Projeto de uma Arquitetura de Software

O que é um projeto arquitetural?

- Decisões estratégicas que terão conseqüências profundas
- As decisões são tomadas num alto nível

Exemplos de decisões

- Modularização do projeto em subsistemas
- Escolha de uma estrutura de comunicação e controle entre os subsistemas
  - Quem chama quem?
- Definição das interfaces entre subsistemas
- Escolha de uma estratégia de persistência
- Escolha do paradigma de DBMS a usar
- Determinação de oportunidades para o reuso de software
- Atendimento a requisitos especiais de desempenho
- Atendimento a outros requisitos (custo, mobilidade, uso de padrões, etc.)
- Exposição das interfaces para facilitar a futura integração de aplicações (Enterprise Application Integration - EAI)
- etc.
- Observe que sempre deve-se ter um olho na satisfação dos requisitos levantados
- Veremos vários aspectos do projeto arquitetural
  - A discussão é direcionada mais a sistemas de informação

Técnicas Fundamentais para Desenvolver Arquiteturas de Software

Abstração

- Característica essencial de um módulo que o

diferencia de qualquer outro módulo e provê assim uma delimitação conceitual clara do módulo
- "Criamos abstrações"

Encapsulação

- Agrupamento dos elementos de uma abstração
  - Incluindo elementos estruturais e comportamentais
- Encapsulação provê barreiras claras entre abstrações

Ocultação de Informação

- Ocultar detalhes de implementação dos clientes de uma abstração
- Detalhes que não são necessários para o uso de uma abstração devem ser escondidos
- Quem oculta informação?
  - Módulos oculta informação
  - Classes ocultam informação
  - Métodos ocultam informação
- Encapsulação usada para prover ocultação de informação

Modularização de Sistemas

- Decomposiçao de um sistema em subsistemas
- Há vários motivos que fazem com que seja interessante modularizar um sistema em subsistemas durante o projeto arquitetural
  - Modularizar ajuda a lidar com a complexidade de sistemas (divida para conquistar), facilitando o design, o entendimento, os testes, ... através de encapsulamento e abstração
  - Modularizar ajuda a manter a coesão de cada subsistema
  - Modularizar ajuda a diminuir o acoplamento geral do sistema (há comunicação entre alguns subsistemas e usando algumas interfaces bem definidas na "borda" dos subsistemas)
  - Modularizar permite escolher como desenvolver cada subsistema: se cada subsistema será

desenvolvido internamente, contratado para desenvolvimento externo ou comprado
    - Modularizar facilita a divisão de trabalho, permitindo desenvolver em paralelo, uma equipe por subsistema
    - Modularizar permite reutilizar subsistemas em várias aplicações
    - Modularizar permite que várias aplicações compartilhem um mesmo subsistema (imagine um BD, por exemplo)
    - Modularizar permite construir sistemas distribuídos em que subsistemas podem estar espalhados fisicamente

Separação de Interesses

- Responsabilidades diferentes e não relacionadas devem ser separadas num sistema
- Exemplo:
    - Tratar com o usuário é fundamentalmente diferente de tratar com o sistema operacional

Acoplamento e Coesão

- Acoplamento: pouca "ligação" intra-módulo
    - "Ligação" é o que um módulo conhece sobre o outro
    - É o que deve ser alterado num módulo se outro módulo mudar
- Coesão: grau de conectividade entre funções e elementos
    - Coisas que "cheiram igual" devem estar juntas
    - Tem vários tipos de coesão que estudaremos depois

Separação de Interface e Implementação

- Separa "o que" deve ser feito do "como é feito"
- Discutiremos isso adiante ao abordar Interfaces e Polimorfismo
- Diminui acoplamento, permite trocar implementações,

etc.

Ponto Único de Referência

- Itens de um sistema são declarados e definidos num único ponto
  - Evita inconsistência

Divisão e Conquista

- Divide problemas grandes em problemas pequenos
- A divisão é essencial para lidar com a complexidade

programa