

DCC / ICEx / UFMG

Projeto Orientado a Objetos

Eduardo Figueiredo
<http://www.dcc.ufmg.br/~figueiredo>

Pensar Orientado a Objetos

- Onde quer que você olhe no mundo real, você vê objetos
 - Pessoas, animais, plantas, carros, etc.
- Humanos pensam em termos de objetos
 - Orientação a objetos é alto nível
 - Ou seja, mais próximo dos humanos que dos computadores

Características de Objetos

- Classificação
 - Animados: possuem vida, se movem...
 - Inanimados: não possuem vida
- Objetos possuem **atributos**
 - Tamanho, forma, cor, peso, etc.
- Objetos exibem **comportamentos**
 - Uma bola rola, um avião voa
 - Uma pessoa anda, fala, pensa, etc.

Classe de Objetos

- Objeto é uma entidade que possui um estado e operações definidas sobre este estado
- Classe é um “esqueleto” para criação (instanciação) de objetos
 - Como a planta baixa é um “esqueleto” para criação de casas

Definições

- Objeto
 - Entidade que descreve uma realidade
- Classe
 - Abstração que define o esqueleto dos objetos
- Instância
 - Objeto criado a partir de uma classe

Comunicação entre Objetos

- A comunicação pode ocorrer de várias formas
 - Envio de mensagens (exemplo, pode ser implementadas por arquivos XML)
 - Invocação de métodos remotos (RMI)
 - Chamada de métodos locais
- Forma mais comum é a chamada de métodos locais
 - Comunicação síncrona

Projeto Orientado a Objetos

- Maneira natural de visualizar o software
 - Documentação de alto nível
 - Comunicação entre membros da equipe
- Modela o software semelhante ao mundo real - usando objetos
- **Objetos** são modelados em termos de seus **atributos** e comportamento (**métodos**)

Dos Requisitos ao Projeto

Desenvolvimento OO

- Análise orientada a objetos
 - Cria um modelo de objetos para o domínio da aplicação (domínio do problema)
- Projeto orientado a objetos
 - Cria um modelo de objetos para implementar requisitos (domínio da solução)
- Programação orientada a objetos
 - Implementa o projeto orientado a objetos usando uma linguagem de programação

Desenvolvimento OO

- A transição entre estágios deve ser contínua e com notações compatíveis
 - Da análise para o projeto
 - Do projeto para a programação

Vantagens de OO

- Facilidade de entendimento
 - Mapeamento de entidades do mundo real para objetos de sistema
- Facilidade de manutenção
 - Mais fácil de alterar pois os objetos são independentes
- Facilidade de reuso
 - Objetos são potencialmente componentes reusáveis

Atividades de Projetar OO

1. Definir o contexto do sistema
2. Projetar a arquitetura
3. Identificar os objetos principais
4. Desenvolver os modelos de projeto
5. Especificar interfaces entre objetos

[Paralelo e Iterativo]

- As atividades não necessariamente são sequenciais
- Geralmente é feito de forma iterativa
 - Define-se parte do contexto do sistema
 - Projeta-se parte da arquitetura
 - Identifica-se alguns objetos
 - Modela-se estes objetos
 - Define-se suas interfaces



[Definir o contexto do sistema]

- Objetivo: compreensão do software que está sendo desenvolvido e de seu ambiente externo
- Técnicas adotadas
 - Diagramas de Casos de Uso
 - Descrição dos Cenários, etc.
- Ao definir o contexto, pode-se identificar alguns objetos do domínio

[Projetar Arquitetura]

- Primeiro passo do projeto de sistema
- O projeto arquitetural envolve
 - Identificação dos componentes principais do sistema (sub-sistemas)
 - Definição das interfaces de comunicação entre os componentes
- Regra geral: modelar de 5 a 9 subsistemas

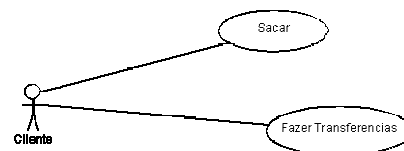
[Identificar os objetos principais]

- Identificação de objeto é um processo iterativo
 - É improvável que você faça certo na primeira vez
- Na verdade, identifica-se as classes de objetos
- Não há fórmula mágica para a identificação de objetos

[Uma Abordagem para Identificação]

- Análise gramatical baseada em
 - Descrição em linguagem natural do sistema
 - Descrição dos cenários de uso
- Como proceder
 - Substantivos são objetos ou atributos
 - Verbos são métodos
 - Refinar e definir novos objetos usando o conhecimento do domínio do sistema

[Diagrama de Casos de Uso]



Exemplo de Cenário

- Nome do Cenário: Sacar
- Ator: Cliente
- Pré-condição: Conta e senha validadas
- Fluxo normal
 1. Entrar com valor do saque
 2. Confirmar dados e operação
 3. Debitar valor da conta do cliente
- Fluxos alternativo: Saldo insuficiente
 - 3.1 Apresentar aviso ao cliente
- Pós-condição: Valor sacado é debitado do saldo do cliente

Exemplo de Cenário

- Nome do Cenário: Sacar
- Ator: Cliente
- Pré-condição: Conta e senha validadas
- Fluxo normal
 1. Entrar com valor do saque
 2. Confirmar dados e operação
 3. Debitar valor da **conta do cliente**
- Fluxos alternativo: Saldo insuficiente
 - 3.1 Apresentar aviso ao cliente
- Pós-condição: Valor sacado é debitado do saldo do cliente

Potenciais
objetos do
sistema

Exemplo de Cenário

- Nome do Cenário: Sacar
- Ator: Cliente
- Pré-condição: Conta e senha validadas
- Fluxo normal
 1. Entrar com **valor** do saque
 2. Confirmar **dados** e operação
 3. Debitar **valor** da conta do cliente
- Fluxos alternativo: Saldo insuficiente
 - 3.1 Apresentar aviso ao cliente
- Pós-condição: Valor sacado é debitado do saldo do cliente

Potenciais
atributos dos
objetos

Exemplo de Cenário

- Nome do Cenário: Sacar
- Ator: Cliente
- Pré-condição: Conta e senha validadas
- Fluxo normal
 1. **Entrar com valor** do saque
 2. **Confirmar dados** e operação
 3. **Debitar valor** da conta do cliente
- Fluxos alternativo: Saldo insuficiente
 - 3.1 Apresentar aviso ao cliente
- Pós-condição: Valor sacado é debitado do saldo do cliente

Potenciais
métodos dos
objetos

Modelos de projeto

- Fazem a ligação entre requisitos (problema) e implementação (solução)
- Mostram os objetos ou as classes de objetos e os relacionamentos entre essas entidades
- Devem incluir detalhes suficientes para facilitar a programação

Várias visões

- Para evitar modelos complexos, eles são quebrados em diversas visões
 - Modelos estáticos descrevem a estrutura estática das classes
 - Modelos dinâmicos descrevem as interações dinâmicas entre os objetos
- O modelo estático mais utilizado é o **Diagrama de Classes**

[Especificar Interfaces entre Objetos]

- Especificação de interfaces permite que objetos e componentes sejam projetados em paralelo
- Objetos podem ter várias interfaces
 - Cada interface é um ponto de vista dos métodos fornecidos
 - O Diagramas de Classes da UML pode ser usado para especificação de interfaces (semelhante a classes)

[Bibliografia]

- Ian Sommerville. **Engenharia de Software**, 9a. Edição. 2011.
 - Cap. 7: Seção 7.1