josif 2025
17ª Jornada Científica e Tecnológica
14º Simpósio de Pós-Graduação
do IFSULDEMINAS
ISSN: 2319-0124
REALIZAÇÃO
INSTITUTO FEDERAL
Sul de Minas Gerais
APOIO
CNPq
FAPEMIG

PADRÕES DE PROJETO COMPORTAMENTAIS: comunicação e responsabilidade entre objetos

Miguel C. PEREIRA[1]; Paulo C. dos SANTOS[2];

RESUMO

A engenharia de software orientada a objetos enfrenta o desafio de gerenciar a complexidade das interações entre componentes. Este artigo, uma Revisão Bibliográfica, analisa os principais padrões de projeto comportamentais e seu papel na gestão da comunicação e distribuição de responsabilidades entre objetos. A análise foca em padrões como *Strategy, Observer* e *Mediator*, com base em publicações técnicas brasileiras recentes e na literatura canônica. A pesquisa revela que esses padrões promovem o baixo acoplamento por meio de soluções distintas: o *Strategy* encapsula algoritmos para permitir sua troca dinâmica, o *Observer* estabelece um mecanismo de notificação um-para-muitos, e o *Mediator* centraliza a comunicação complexa. Conclui-se que a seleção de um padrão comportamental é uma decisão de *design* crucial que pondera os benefícios de flexibilidade contra a complexidade estrutural introduzida.

Palavras-chave: Padrões de Projeto; Baixo Acoplamento; Arquitetura de *Software*; *Strategy*; *Mediator*; *Observer*.

1. INTRODUÇÃO

A complexidade dos sistemas de *software* modernos exige arquiteturas que sejam flexíveis e de fácil manutenção. Conforme os sistemas crescem, as dependências diretas entre objetos podem criar um código de alto acoplamento, tornando-o rígido e difícil de evoluir (BENTO, 2020). A questão central desta pesquisa é: como os padrões de projeto comportamentais gerenciam a comunicação e a distribuição de responsabilidades para construir sistemas mais resilientes? O objetivo deste estudo é analisar as soluções propostas pelos padrões *Strategy*, *Observer* e *Mediator*. A relevância do tema reside na necessidade de desenvolver software robusto. Os padrões comportamentais, em particular, tratam da forma como as classes e objetos interagem e distribuem suas tarefas (GAMMA et al., 1995), sendo fundamentais para organizar a colaboração entre eles e aumentar a flexibilidade do sistema.

2. FUNDAMENTAÇÃO TEÓRICA

Os padrões de projeto comportamentais concentram-se nos algoritmos e na atribuição de responsabilidades entre objetos, definindo seus padrões de comunicação (SOUZA, 2023). Eles aumentam a flexibilidade na forma como os objetos colaboram para realizar tarefas, frequentemente

[1]Discente, IFSULDEMINAS – *Campus* Muzambinho. E-mail: miguel.pereira@alunos.ifsuldeminas.edu.br.
[2]Orientador, IFSULDEMINAS – *Campus* Muzambinho. E-mail: paulo.santos@muz.ifsuldeminas.edu.br.

externalizando o controle sobre um comportamento ou uma comunicação. A seguir, analisamos três padrões amplamente aplicados.

2.1 *Strategy*: Alternância Algorítmica com Flexibilidade

Como proposto por Gamma et al. (1995), o padrão *Strategy* organiza algoritmos em famílias intercambiáveis, abordagem que se mostra eficaz em cenários onde há múltiplas variações de um mesmo comportamento. Por exemplo, regras de desconto em sistemas de venda podem ser implementadas como estratégias distintas, desacoplando o objeto cliente das implementações específicas (CIPRIANI, 2023). No entanto, seu uso exige cautela: a proliferação de classes pode dificultar a leitura e manutenção do código, sendo indicado quando a variação algorítmica justifica essa complexidade adicional.

2.2 *Observer*: Notificações Reativas com Baixo Acoplamento

O padrão *Observer* estabelece uma relação de dependência "um-para-muitos", na qual alterações de estado em um objeto (*Subject*) notificam automaticamente os objetos observadores (*Observers*). Um exemplo cotidiano é o sistema de notificação de aplicativos de mensagens. Embora promova desacoplamento, esse padrão exige atenção com fluxos de atualização encadeados e gerenciamento de memória — especialmente para evitar vazamentos causados por *Observers* não removidos corretamente (BAJAJ, 2023).

2.3 *Mediator*: Centralização de Comunicação Complexa

O padrão *Mediator* propõe uma abstração que centraliza a comunicação entre objetos interdependentes, chamados *Colleagues*, reduzindo o número de conexões diretas entre eles (GAMMA et al., 1995). Essa abordagem simplifica o modelo de interação, mas concentra a lógica em um único ponto, que pode se tornar um componente excessivamente carregado e difícil de manter. Sua aplicação é recomendada em sistemas com comunicações cruzadas frequentes, como interfaces gráficas ou módulos de workflow.

3. MATERIAL E MÉTODOS

Esta pesquisa caracteriza-se como uma revisão bibliográfica. A busca pelas fontes foi realizada em bases de dados como *Google Scholar* e *SciELO*, além de repositórios institucionais de universidades brasileiras. Foram utilizados os seguintes descritores e operadores booleanos: ("padrões de projeto comportamentais" *OR* "*behavioral design patterns*") *AND* (*Strategy OR Observer OR Mediator*). Os critérios de inclusão foram: artigos, dissertações e publicações técnicas, com foco em publicações brasileiras e fontes canônicas sobre o tema. Foram excluídos posts de blog sem autoria clara e materiais de *marketing*. A busca inicial retornou

16 trabalhos. Após a análise de títulos e resumos, 10 foram selecionados para leitura, resultando em um corpus final de 5 fontes que atendiam plenamente aos critérios e foram utilizadas para a elaboração deste artigo.

4. RESULTADOS E DISCUSSÃO

A análise da literatura confirma que os padrões comportamentais são soluções consolidadas para os desafios de interação em sistemas orientados a objetos. A discussão revela abordagens distintas: o *Strategy* oferece flexibilidade dentro de um objeto, tornando seu comportamento interno intercambiável, enquanto *Observer* e *Mediator* gerenciam a comunicação entre objetos.

Padrão	Problema Central	Solução Arquitetural	Trade-off Principal
Strategy	Variação de comportamento interno	Encapsulamento de algoritmos em classes intercambiáveis	Aumento do número de classes e indireção
Observer	Notificação reativa para múltiplos objetos	Dependência "um-para-muitos" com notificação automática	Fluxos de atualização difíceis de rastrear
Mediator	Comunicação desordenada entre múltiplos objetos	Centralização das interações em um objeto mediador	Complexidade concentrada em um único ponto

5. CONCLUSÃO

Nossa análise evidenciou que os padrões comportamentais *Strategy*, *Observer* e *Mediator* são ferramentas fundamentais para o desenvolvimento de sistemas modulares, manuteníveis e desacoplados. Cada padrão apresenta forças e limitações, e sua escolha deve ser motivada por necessidades específicas de comunicação e controle entre objetos.

Ao responder à questão proposta na introdução, verificamos que esses padrões de projeto oferecem caminhos distintos para melhorar a organização estrutural do código. Concluímos que sua

adoção deve ser guiada por diagnósticos claros de acoplamento e complexidade, e preferencialmente aplicada via refatoração incremental, evitando sobrecarga arquitetural prematura.

Como perspectiva para trabalhos futuros, recomenda-se investigar como esses padrões comportamentais se comportam em arquiteturas distribuídas baseadas em microsserviços, especialmente quanto aos impactos de latência e sincronização na comunicação entre módulos autônomos.

REFERÊNCIAS

BENTO, Gabriel do J. T. R. **Análise da Qualidade de Código e Refatoração do Jogo Bicho UFC Rampage**. 2020. 58 f. Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) , 2020. Disponível em: <https://repositorio.ufc.br/bitstream/riufc/55596/1/2020_tcc_gjtrbento.pdf.> Acesso em: 16 jul. 2025

BAJAJ, Manik. Criticism and caveats of Observer Pattern. **Cloudaffle**, 2023. Disponível em: <https://cloudaffle.com/series/behavioral-design-patterns/observer-pattern-criticism/.> Acesso em: 16 jul. 2025

GAMMA, Erich et al. **Design Patterns***:* Elements of Reusable Object-Oriented Software. Reading: Addison-Wesley, 1995.

CIPRIANI, Felipe. Padrão de Projeto Strategy: O que é, quando devo usar, como aplicar e exemplos. ***Tech Writers Softplan***, 2021. Disponível em: <https://www.softplan.com.br/tech-writers/descomplicando-o-strategy/.> Acesso em: 16 jul. 2025

SOUZA, Marcus Vinicius Alves da Silva. **Adoção de Padrões de Projeto de Software no Desenvolvimento de Aplicações Corporativas**. 2023. 45 f. Trabalho de Conclusão de Curso (Graduação em Análise e Desenvolvimento de Sistemas) . Disponível em: <https://repositorio.pgsscogna.com.br/bitstream/123456789/65222/1/Marcus_Vinicius_Alves_da_Silva_Souza.pdf.> Acesso em: 16 jul. 2025