PUC Minas

Resenha

Projeto de Software

Gustavo Delfino Guimarães

1489062@sga.pucminas.br

Matrícula: 836079

Belo Horizonte, MG – Fevereiro de 2025

Engenharia de Software Moderna

Capítulo 6: Padrões de Projeto

O Capítulo 6 do livro "Engenharia de Software Moderna" trata dos Padrões de Projeto, soluções reutilizáveis para problemas frequentemente encontrados no desenvolvimento de software, que tem o propósito de construir sistemas que sejam mais flexíveis, coesos e de fácil manutenção.

Introdução aos padrões de projeto

Baseados no trabalho do arquiteto Christopher Alexander, os padrões de projeto foram desenvolvidos e adaptados para o desenvolvimento de software por Erich Gamma, Richard Helm, Ralph Johnson e John Vlissides, conhecidos como a Gang of Four, que os descrevem como descrições de classes e objetos que realizam um trabalho em conjunto para resolver um problema de projeto de sistema em um determinado contexto. Além de prover soluções, eles também estabelecem um vocabulário comum entre os desenvolvedores, resultando em uma comunicação mais efetiva e na documentação.

Os padrões de projetos são divididos em 3 categorias:

- **Criacionais:** Soluções para criação de objetos.
- **Estruturais:** Soluções flexíveis para composição de classes e objetos.
- **Comportamentais:** Soluções flexíveis para interação e divisão de responsabilidades entre classes e objetos.

Os padrões de projetos **Criacionais** abordados no capítulo são:

- **Fábrica**: encapsula a criação de objetos, permitindo a troca fácil de instâncias.

- **Singleton**: assegura que uma classe tenha somente uma instância e fornece um ponto de acesso global a ela.

Os padrões de projetos **Estruturais** abordados no capítulo são:

- **Proxy**: fornece um substituto ou representante para outro objeto; esse padrão controla o acesso ao objeto original.

- **Adaptador**: permite que interfaces incompatíveis trabalhem juntas. Este padrão traduz a interface de uma classe para uma interface esperada pelos clientes.

- **Fachada**: provê uma interface única (simplificada) para um conjunto de interfaces em um subsistema.

- **Decorador**: anexa responsabilidades adicionais a um objeto, dinamicamente, de modo a permitir a extensão flexível de funcionalidades.

Os padrões de projetos **Comportamentais** abordados no capítulo são:

- **Strategy**: define uma família de algoritmos, encapsula cada um e torna-os intercambiáveis.

- **Observador**: define uma dependência um-para-muitos entre objetos, assim, quando um objeto muda de estado, todos os seus dependentes são notificados.

- **Template Method:** Define o esqueleto de um algoritmo em uma operação, delegando a implementação de alguns passos para subclasses. Visitor: Permite adicionar novas operações em uma estrutura de objetos sem alterar as classes sobre as quais opera.

Apesar de os padrões de projeto apresentarem soluções comprovadas, o capítulo alerta em relação ao uso excessivo ou incorreto desses padrões, fenômeno popularizado como "paternite". É crucial avaliar cuidadosamente a necessidade de um padrão específico, garantindo, assim, que eles verdadeiramente façam sentido no projeto em questão.

Em suma, o capítulo 6 traz uma visão geral sobre padrões de projeto, ressaltando a relevância e aplicação prática deles em desenvolvimento de software moderno.

O capítulo 6 trouxe a tona uma aula específica de Programação Modular onde foi totalmente voltada para padrões de projeto e sua importância mostrando como a PUC e o curso de engenharia de software trata de um tema às vezes subvalorizado mas muito importante para um bom desenvolvimento de software.