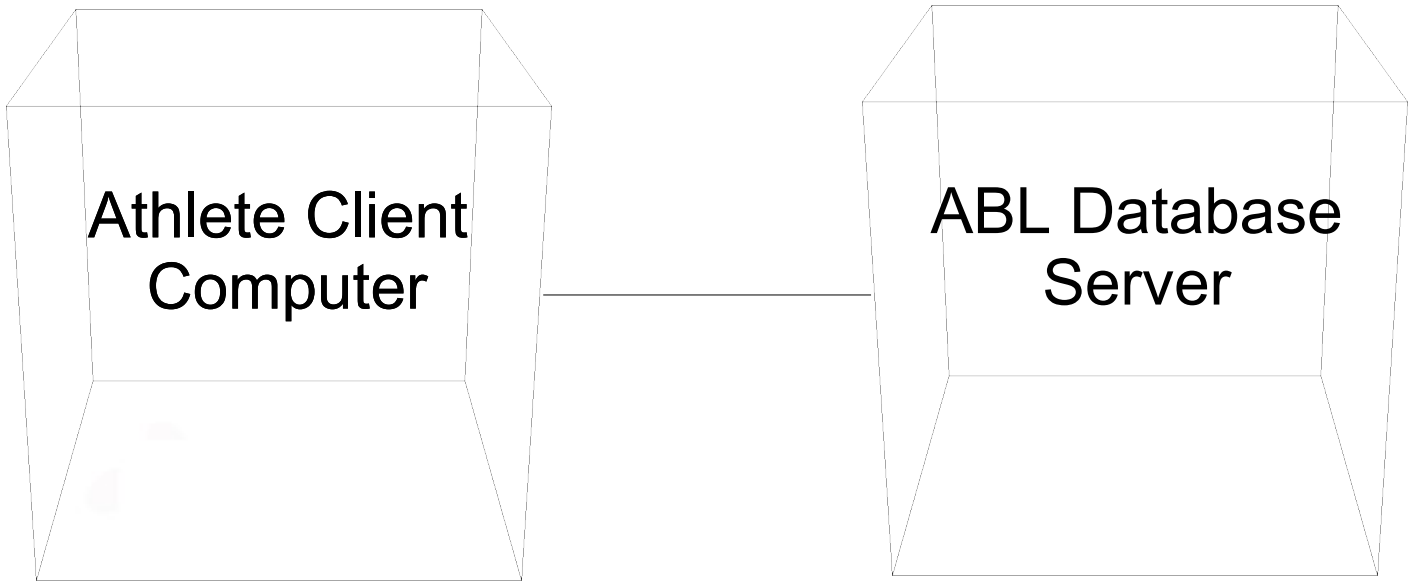
Lição 4

Engenharia de Projetos – Parte 3

Projeto de Implementação

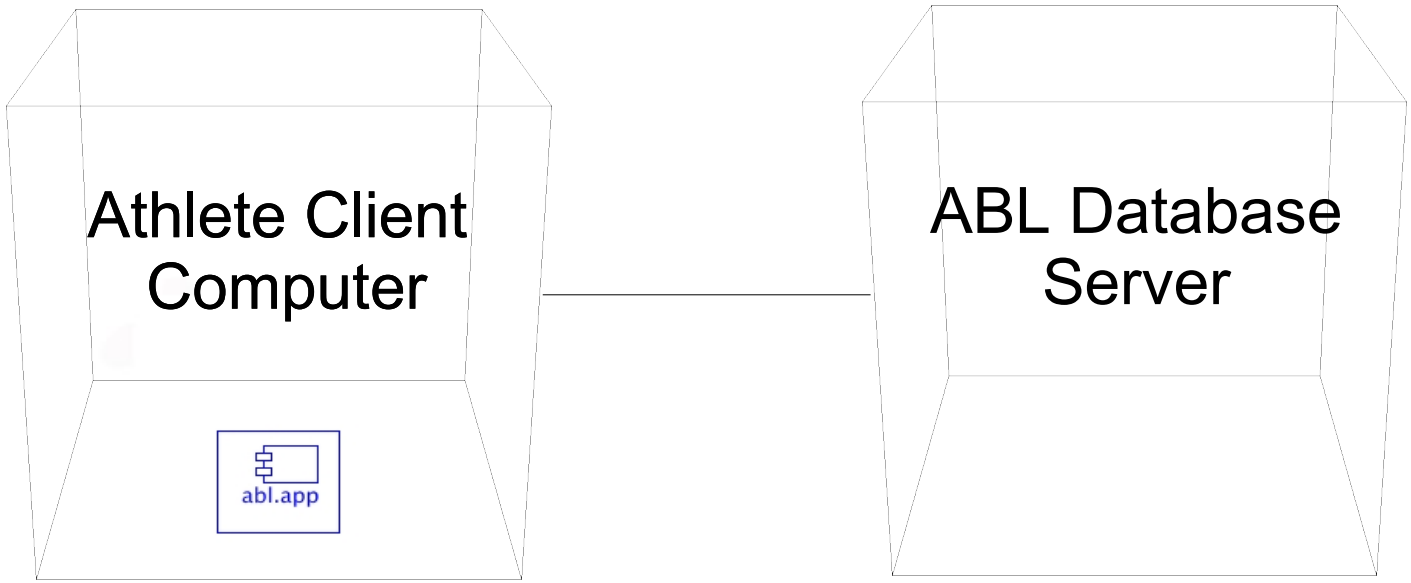
- Mostra a arquitetura física de hardware e software
- Mostra a distribuição dos componentes do software pelos componentes de hardware

Diagrama de Implementação



Desenvolvendo o Diagrama de Implementação

- Distribuir os componentes do software pelos *nodes* do computador



Razões para o modelo de Validação

- A validação é necessária para ver se o modelo:
  - Cumpre os requisitos do sistema
  - é consistente com o guia de projeto
  - serve como uma boa base para a implementação

Validações

- Componente de Software
- Classe
- Operação
- Atributo

Métricas de Projeto

- Medidas e métricas são necessárias para garantir a qualidade do design

- A classe é a unidade fundamental do design. A avaliação deve ser baseada em:
  - Classe
  - Hierarquia de Classes
  - Colaboração de Classes

Métricas CK

- Foram propostas por Chidamber e Kemerer

- Consistem de seis classes baseadas em métricas
  - Métodos por classe
  - Profundidade da árvore de herança
  - Número de classes filhas
  - Acoplamento entre objetos das classes
  - Respostas para uma classe
  - Falta de coesão nos métodos

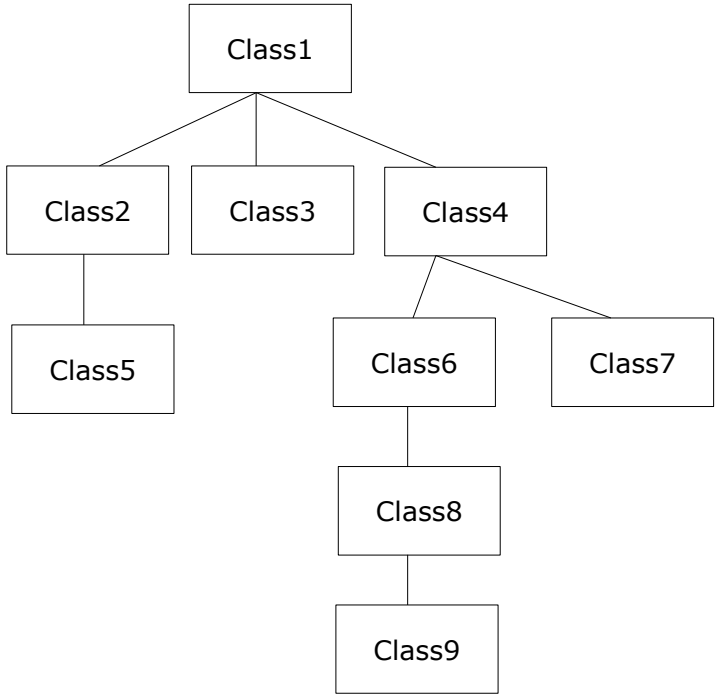
Métodos por Classe (WMC)

- Esta é calculada como sendo a soma da complexidade de todos os métodos de uma classe. Assuma que existam *n* métodos definidos numa classe. Calcula-se a a complexidade de cada método, e depois soma-se cada resultado. Existem muitas métricas que podem ser utilizadas, mas a mais comum é a métrica ciclomática

- O número de métodos e sua complexidade indicam:
  - O tamanho do esforço que será utilizado na implementação e teste da classe. Quanto maior for o número de métodos, mais complexa será a árvore de herança
  - Se o número de métodos crescer acompanhando a classe, esta estará mais propensa a se tornar mais complicada

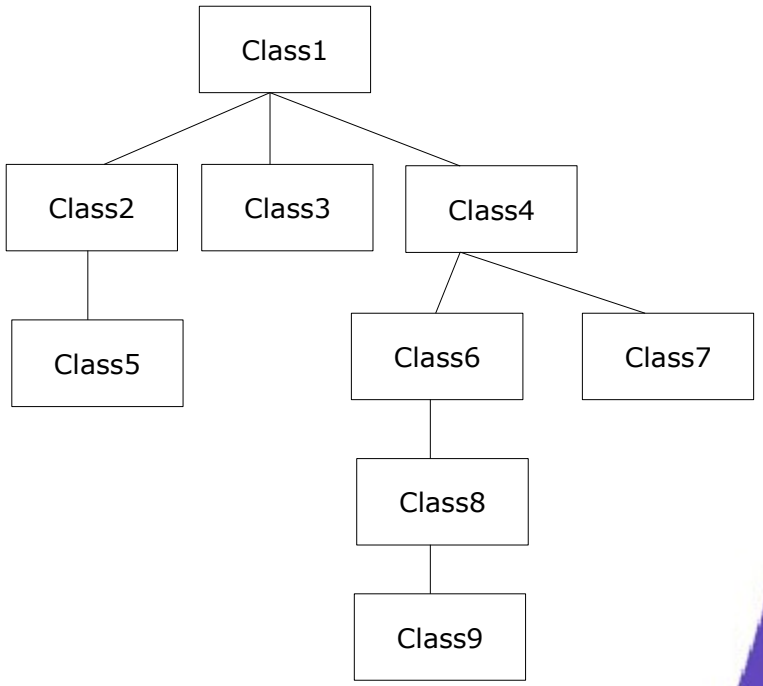
Profundidade na árvore de Heranças (DIT)

- È o número máximo desde a classe base ate a classe mais baixa
- Com o crescimento do DIT, é compreensível que o número de métodos nas classes mais baixas será maior
- No diagrama, o número de DIT é 5



Número de classes filhas (NOC)

- Número de classes filhas imediatas
- Com o aumento do número de filhas, o reuso aumenta
- Garante que uma classe base não sera diluída por uma filha inapropriada
- Na ilustração, o NOC da Class4 é 2



Acoplamento entre os objetos (CBO)

- Número de colaborações que uma classe realiza com outros objetos
- Com o crescimento deste número, o fator reusabilidade diminui
- CBO deve ser mantido no mínimo possível

Respostas para uma classe (RFC)

- Número de métodos executados em resposta a uma mensagem dada ao objeto de uma classe

- Com o crescimento deste número, o esforço requerido para testar também aumentará pois este aumenta os testes de sequência

Falta de Coesão nos Métodos (LCOM)

- Número de métodos que acessam um atributo numa classe
- Se a LCOM for alta, os métodos deverão ser acoplados juntos através de um atributo
- LCOM deverá ser mantido em seu minimo possível

Sumário

- Engenharia de Projeto
- Dicas de Qualidade para o Projeto
- Conceitos de Projeto
- Modelo de Projeto
- Arquitetura de Software
- Diagrama de Pacote
- Desenvolvendo a Arquitetura do Software
- Validação da Arquitetura
- Padrões de Projeto
- Elementos de Documentação nos Padrões de Projeto
- Amostras de Padrões de Projeto
- Conceito de Projeto de Dados

Sumário

- Padrão de Persistência do Java Database Connectivity (JDBC)
- Desenvolvendo o modelo de Projeto dos Dados
- Projeto de Relatório
- Projeto de Formulários
- Projeto de Tela e Diálogos
- Princípios básicos no Projeto de componentes
- Diagrama de Componentes
- Desenvolvendo os componentes do Software
- Diagrama de Implementação
- Desenvolvendo o modelo de Implementação
- Modelo de Projeto para a Validação
- Métricas de Projeto

Parceiros

- **Os seguintes parceiros tornaram JEDI™ possível em Língua Portuguesa:**