutilizados é um desafio, pois técnicas de classificação, catalogação e recuperação de componentes ainda são imaturas.

Dúvidas?