

Projeto de Software Orientado a Objetos

Prof. Fernando Castor Filho
Prof. Márcio Lopes Cornélio

1

Desenvolvimento Orientado a Objetos

- Análise, projeto e programação orientados a objetos estão relacionados, mas são distintos.
- **Análise** => Modelo de domínio
- **Projeto** => Modelo para implementar os requisitos
- **Programação** => materialização do projeto em uma linguagem que o computador entende

2

Desenvolvimento Orientado a Objetos

- Análise, projeto e programação orientados a objetos estão relacionados, mas são distintos.
- Análise => Modelo de domínio
- **Projeto** => Modelo para implementar os requisitos

Programação => materialização do projeto

O projeto normalmente está dividido em duas etapas:

- Projeto arquitetural (visto nas últimas aulas)
- **Projeto detalhado** (assunto desta aula)

3

O que é Projeto Orientado a Objetos?

Conjunto de atividades e artefatos que visa fornecer uma **solução** para o **problema** proposto pelos requisitos do sistema

- Envolve a modelagem do sistema
 - Ênfase em **resolução de problemas**
 - Pensar antes para evitar dificuldades durante a implementação
 - Ignorando "detalhes"
 - De acordo com **princípios** bem estabelecidos

4

Características do Projeto Orientado a Objetos

- Objetos são abstrações do mundo real ou entidades de sistema e gerenciam a si próprios
- Os objetos são independentes e englobam estados e informações de representação
- A funcionalidade do sistema é expressa em termos de serviços de objetos
- Áreas de **dados compartilhados** são eliminadas e os objetos se comunicam por meio da passagem de mensagens
- Objetos podem ser distribuídos e executados seqüencialmente ou em paralelo

5

Projetar é Difícil!

- **Projetar envolve criatividade**
- **Exige a avaliação de diferentes opções**
- **Requer conhecimento sobre o contexto da aplicação**
- **Em suma, para cada sistema, um projeto diferente pode ser necessário**
- **Há soluções e princípios que são recorrentes, porém**
 - Quase universalmente aceitos ou
 - Adequados para certas situações
- O foco dessa aula está nestes **princípios** e **soluções** gerais

6

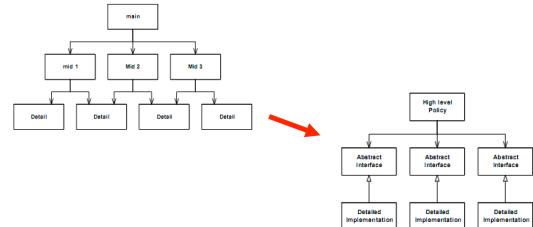
Princípios do Projeto Orientado a Objetos

- **Coesão** alta e **Acoplamento** baixo
 - Classes devem ter um conjunto pequeno e bem-definido de responsabilidades
 - Além disso, devem depender umas das outras o mínimo possível
 - **Implicam em facilidade de manutenção e compreensão**

7

Princípios do Projeto Orientado a Objetos

- Desenvolva para uma **interface** e não para uma **implementação**



8

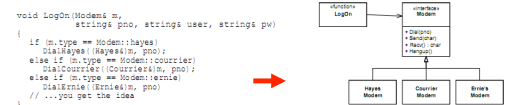
Princípios do Projeto Orientado a Objetos

- Prefira **composição** de objetos a **herança**
 - Reuso **caixa branca** vs. reuso **caixa preta**
 - **Herança** quebra o encapsulamento
 - Maior flexibilidade
 - Estrutura do sistema pode mudar em tempo de execução
 - Maior coesão
 - Classes mais focadas em um único objetivo
 - Composição implica em **mais classes**

9

Princípios do Projeto Orientado a Objetos

- Um módulo deve ser **aberto para extensão** e **fechado para modificação**

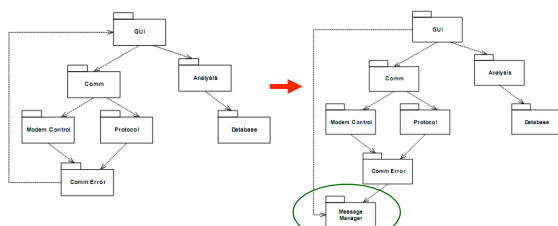


- Subclasses devem ser capazes de **substituir** suas superclasses
 - Elipses vs. Círculos
 - Listas ligadas vs. Pilhas e Filas

10

Princípios do Projeto Orientado a Objetos

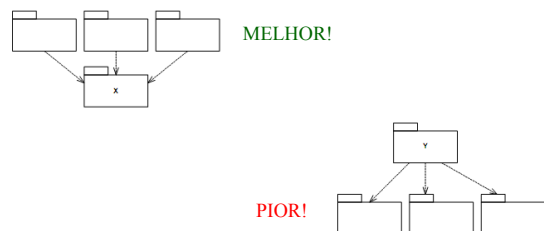
- Dependências entre pacotes não devem formar **ciclos**



11

Princípios do Projeto Orientado a Objetos

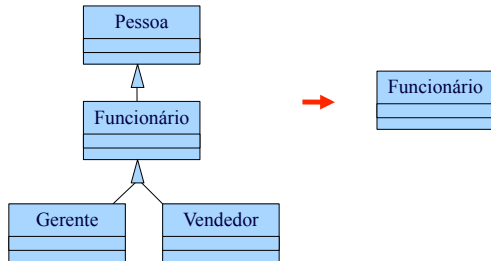
- Dependência na direção da **estabilidade**



12

Princípios do Projeto Orientado a Objetos

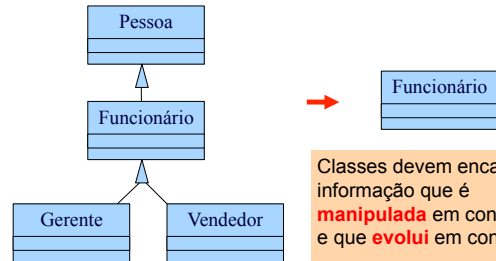
- Crie classes **apenas quando necessário**



13

Princípios do Projeto Orientado a Objetos

- Crie classes **apenas quando necessário**



Classes devem encapsular informação que é **manipulada** em conjunto e que **evolui** em conjunto

14

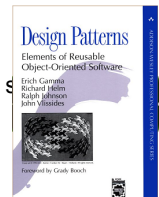
Padrões de Projeto

- Escritores de livros, histórias em quadrinhos e roteiros raramente inventam novas histórias
- Idéias frequentemente são reusadas
 - "Herói Trágico" => Hamlet, Macbeth
 - "Anti-Herói" => Aquiles, Sawyer, Lobo
- Projetistas também reutilizam soluções, de preferência as boas
 - Experiência é o que torna uma pessoa um *expert*
- Problemas são tratados de modo a evitar a reinvenção de soluções

15

O que é um Padrão de Projeto?

- Abstração de uma solução de projeto recorrente
- Envolve dependências, estruturas, interações e convenções relativos a classes e objetos
- Documentam **experiência** no de sistemas de
- Tornam projetos OO mais flexíveis, elegantes e reusáveis



16

Definição de Alexander

"... describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice"

Alexander, Christopher, Sara Ishikawa and Murray Silverstein
A Pattern Language: Towns, Buildings, Construction.
 Oxford University Press, USA, 1977.

17

Existem Muitos Tipos de Padrão!

- Padrões de gerenciamento
 - Ex. Métodos ágeis como SCRUM e XP
- Padrões organizacionais
- Padrões de análise
- Padrões arquiteturais => **Estilos Arquiteturais**
- Padrões de implementação => **Idioms**

18

Propriedades dos Padrões de Projeto

- **Padrões...**
 - fornecem um vocabulário comum
 - fornecem uma abreviação para comunicar princípios complexos
 - auxiliam na documentação do projeto do sistema
 - capturam partes essenciais de um projeto de maneira compacta
 - mostram mais de uma solução
- **Padrões não...**
 - fornecem uma solução exata
 - resolvem todos os problemas de projeto
 - aplicam-se apenas ao projeto orientado a objetos
 - Em computação, surgiram nesse contexto, porém

19

Elementos de um Padrão de Projeto

1. Nome do padrão

- Apelido usado para descrever uma solução de projeto
- Facilita discussões de projeto
- Um padrão pode ter vários nomes distintos

20

Elementos de um Padrão de Projeto

2. Problema

- Descreve quando o padrão pode ser aplicado
- Pode incluir condições sobre a aplicabilidade do padrão (contexto)
- Sintomas de um projeto inflexível ou inadequado

21

Elementos de um Padrão de Projeto

3. Solução

- Descreve elementos do projeto
- Inclui relacionamentos, responsabilidades e colaborações
- Não descreve projetos concretos ou implementações
- Funciona como um **modelo**
 - Um **estilo arquitetural** nada mais é que um padrão aplicado no nível da arquitetura

22

Elementos de um Padrão de Projeto

4. Consequências

- Resultados e compromissos
- Normalmente classificadas em **vantagens e desvantagens**
- Fundamentais para se avaliar a aplicabilidade do padrão em determinado contexto
- Em geral, padrões de projeto implicam em maior **flexibilidade**
 - É necessário avaliar se essa flexibilidade é necessária

23

Elementos de um Padrão de Projeto

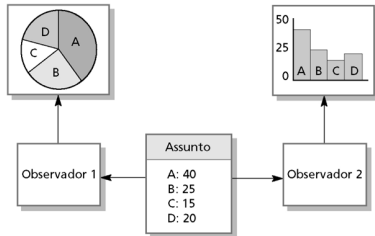
5. Usos conhecidos

- Padrões são soluções já **testadas em vários contextos**, potencialmente por várias pessoas
- Benefícios e desvantagens bem conhecidos!
- Regra dos três
 - Uma ocorrência é um evento isolado
 - Duas podem significar uma coincidência
 - Três fazem um padrão

24

Múltiplas Formas de Exibição

Figura 18.2
Vários displays.



25

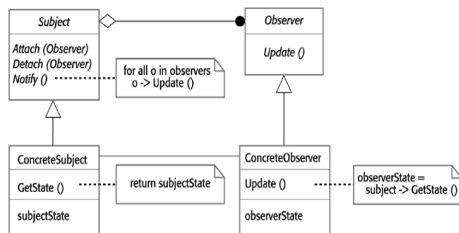
O Padrão Observer

- **Nome:** *Observer*.
- **Descrição do problema:** É necessário representar um mesmo conjunto de dados de diversas maneiras, de modo que mudanças nesses dados se reflitam em todos os modos de exibição.
- **Descrição da solução:** Mecanismo para que o *display* seja notificado de modificações no estado sem que, para isso, o estado precise conhecê-lo.
- **Consequências:**
 - Estado e *display* desacoplados
 - É fácil atualizar tanto um *display* quanto vários *displays*
 - Invocações implícitas podem resultar em erros de programação difíceis de rastrear
 - Exige código extra

26

O Padrão Observer

Figura 18.3
Padrão Observer.



27

Usos Conhecidos

- AWT do Java Development Kit
- Diversas partes da implementação da plataforma Eclipse
- Um grande número de arcabouços modernos para a construção de interfaces com o usuário

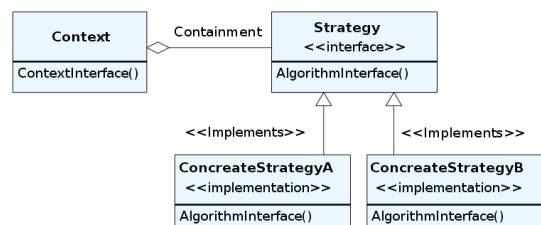
28

O Padrão Strategy

- **Nome:** *Strategy*
- **Descrição do problema:** Dependendo do contexto, é necessário usar algoritmos diferentes para resolver um problema, embora seja desejável não ter que modificar esse contexto por causa do algoritmo empregado.
- **Descrição da solução:** Definir uma interface para todos os algoritmos e fazer com que o contexto conheça apenas essa interface, não suas implementações.
- **Consequências:**
 - Separa contexto e cliente das implementações do algoritmo
 - Contexto não está ciente da estratégia usada; o cliente configura o contexto
 - *Strategies* podem ser substituídas em tempo de execução
 - Normalmente mais efetivo se usado junto com uma *Factory*

29

Padrão Strategy



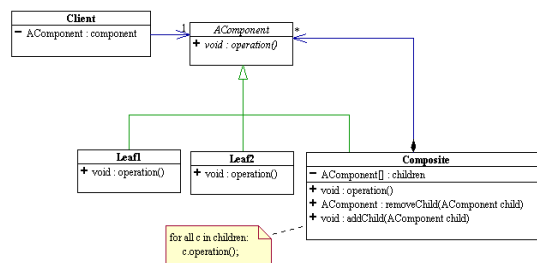
30

O Padrão Composite

- **Nome:** *Composite*
- **Descrição do problema:** Dados que o programa precisa manipular podem ser tanto elementos atômicos quanto grupos de elementos (podendo conter, inclusive, outros grupos). O programa deveria tratar esses itens de maneira uniforme
- **Descrição da solução:** Definir uma interface compartilhada tanto por itens atômicos quanto por grupos de elementos e fazer com que o programa lide com ela, sem saber se o objeto subjacente é atômico ou um grupo
- **Consequências:**
 - Trata todos os componentes do mesmo jeito, independentemente da complexidade
 - Novos tipos de componentes podem ser incluídos com facilidade
 - Pode exigir um número grande de objetos

31

O Padrão Composite



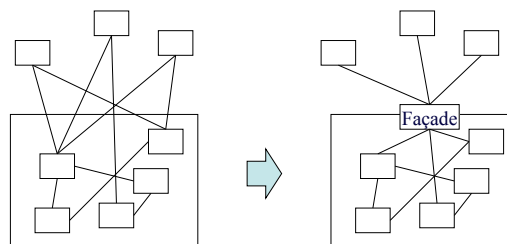
2

O Padrão *Iterator*

- **Nome:** *Iterator*
- **Descrição do problema:** É necessário separar as operações que serão realizadas sobre uma ou mais estruturas de dados da maneira como essas estruturas de dados serão percorridas
- **Descrição da solução:** Definir uma interface padrão para percorrer estruturas de dados em geral e fazer com que seus usuários vejam apenas a interface e não a maneira como a estrutura de dados é percorrida
- **Consequências:**
 - Independência entre a estrutura de dados e a maneira de percorrê-la
 - Múltiplos itadores => múltiplas maneiras de percorrer uma estrutura de dados
 - Comunicação extra entre o iterador e a estrutura
 - Modificações à estrutura podem criar inconsistências

33

O Padrão *Façade*



3

Mais sobre Padrões de Projeto

Dêem uma olhada em:

- O texto sobre padrões está legal e tem *links* para as descrições de todos os padrões do livro (com código!)

26