UNIVERSIDADE FEDERAL DO MARANHÃO

Engenharia da Computação

João Batista Santos Silva Neto

Uma prática de ensino para Padrões de Projeto

São Luís - MA

2024

João Batista Santos Silva Neto

Uma prática de ensino para Padrões de Projeto

Trabalho presentado como requisito parcial para obtenção da nota de Trabalho de Conclusão de Curso II, ao Programa de Graduação em Engenharia da Computação, da Universidade Federal do Maranhão.

Engenharia da Computação

Universidade Federal do Maranhão

Orientador: Prof. Dr. Davi Viana dos Santos

São Luís - MA

2024

Resumo

Este trabalho apresenta a aplicação e avaliação de uma prática para o ensino de Padrões de Projeto, utilizando o modelo Four-Component Instructional Design (4C/ID). A pesquisa busca validar a eficácia de uma abordagem prática no aprendizado desses padrões, essenciais para a criação de software reutilizável e de fácil manutenção. O estudo envolveu a implementação de atividades incrementais em uma disciplina de Engenharia de Software, combinando teoria e prática para facilitar a assimilação dos conceitos. Além dos padrões de projeto, a metodologia explorou os princípios SOLID, com foco nos princípios de Responsabilidade Única, Aberto-Fechado e Inversão de Dependência. A avaliação foi realizada por meio da análise do desempenho e percepção dos alunos, destacando os impactos positivos da abordagem prática no aprendizado. Os resultados indicam que a prática estruturada contribui significativamente para a compreensão e aplicação dos padrões de projeto, sugerindo melhorias para futuras implementações da metodologia.

Palavras-chave: Padrões de Projeto, Ensino, Programação orientada a objeto.

Abstract

This work presents the implementation and evaluation of a practice for teaching Design Patterns using the Four-Component Instructional Design (4C/ID) model. The research aims to validate the effectiveness of a practical approach in learning these patterns, which are essential for developing reusable and maintainable software. The study involved the implementation of incremental activities in a Software Engineering course, combining theory and practice to facilitate concept assimilation. In addition to design patterns, the methodology explored SOLID principles, focusing on the principles of Single Responsibility, Open-Closed, and Dependency Inversion. The evaluation was conducted through an analysis of student performance and perception, highlighting the positive impact of the practical approach on learning. The results indicate that structured practice significantly contributes to the understanding and application of design patterns, suggesting improvements for future implementations of the methodology.

Keywords: Design Pattern, Teaching, Object Oriented Programming.

Sumário

1 Introdução

Gamma et al. (2000) nos fala sobre as dificuldades e desafios encontrados na criação de software reutilizável e flexível utilizando programação orientada a objetos. Então, definiram um conjunto de 23 padrões de projeto, que conforme (VAHLDICK; PABLO; PAOLO, 2019), um padrão de projeto descreve um problema e como soluciona-lo, considerando objetos e interfaces. Embora a empregabilidade de padrões de projeto auxilie na criação de software reutilizável, flexível e de fácil manutenção, (PILLAY, 2010) nos diz que o ensino e aprendizagem de desing patterns são tarefas difíceis.

Gómez-Martín et al. (2009) diz que a melhor maneira de aprender padrões de projeto é através de atividades práticas, pois a teoria pode se mostrar mais complexa visto que os alunos não apresentam o conhecimento necessário sobre programação orientado a objetos. Gómez-Martín et al. (2009) reforça ainda que atividades incrementais, que alterem o nível de dificuldade de uma para outra, se apresentam como uma boa estratégia para o ensino de padrões de projeto.

Um dos maiores desafios no ensino de padrões de projeto é a complexidade dos conceitos envolvidos, que exige dos alunos um alto grau de abstração e conhecimento prévio em engenharia de software e programação orientada a objetos. Estudos como o de (SILVA; MIRANDA; PEREIRA, 2020) indicam que muitos alunos enfrentam dificuldades para compreender e aplicar esses conceitos de forma eficaz sem uma base sólida em programação. Além disso, a introdução de padrões de projeto muitas vezes é percebida como uma tarefa muito abstrata.

Outro aspecto que contribui para a dificuldade no ensino de padrões de projeto é a falta de experiência prática dos alunos. Conforme (CRUZ; PAZOTI; MARACCI, 2020), muitos estudantes tem uma compreensão teórica dos padrões, mas lutam para aplicá-los em contextos reais do dia a dia do desenvolvimento de software. Esta diferença entre teoria e prática pode ser reduzida através de métodos de ensino mais dinâmicos e interativos, como o uso de projetos práticos, que tem mostrado ser eficaz para o entendimento dos conceitos de design patterns.

De acordo com Martin (2018): "Bons sistemas de software começam com o código limpo". Os princípios SOLID, propostos por Robert C. Matin, são cinco diretrizes que visam melhorar a qualidade do código em sistemas orientados a objetos. Esses princípios nos ajudam a criar software mais flexível, escalável, com uma manutenção simplificada, com código mais legível e compreensível (SILVA, 2023a).

O ensino de engenharia de software tem sido amplamente estudado na literatura, pois representa um desafio recorrente devido à necessidade de equilibrar conceitos teóricos

e aplicação prática. Diferentes abordagens pedagógicas vêm sendo exploradas para facilitar esse processo, como o uso de ferramentas educacionais, como a proposta por Cruz, Pazoti e Maracci (2020), que auxilia na implementação de padrões de projeto, ou até mesmo a utilização de jogos no processo de aprendizagem, conforme discutido no estudo de Silva, Miranda e Felski (2016).

Uma abordagem que também tem se destacado é o modelo Four-Component Instructional Design (4C/ID). Esse modelo tem sido amplamente utilizado na estruturação de cursos e treinamentos voltados para o desenvolvimento de habilidades complexas, pois organiza o aprendizado de forma progressiva e busca reduzir a carga cognitiva dos alunos (VAHLDICK; SCHOEFFEL; MOSER, 2020).

Diante desse cenário, este trabalho propõe e avalia uma abordagem baseada no modelo 4C/ID para o ensino de padrões de projeto e princípios SOLID. O foco está na aplicação prática e no desenvolvimento progressivo das habilidades dos alunos, estruturando o aprendizado em etapas que favoreçam a fixação dos conceitos e sua aplicação no desenvolvimento de software.

1.1 Objetivos

Apoiar o ensino de padrões de projeto e princípos SOLID em engenharia de software por meio de práticas estruturadas com base no modelo 4C/ID.

1.1.1 Objetivos Específicos

Este trabalho busca os seguintes objetivos:

- Identificar as principais dificuldades no ensino de padrões de projeto em engenharia de software;
- Definir uma abordagem pedagógica baseada no modelo 4C/ID para o ensino de padrões de projeto e princípios SOLID;
- Aplicar e avaliar a abordagem em uma disciplina de Engenharia de Software, analisando a percepção dos alunos e os impactos da metodologia.

2 Revisão da literatura e trabalhos relacionados

2.1 Padrões de Projeto

Padrões de projeto, ou Design Patterns, são soluções reutilizáveis para problemas comuns enfrentados no desenvolvimento de software (GAMMA et al., 2000). Segundo Gamma et al. (2000), pode-se dividir esses padrões em três tipos: padrões de criação, estruturais e comportamentais.

2.1.1 Padrões de Criação

Os padrões de criação tratam da maneira como os objetos são criados. Eles ajudam a tornar o sistema independente de como os objetos são instanciados, compostos ou representados (GAMMA et al., 2000). Exemplos:

- **Singleton**: Garante que uma classe tenha apenas uma instância e fornece um ponto de acesso global a essa instância (FREEMAN et al., 2004).
- **Factory Method**: Define uma interface para criar um objeto, mas permite que as subclasses alterem o tipo de objeto que será criado (SHVETS, 2020c).
- **Abstract Factory**: Fornece uma interface para criar famílias de objetos relacionados ou dependentes sem especificar suas classes concretas (GAMMA et al., 2000).
- **Builder**: Separa a construção de um objeto complexo de sua representação, permitindo que o mesmo processo de construção crie diferentes representações (FREEMAN et al., 2004).
- **Prototype**: Permite criar novos objetos a partir de um protótipo existente, clonando-o. É útil quando a criação de uma nova instância do objeto é custosa (SHVETS, 2020e).

2.1.2 Padrões Estruturais

Os padrões estruturais têm como objetivo principal fazer a composição das classes ou objetos. Eles ajudam a garantir que se obtenha novos funcionamentos a partir de objetos existentes (GAMMA et al., 2000). Exemplos:

- **Adapter**: Permite que a interface de uma classe seja usada como outra interface. É frequentemente usado para fazer com que classes com interfaces incompatíveis trabalhem juntas (FREEMAN et al., 2004).
- **Composite**: Compõe objetos em estruturas de árvore para representar hierarquias parte-todo. Permite que os clientes tratem objetos individuais e composições de objetos de maneira uniforme (SHVETS, 2020a).
- **Decorator**: Anexa responsabilidades adicionais a um objeto dinamicamente. Os decoradores fornecem uma alternativa flexível à subclasse para estender a funcionalidade (FREEMAN et al., 2004).
- **Facade**: Fornece uma interface simplificada para um conjunto de interfaces em um subsistema, facilitando seu uso (SHVETS, 2020b).
- **Flyweight**: Utiliza o compartilhamento para suportar eficientemente grandes quantidades de objetos de forma granular (GAMMA et al., 2000).
- **Proxy**: Fornece um substituto ou ponto através do qual um objeto pode controlar o acesso a outro (SHVETS, 2020f).

2.1.3 Padrões Comportamentais

Os padrões comportamentais estão relacionados com algoritmos e a atribuição de responsabilidades entre objetos (GAMMA et al., 2000). Exemplos:

- **Observer**: Define uma dependência de um-para-muitos entre objetos para que quando um objeto muda de estado, todos os seus dependentes sejam notificados e atualizados automaticamente (FREEMAN et al., 2004).
- **Strategy**: Define uma família de algoritmos, encapsula cada um deles e os torna intercambiáveis. Permite que o algoritmo varie independentemente dos clientes que o utilizam (SHVETS, 2020g).
- **Command**: Encapsula uma solicitação como um objeto, permitindo parametrizar clientes com diferentes solicitações, enfileirar ou fazer log de solicitações, e suportar operações que podem ser desfeitas (FREEMAN et al., 2004).
- **Chain of Responsibility**: Evita o acoplamento do remetente de uma solicitação ao seu receptor ao dar a mais de um objeto a chance de tratar a solicitação. Encadeia os objetos receptores e passa a solicitação ao longo da cadeia até que um objeto a trate (GAMMA et al., 2000).

- **Interpreter**: Dada uma linguagem, define uma representação para sua gramática junto com um interpretador que usa a representação para interpretar sentenças na linguagem (SHVETS, 2020d).
- **Iterator**: Fornece uma maneira de acessar sequencialmente os elementos de um objeto agregado sem expor sua representação subjacente (FREEMAN et al., 2004).
- **Mediator**: Define um objeto que encapsula a forma como um conjunto de objetos interage. O mediador promove o acoplamento fraco ao evitar que os objetos se refiram uns aos outros explicitamente, permitindo variar suas interações independentemente (GAMMA et al., 2000).
- **Memento**: Sem violar o encapsulamento, captura e externaliza o estado interno de um objeto para que o objeto possa ser restaurado a esse estado mais tarde (GAMMA et al., 2000).
- **State**: Permite que um objeto altere seu comportamento quando seu estado interno muda. O objeto parecerá ter mudado sua classe (FREEMAN et al., 2004).
- **Template Method**: Define o esqueleto de um algoritmo em uma operação, postergando a definição de alguns passos para subclasses. Permite que subclasses redefinam certos passos de um algoritmo sem alterar a estrutura do algoritmo (GAMMA et al., 2000).
- **Visitor**: Representa uma operação a ser executada nos elementos de uma estrutura de objeto. Permite definir uma nova operação sem mudar as classes dos elementos sobre os quais opera (SHVETS, 2020h).

Neste trabalho foi feita a escolha de alguns padrões que se baseou em sua relevância e aplicabilidade prática no desenvolvimento de software. Os padrões selecionados foram:

- **Singleton**: Este padrão é fundamental para garantir que uma classe tenha apenas uma instância e é amplamente utilizado em contextos onde um único ponto de controle é necessário (FREEMAN et al., 2004).
- **Abstract Factory**: Este padrão é essencial para criar famílias de objetos relacionados sem depender de suas classes concretas (GAMMA et al., 2000).
- **Adapter**: O Adapter foi escolhido por sua capacidade de permitir que interfaces incompatíveis trabalhem juntas (FREEMAN et al., 2004).
- **Composite**: Este padrão é importante para a composição de objetos em estruturas de árvore para representar hierarquias parte-todo (SHVETS, 2020a).

- **Observer**: O padrão Observer é crucial para implementar a notificação automática de mudanças de estado (FREEMAN et al., 2004).
- **Strategy**: A escolha do padrão Strategy se dá pela sua flexibilidade em encapsular algoritmos e permitir sua intercambialidade, promovendo a reutilização e manutenção do código (SHVETS, 2020g).

Esses padrões foram selecionados devido à sua importância no ensino de boas práticas de design de software, sua aplicação ampla em diversos contextos e sua capacidade de resolver problemas comuns de desenvolvimento de maneira eficiente e eficaz.

2.2 Princípios SOLID

Os princípios SOLID são um conjunto de cinco diretrizes que tem como objetivo melhorar a qualidade do código em sistemas que utilizam programação orientada a objetos. Criados por (MARTIN, 2018), esses princípios ajudam a criar software mais flexível, escalável e de fácil manutenção. Os cinco princípios são: Responsabilidade Única (Single Responsability), Aberto-Fechado (Open-Closed), Substituição de Liskov (Liskov Substitution), Segregação de Interface (Interface Segregation) e Inversão de Dependência (Dependency Inversion) (SILVA, 2023b).

2.2.1 Princípio da Responsabilidade Única (SRP)

O Princípio da Responsabilidade Única afirma que uma classe deve ter apenas uma razão para mudar, ou seja, deve ter apenas uma responsabilidade. Segundo (MARTIN, 2018), seguir este princípio ajuda a manter o código coeso e facilita a manutenção e evolução do sistema (SILVA, 2023b).

2.2.2 Princípio Aberto-Fechado (OCP)

O Princípio Aberto-Fechado propõe que as classes devem ser abertas para extensão, mas fechadas para modificação. Sendo assim, o comportamento de uma classe pode ser estendido sem alterar seu código-fonte original (MARTIN, 2018). Este princípio incentiva o uso de herança e interfaces para permitir a flexibilidade do código (MARTIN, 2000).

2.2.3 Princípio da Substituição de Liskov (LSP)

O Princípio da Substituição de Liskov, descrito por (MARTIN, 2018), estabelece que objetos de uma classe derivada devem poder substituir objetos de sua classe base

sem alterar o comportamento esperado do programa. Este princípio visa garantir que a hierarquia de classes seja usada corretamente, mantendo a integridade do sistema (SILVA, 2023b).

2.2.4 Princípio da Segregação de Interface (ISP)

O Princípio da Segregação de Interface afirma que os clientes não devem ser forçados a depender de interfaces que não utilizam. (MARTIN, 2018) propõe que é melhor ter várias interfaces específicas do cliente do que uma única interface geral. Isso ajuda a evitar a implementação de métodos desnecessários e promove a coesão das interfaces (MARTIN, 2000).

2.2.5 Princípio da Inversão de Dependência (DIP)

O Princípio da Inversão de Dependência sugere que módulos de alto nível não devem depender de módulos de baixo nível, mas ambos devem depender de abstrações. Além disso, essas abstrações não devem depender de detalhes, mas os detalhes devem depender das abstrações (MARTIN, 2018). Este princípio tem como objetivo promover o desacoplamento e a flexibilidade do sistema (SILVA, 2023b).

Neste trabalho abordaremos os princípios de Responsabilidade Única, Aberto-Fechado e Inversão de Dependência. Sua escolha foi baseada na sua importância na construção de software flexível e de fácil manutenção. O Princípio da Responsabilidade Única foi selecionado por promover a coesão, garantindo que cada classe tenha uma única responsabilidade (MARTIN, 2018). O Princípio Aberto-Fechado é crucial para permitir que o software seja estendido sem necessidade de modificar o código existente, promovendo a reutilização e a robustez do software (MARTIN, 2000). E o Princípio da Inversão de Dependência foi escolhido por seu papel fundamental no desacoplamento de módulos de alto e baixo nível, o que aumenta a flexibilidade e a capacidade de adaptação do sistema a mudanças futuras (MARTIN, 2018).

2.3 Ensino de Padrões de Projeto

O ensino de padrões de projeto é um desafio conhecido nas instituições de ensino superior, devido à alta complexidade desses conceitos e à necessidade de maturidade dos alunos em Engenharia de Software e Programação orientada a objetos. Diversos estudos apontam para a importância de estratégias pedagógicas inovadoras para promover a compreensão e a aplicação eficaz dos padrões de projeto.

Um dos métodos eficazes identificados é a utilização de projetos práticos, como o desenvolvimento de jogos. Essa abordagem, além de motivar os alunos, também fornece um contexto rico e interdisciplinar onde os conceitos de padrões de projeto podem ser aplicados de maneira concreta. Por exemplo, a criação de um jogo de corrida permitiu a aplicação prática de padrões como Singleton e Abstract Factory, facilitando a assimilação dos conceitos teóricos através de um projeto familiar e de interesse para os alunos (SILVA; MIRANDA; PEREIRA, 2020).

O uso de ferramentas educacionais também tem se mostrado eficaz para tornar o ensino de padrões de projeto mais dinâmico e acessível. Um exemplo disso é o LEPATTERN, uma ferramenta desenvolvida para auxiliar no aprendizado e aplicação desses padrões, permitindo que os alunos implementem estruturas de design de forma guiada e automatizada (CRUZ; PAZOTI; MARACCI, 2020). A utilização de plugins como o LEPATTERN possibilita que os estudantes compreendam melhor os conceitos ao visualizar a geração de código estruturado e comentado, reduzindo a dificuldade de assimilação e tornando o aprendizado mais interativo e prático.

Outro método destacado é a aplicação do modelo Four Component/Instructional Design (4C/ID), que visa reduzir a carga cognitiva dos alunos ao estruturar o material instrucional em componentes integrados (VAHLDICK; SCHOEFFEL; MOSER, 2020). Essa abordagem inclui o uso de estudos de caso, projetos e problemas práticos que incentivam os alunos a construir esquemas cognitivos através de experiências concretas. Esse modelo tem demonstrado sucesso ao aumentar o engajamento e a compreensão dos alunos sobre padrões de projeto (VAHLDICK; SCHOEFFEL; MOSER, 2020).

Porem, é importante reconhecer que o ensino de padrões de projeto deve ser adaptado às necessidades e ao nível de maturidade dos alunos. Alguns estudos sugerem que a introdução desses conceitos deve ser gradual e integrada ao longo do currículo, permitindo que os alunos desenvolvam uma compreensão profunda e prática ao longo do tempo (KöPPE, 2020). A estratégia baseada no desenvolvimento de jogos também corrobora essa abordagem gradual, permitindo que os alunos se familiarizem com os padrões de forma progressiva e aplicada (SILVEIRA, 2020).

Portanto, a escolha das praticas de ensino de padrões de projeto deve considerar tanto a motivação dos alunos quanto a eficácia das estratégias aplicadas, permitindo que os alunos não apenas compreendam os conceitos, mas também sejam capazes de usa-los de maneira correta e eficaz em seus futuros projetos.

2.4 Modelo Four-Component Instructional Design (4C/ID)

O Modelo Four-Component Instructional Design (4C/ID) é uma abordagem de design instrucional, desenvolvido para lidar com o ensino de tarefas complexas. O modelo 4C/ID foi desenvolvido por Jeroen J. G. van Merriënboer e é baseado em quatro componentes principais: Tarefas de Aprendizagem, Informação de Suporte, Informação Procedural e Prática de Parte-Tarefa. Essa abordagem busca reduzir a carga cognitiva dos alunos e favorecer um aprendizado progressivo e integrado (MERRIëNBOER, 2023).

2.4.1 Tarefas de Aprendizagem

As Tarefas de Aprendizagem são a espinha dorsal do modelo e representam atividades autênticas que os alunos realizam durante o processo para auxiliar no aprendizado habilidades mais complexas. Essas tarefas são organizadas em níveis progressivos de dificuldade, permitindo que os alunos adquiram habilidades de maneira gradual. Além disso, são planejadas para simular o máximo possível de situações reais, garantindo que o aprendizado seja significativo e aplicável (MERRIëNBOER, 2023).

2.4.2 Informação de Suporte

A Informação de Suporte se diz repeito a todo o conhecimento necessário para que os alunos possam executar as Tarefas de Aprendizagem. Esse componente inclui explicações teóricas, exemplos práticos e materiais complementares, como vídeos, livros e artigos, que ajudam na construção do conhecimento conceitual. Diferentemente da Informação Procedural, a Informação de Suporte não é fornecida no momento exato da execução das tarefas, mas sim como um recurso de estudo para consulta ao longo do aprendizado (VAHLDICK; SCHOEFFEL; MOSER, 2020).

2.4.3 Informação Procedural

A Informação Procedural refere-se ao passo a passo que auxilia os alunos na execução de atividades dentro das Tarefas de Aprendizagem. Esse componente inclui instruções passo a passo, guias de implementação e feedback imediato, garantindo que os alunos consigam realizar corretamente os procedimentos necessários. Esse tipo de informação é particularmente importante no início do aprendizado, pois fornece suporte para que os estudantes desenvolvam gradualmente autonomia na execução das tarefas (MERRIëNBOER, 2023).

2.4.4 Prática de Parte-Tarefa

A Prática de Parte-Tarefa consiste na repetição e automatização de componentes específicos das Tarefas de Aprendizagem. Esse componente permite que os alunos desenvolvam fluência em habilidades fundamentais antes de aplicá-las em contextos mais complexos. A repetição estruturada ajuda a consolidar o conhecimento e reduz a carga cognitiva durante a execução de tarefas mais avançadas, tornando o aprendizado mais eficiente (VAHLDICK; SCHOEFFEL; MOSER, 2020).

3 Metodologia

A metodologia adotada para este trabalho foi desenvolvida com base no modelo Four-Component Instructional Design (4C/ID), que visa estruturar o ensino de tarefas complexas através de componentes integrados. Este modelo foi escolhido visando reduzir a carga cognitiva dos alunos e facilitar a aquisição de conhecimentos complexos relacionados à programação orientada a objetos e, por sua vez, aos padrões de projeto.

O processo educativo foi realizado na disciplina de Engenharia de Software com alunos do 5º período da Universidade Federal do Maranhão e incluiu aulas teóricas e práticas sobre os princípios SOLID e Design Patterns. Durante as aulas, foram apresentados os conceitos fundamentais desses temas, acompanhados de exemplos de código e diagramas de classes, permitindo que os alunos visualizassem e compreendessem sua aplicação prática.

A estrutura do ensino seguiu os quatro componentes do modelo 4C/ID:

- **Tarefas de Aprendizagem**: No modelo 4C/ID, as tarefas de aprendizagem são o centro do processo educacional. Neste estudo, foram definidas duas tarefas principais para os alunos:

    1. **Primeira atividade**: Os alunos escolheram dois padrões de projeto para apresentar, explicando seus conceitos e aplicações por meio de diagramas e exemplos de código. Essa etapa serviu como um primeiro contato prático com os padrões de projeto.
    2. **Projeto final**: Os alunos receberam duas própostas de projetos previamente desenvolvidos, sem a aplicação de padrões de projeto ou conceitos de clean code. Eles deveriam escolher apenas um dos projetos e entregar um código refatorado e funcional, aplicando os princípios SOLID (especificamente os princípios **S, I e D**) e pelo menos dois padrões de projeto, como *Singleton, Abstract Factory, Adapter, Composite, Observer* ou *Strategy*.

- **Informação de Suporte e Informação Procedural**: A informação de suporte abrangeu todo o conhecimento teórico necessário para a realização das tarefas, incluindo aulas expositivas, materiais de apoio e exemplos práticos. Por outro lado, a informação procedural foi disponibilizada diretamente no contexto das atividades, por meio de guias de implementação, explicações detalhadas durante um acompanhamento em que os alunos apresentaram seus desenvolvimentos, esclareceram dúvidas e receberam feedbacks, além do auxílio oferecido extra-classe, proporcionando um suporte contínuo ao aprendizado e à aplicação dos padrões de projeto.

- **Prática de Parte-Tarefa**: A prática de parte-tarefa esteve presente ao longo do processo, permitindo que os alunos exercitassem partes específicas das tarefas antes da implementação completa. Isso ocorreu tanto nas apresentações iniciais, que exigiam a exploração de padrões de projeto isoladamente, quanto na refatoração do código, onde aplicavam gradualmente os conceitos aprendidos.

Os projetos foram escolhidos com o intuito de minimizar a complexidade, sempre focando no aprendizado dos padrões de projeto. Os projetos selecionados foram:

3.1 Projeto 1: Sistema de Gerenciamento de Biblioteca

Descrição: Desenvolver um sistema simples de gerenciamento de biblioteca onde os usuários podem buscar livros, fazer reservas e administrar seus empréstimos.

Requisitos:

- Cadastro de livros (título, autor, ISBN, categoria).
- Cadastro de usuários (nome, email, telefone).
- Sistema de busca de livros por título, autor ou categoria.
- Reserva de livros pelos usuários.

3.2 Projeto 2: Sistema de Gerenciamento de Tarefas

Descrição: Desenvolver um sistema simples de gerenciamento de tarefas onde os usuários podem criar, editar, marcar como concluídas e deletar tarefas.

Requisitos:

- Cadastro de tarefas (título, descrição, status).
- Edição de tarefas.
- Marcação de tarefas como concluídas.
- Exclusão de tarefas.

A aplicação prática desta metodologia foi avaliada em uma turma da disciplina de Engenharia de Software, e os resultados esperados incluem uma melhoria significativa na compreensão e aplicação dos padrões de projeto pelos alunos.

3.3 Formulário de Avaliação

Um formulário de avaliação foi elaborado para coletar feedback dos alunos sobre a eficácia da metodologia aplicada no ensino de padrões de projeto. Além de avaliar a percepção dos alunos sobre a didática utilizada, o formulário busca compreender a experiência prévia dos alunos com desenvolvimento de software, programação orientada a objetos e sua familiaridade com padrões de projeto e princípios SOLID.

Cada questão do formulário é estruturada para capturar a experiência dos alunos em diferentes aspectos do curso:

- **Experiência Prévia com Desenvolvimento de Software:** Identifica se o aluno já trabalhou na área de desenvolvimento de software e sua frequência de uso de programação orientada a objetos, o que ajuda a contextualizar suas respostas no restante do formulário.
- **Familiaridade com Padrões de Projeto e Princípios SOLID:** Avalia o nível de familiaridade prévia dos alunos com padrões de projeto e os princípios SOLID, proporcionando uma base para entender como o curso impactou o conhecimento existente dos alunos.
- **Eficiência da Metodologia:** Avalia se a abordagem de ensino adotada foi eficaz no aprendizado dos padrões de projeto.
- **Suficiência das Explicações:** Mede se as explicações teóricas e os exemplos dados foram adequados para a compreensão dos conceitos.
- **Complexidade das Atividades:** Avalia a percepção dos alunos sobre a dificuldade da atividade prática, que envolveu a refatoração de código.
- **Impacto do Projeto Prático:** Verifica se o projeto prático ajudou os alunos a entender melhor os padrões de projeto.
- **Aplicação dos Princípios SOLID:** Avalia se a prática dos princípios SOLID contribuiu para a criação de código mais reutilizável e flexível.
- **Confiança na Aplicação:** Mede o nível de confiança dos alunos em aplicar padrões de projeto após a experiência educativa.

As questões abertas no final do formulário permitem que os alunos expressem quais aspectos das atividades contribuíram mais para o aprendizado e forneçam sugestões para melhorar o ensino de padrões de projeto. Essas respostas são valiosas para ajustar e aprimorar futuras abordagens pedagógicas.

4 Resultados

Este capítulo apresenta os resultados da pesquisa realizada para avaliar a eficácia da metodologia proposta para o ensino de padrões de projeto e princípios SOLID. Os dados coletados foram analisados a fim de identificar o perfil dos participantes, sua familiaridade prévia com os conceitos abordados e a percepção sobre a metodologia aplicada.

4.1 Perfil dos Participantes

A turma tinha um total de 36 alunos; no entanto, a pesquisa contou com a participação de **23 alunos**, que responderam a um questionário estruturado para avaliar sua experiência prévia com desenvolvimento de software, programação orientada a objetos e padrões de projeto.

4.1.1 Experiência em Desenvolvimento de Software

Os dados coletados revelam que a maioria dos participantes (**78,3%**) **nunca exerceu atividades profissionais** na área de desenvolvimento de software. Apenas **17,4%** dos alunos atuam atualmente na área, enquanto **4,3%** já trabalharam anteriormente. A Figura 1 ilustra essa distribuição.

A estrutura dessa questão foi baseada no estudo de Silva, Miranda e Pereira (2020), que também investigou a experiência prévia dos alunos com desenvolvimento de software para compreender como esse fator influencia na assimilação dos padrões de projeto.

Figura 1 – Distribuição da experiência dos participantes no desenvolvimento de software.

4.1.2 Utilização da Programação Orientada a Objetos

A familiaridade com programação orientada a objetos variou entre os participantes. Conforme ilustrado na Figura 2, 30,4% dos alunos relataram utilizar POO com frequência, enquanto 30,4% afirmaram utilizar ocasionalmente e outros 34,8% raramente. Apenas **4,4%** nunca utilizaram essa abordagem.

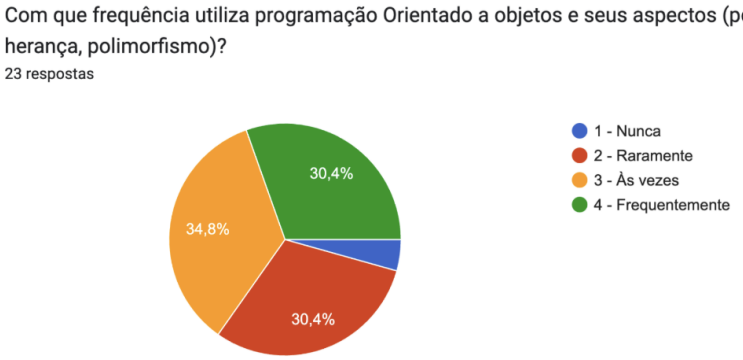

Figura 2 – Distribuição do uso da Programação Orientada a Objetos entre os participantes.

4.1.3 Experiência com Padrões de Projeto

Os resultados demonstram que **65,2%** dos participantes nunca haviam utilizado padrões de projeto anteriormente, enquanto **34,8%** relataram já ter aplicado esses conceitos ocasionalmente. A Figura 3 apresenta essa distribuição.

A formulação dessa questão foi baseada no estudo de Silva, Miranda e Pereira (2020), que investigou a familiaridade prévia dos alunos com padrões de projeto como um fator relevante para o aprendizado dessas técnicas.

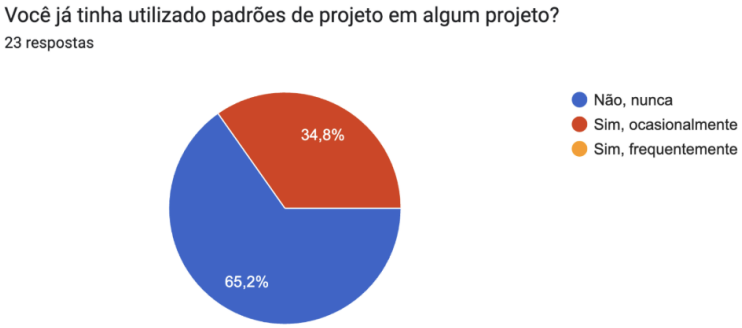

Figura 3 – Familiaridade dos participantes com padrões de projeto antes da atividade.

4.1.4 Conhecimento sobre os Princípios SOLID

Em relação aos princípios SOLID, verificou-se que **60,9%** dos participantes nunca haviam ouvido falar sobre esses conceitos antes da atividade, enquanto **39,1%** afirmaram ter conhecimento prévio, mas sem domínio aprofundado. Nenhum dos participantes declarou possuir conhecimento avançado sobre princípios SOLID. A Figura 4 ilustra esses resultados.

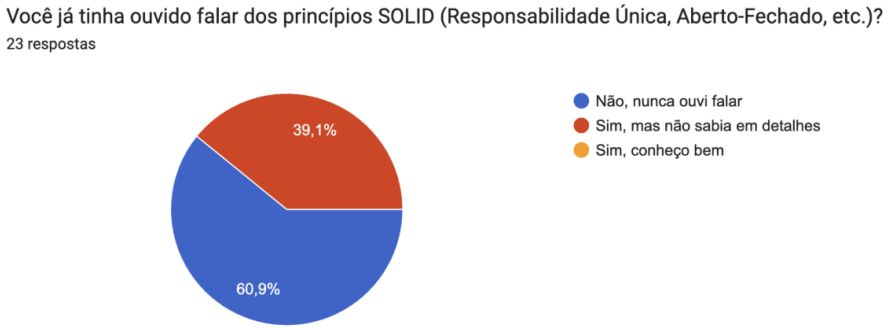

Figura 4 – Familiaridade dos participantes com os princípios SOLID antes da atividade.

4.2 Avaliação da Metodologia Aplicada

Os participantes foram questionados sobre a eficácia da metodologia utilizada para o ensino de padrões de projeto e princípios SOLID, bem como sobre a clareza das explicações teóricas e a complexidade das atividades propostas.

As perguntas a seguir foram inspiradas no estudo de Vahldick, Schoeffel e Moser (2020), que avaliou metodologias de ensino para padrões de projeto utilizando o modelo 4C/ID. No entanto, as questões foram adaptadas para o contexto específico desta pesquisa, considerando as atividades realizadas durante o processo de aprendizagem.

4.2.1 Eficiência da Metodologia no Ensino de Padrões de Projeto

A maioria dos participantes (**78,3%**) avaliou a metodologia como **eficiente**, enquanto **21,7%** a consideraram **moderadamente eficiente**. Nenhum dos respondentes classificou a abordagem como ineficaz. A Figura 5 apresenta os resultados.

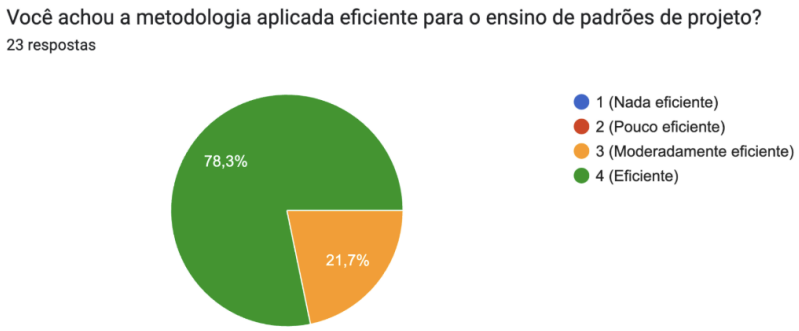

Figura 5 – Avaliação da eficiência da metodologia aplicada.

4.2.2 Clareza das Explicações Teóricas e Exemplos

Quanto à clareza das explicações teóricas e exemplos práticos apresentados em sala, **39,1%** dos participantes as classificaram como **suficientes**, enquanto **52,2%** consideraram-nas **moderadamente suficientes** e **8,7%** as avaliaram como **pouco suficientes**. Nenhum participante as considerou completamente insuficientes. A Figura 6 apresenta essa distribuição.

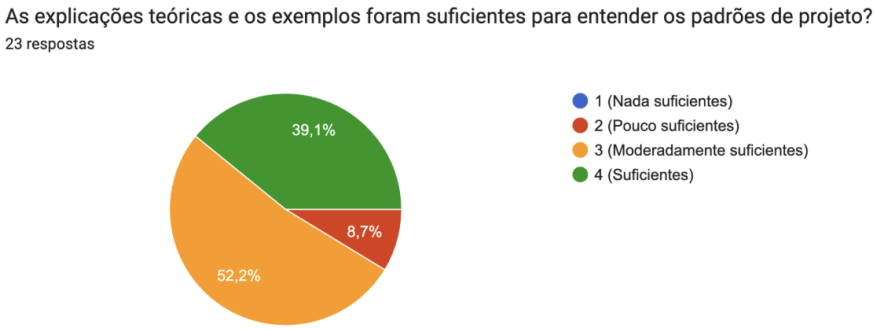

Figura 6 – Avaliação da clareza das explicações teóricas e exemplos práticos.

4.2.3 Complexidade da Primeira Atividade

A percepção da complexidade da atividade inicial variou entre os participantes: **69,6%** avaliaram-na como **moderada**, **21,7%** a consideraram **fácil** e **8,7%** a classificaram como **difícil**. Nenhum respondente julgou a atividade como "muito fácil". A Figura 7 ilustra esses resultados.

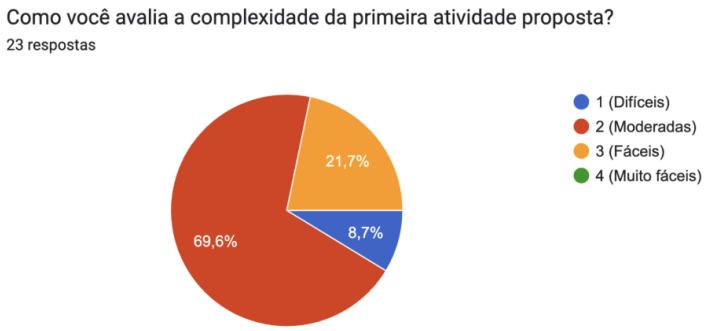

Figura 7 – Avaliação da complexidade da primeira atividade prática.

4.2.4 Impacto do Projeto Prático no Entendimento dos Padrões de Projeto

Os participantes foram questionados se o projeto prático auxiliou no entendimento dos padrões de projeto. A maioria (**78,3%**) respondeu que o projeto **ajudou bastante**, enquanto **21,7%** afirmaram que **ajudou moderadamente**. Nenhum participante indicou que o projeto ajudou pouco ou que não ajudou. A Figura 8 ilustra os resultados dessa questão.

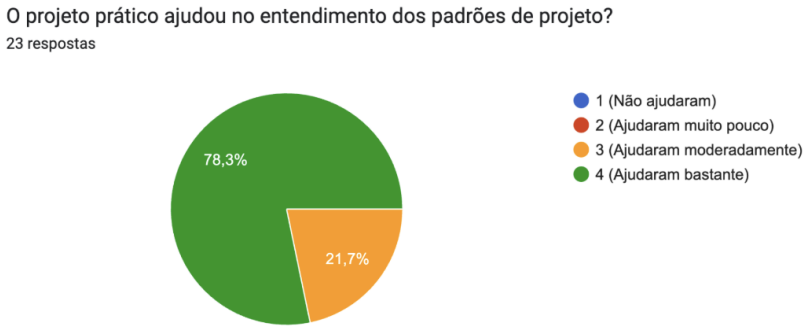

Figura 8 – Percepção dos participantes sobre o impacto do projeto prático no entendimento dos padrões de projeto.

4.2.5 Impacto dos Princípios SOLID na Criação de Código

A influência dos princípios SOLID na construção de código mais reutilizável e flexível foi destacada por **69,6%** dos respondentes, que afirmaram que os princípios **auxiliaram significativamente** nesse aspecto. Outros **30,4%** consideraram que o impacto foi **moderado**. Nenhum participante indicou que os princípios SOLID foram irrelevantes para a qualidade do código produzido. A Figura 9 apresenta essa análise.

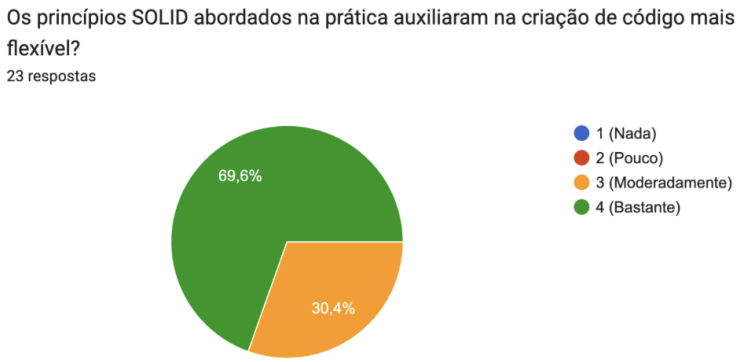

Figura 9 – Impacto dos princípios SOLID na criação de código reutilizável e flexível.

4.2.6 Confiança na Aplicação de Padrões de Projeto

Os participantes também foram questionados sobre sua confiança em aplicar padrões de projeto após a experiência proposta. A maioria dos respondentes, aproximadamente (**74%**) se declarou **moderadamente confiante**, enquanto **13%** se consideraram **confiantes** e **13%** relataram estar **pouco confiantes**. Nenhum participante se classificou como totalmente inseguro. A Figura 10 ilustra esses resultados.

Figura 10 – Nível de confiança dos participantes na aplicação de padrões de projeto após a experiência.

4.2.7 Importância da Prática no Aprendizado de Padrões de Projeto

Os participantes destacaram, de maneira quase que unânime, que a prática foi o aspecto mais relevante para a assimilação dos padrões de projeto. A possibilidade de aplicar os conceitos teóricos diretamente no desenvolvimento do projeto proporcionou uma compreensão mais profunda e concreta do tema.

Dentre os principais pontos mencionados, destacam-se:

- **Implementação prática**: a necessidade de aplicar os padrões diretamente no código permitiu entender melhor sua utilidade e funcionamento;
- **Organização do código**: a separação em módulos e classes ajudou na estruturação e legibilidade do projeto;
- **Interação com os colegas**: o trabalho em grupo e a necessidade de explicar as decisões tomadas consolidaram o aprendizado;
- **Testes e melhorias**: o desenvolvimento contínuo e a revisão antes da apresentação final permitiram ajustes e aprimoramento das implementações;
- **Relação entre teoria e prática**: a aplicação prática dos conceitos teóricos estudados foi considerada essencial para a retenção do conhecimento.

Os relatos indicam que, mais do que apenas estudar os padrões, a necessidade de implementá-los, testar suas aplicações e apresentar os resultados contribuiu significativamente para a fixação do conteúdo. O desenvolvimento do projeto se mostrou uma estratégia eficiente no ensino de padrões de projeto.

4.2.8 Sugestões para Melhorar o Ensino de Padrões de Projeto

Os participantes sugeriram diversas melhorias para o ensino de padrões de projeto, com forte ênfase no aumento da prática e na aplicação de exemplos mais concretos. As principais recomendações foram:

- **Criação de projetos do zero**: permitir que os alunos desenvolvam um sistema completo desde o início, aplicando os padrões conforme necessário;
- **Mais exemplos e comparações**: apresentar mais exemplos de códigos com o antes e depois da aplicação dos padrões.
- **Aplicação em cenários reais**: explorar casos de uso do mercado e frameworks conhecidos para contextualizar a importância dos padrões;
- **Acompanhamento contínuo**: adicionar mais checkpoints durante a execução do projeto para garantir um melhor acompanhamento da evolução dos alunos;
- **Aulas mais dinâmicas e focadas na prática**: reduzir a carga de conceitos puramente teóricos e dedicar mais tempo a explicações práticas, com mais demonstrações aplicadas e exercícios interativos que permitam aos alunos visualizar e testar os padrões em diferentes contextos;

- **Liberdade para refatoração**: incentivar os alunos a aprimorar o código base para poder permitir maior experimentação.

A partir dessas sugestões, percebe-se que os estudantes valorizam uma abordagem mais aplicada e interativa para o ensino de padrões de projeto. Mostrar mais exemplos práticos e aumentar o acompanhamento durante o projeto podem contribuir para um aprendizado mais eficiente e motivador.

4.2.9 Observações Gerais

Ao final do questionário, foi disponibilizado um espaço para que os alunos comentassem de forma espontânea suas opiniões. Os participantes fizeram comentários positivos sobre a experiência com o projeto, destacando sua importância para o aprendizado prático e o desenvolvimento profissional. Muitos mencionaram que a aplicação dos padrões de projeto contribuiu para a compreensão da estrutura e manutenção de códigos mais complexos.

Alguns alunos enfatizaram que a iniciativa foi valiosa, pois há poucas oportunidades para esse tipo de prática ao longo da formação acadêmica. Além disso, houve um reconhecimento da didática do professor e do monitor envolvido no processo.

Sugestões foram feitas para que futuras edições do projeto incluam mais momentos de prática e esclarecimento de dúvidas em sala de aula, além de um ritmo um pouco mais desacelerado para facilitar a assimilação dos conceitos. No geral, a experiência foi vista como enriquecedora e relevante para a formação na área de desenvolvimento de software.

5 Discussão

Os resultados da pesquisa são analisados e interpretados com base na literatura sobre o ensino de padrões de projeto e princípios SOLID usando o modelo 4C/ID. A discussão aborda o impacto da metodologia utilizada, as dificuldades relatadas pelos alunos participantes e as sugestões propostas para aprimorar o ensino desses conceitos.

5.1 Impacto da Metodologia Baseada em Prática

Os resultados mostram que a abordagem focada na aplicação prática dos padrões de projeto foi bem aceita pelos alunos. A maioria dos deles afirmou que a prática foi fundamental para entender os conceitos, o que está de acordo com os estudos de Gómez-Martín et al. (2009) e Silveira (2020), que destacam a importância do aprendizado ativo para a retenção de conhecimento sobre padrões de projeto.

A necessidade de aplicar e explicar os padrões durante o desenvolvimento do projeto ajudou os alunos a compreender melhor os princípios estudados. Além disso, o trabalho em grupo e a apresentação final permitiram a troca de conhecimento entre os participantes, reforçando o que foi apontado por Vahldick, Pablo e Paolo (2019) sobre os benefícios de metodologias colaborativas no ensino de engenharia de software.

5.1.1 Vantagens da Abordagem Baseada em Prática

A metodologia aplicada neste estudo, baseada no modelo Four-Component Instructional Design (4C/ID), demonstrou vantagens significativas para o ensino de padrões de projeto e princípios SOLID. A estrutura progressiva do modelo, que divide o aprendizado em tarefas de complexidade crescente, facilitou a assimilação dos conceitos, reduzindo a carga cognitiva dos alunos (VAHLDICK; SCHOEFFEL; MOSER, 2020).

A necessidade de aplicar os padrões diretamente na refatoração de código permitiu que os alunos compreendessem melhor os benefícios práticos dessas técnicas, tornando o aprendizado mais concreto e alinhado com a realidade do desenvolvimento de software. Além disso, o uso de atividades incrementais reforçou a fixação do conhecimento, como destacado por Gómez-Martín et al. (2009), que apontam que abordagens práticas são mais eficazes do que a simples exposição teórica.

Outro benefício foi a organização do ensino em fases estruturadas, característica central do modelo 4C/ID. A divisão entre informação de suporte e informação procedural

ajudou os alunos a desenvolverem autonomia na aplicação dos padrões, enquanto a prática de parte-tarefa garantiu a repetição necessária para consolidar os conceitos antes da implementação completa nos projetos(MERRIëNBOER, 2023).

Por fim, os resultados evidenciaram um aumento na confiança dos alunos para utilizar padrões de projeto em seus próprios códigos, alinhando-se ao que foi observado por Vahldick, Pablo e Paolo (2019), que destacam que metodologias progressivas contribuem para um aprendizado mais efetivo e duradouro.

5.2 Dificuldades Enfrentadas pelos Alunos

Embora a metodologia tenha sido bem aceita, alguns participantes relataram dificuldades. Muitos tiveram problemas para entender e aplicar alguns princípios SOLID, principalmente os que envolvem modularidade e separação de responsabilidades. Isso está de acordo com Silva (2023a), que destaca que a abstração dos princípios SOLID pode ser desafiadora para iniciantes e exige uma base sólida em design de software.

Outro ponto levantado foi o tempo para aprender e aplicar os conceitos de forma estruturada. Alguns alunos sugeriram o uso de mini projetos ao longo do curso para reforçar os padrões de maneira gradual, o que está alinhado com Vahldick, Schoeffel e Moser (2020), que propõe o modelo 4C/ID como uma abordagem eficaz para fragmentar o ensino de padrões de projeto e otimizar a assimilação do conteúdo.

5.3 Sugestões dos Participantes e Possíveis Melhorias

Os alunos sugeriram várias melhorias para o ensino de padrões de projeto, principalmente a inclusão de mais exemplos práticos, mais tempo para a realização dos projetos e um acompanhamento mais próximo por meio de checkpoints. Essas sugestões estão alinhadas com Vahldick, Schoeffel e Moser (2020), que destaca a importância de um ensino estruturado em etapas, permitindo que os alunos assimilem os conceitos gradualmente e os apliquem de forma mais eficiente. O modelo 4C/ID enfatiza a necessidade de dividir o aprendizado em componentes menores, proporcionando um desenvolvimento progressivo das habilidades relacionadas à aplicação dos padrões de projeto.

Além disso, alguns participantes destacaram a importância de uma abordagem mais orientada à prática, sugerindo atividades que incentivem a experimentação e a aplicação direta dos padrões de projeto. Estudos como os de Silveira (2020) indicam que métodos baseados em resolução de problemas e estudo de casos reais contribuem para uma melhor assimilação dos conceitos. Quando combinadas com a estruturação do ensino baseada no modelo 4C/ID, essas estratégias possibilitam um aprendizado mais progressivo e eficaz,

permitindo que os alunos desenvolvam suas habilidades de forma mais natural e aplicada ao contexto do desenvolvimento de software.

5.4 Considerações sobre a Efetividade da Metodologia

A análise dos dados indica que a abordagem focada na aplicação prática dos padrões de projeto teve um impacto positivo para a maioria dos participantes. A combinação entre teoria e prática foi fundamental para a compreensão dos conceitos, reforçando a importância do aprendizado experiencial. Essa estratégia está alinhada com o modelo 4C/ID, que propõe a divisão do ensino em etapas progressivas para facilitar a assimilação e a aplicação dos conhecimentos de forma estruturada.

Por outro lado, os resultados também apontam a necessidade de ajustes na organização do curso, especialmente na forma como o conteúdo é apresentado e no suporte oferecido durante a implementação dos padrões. A introdução de atividades menores e a adoção de um cronograma mais flexível podem tornar o aprendizado mais acessível e equilibrado, permitindo que os alunos avancem no domínio dos padrões de projeto de maneira mais natural e eficiente

5.5 Limitações do Estudo

Este estudo apresenta algumas limitações que devem ser consideradas. Uma delas diz respeito à diversidade de níveis de experiência dos participantes, o que pode ter influenciado a forma como absorveram e aplicaram os padrões de projeto. Alunos com maior familiaridade com desenvolvimento de software, especialmente com Programação Orientada a Objetos (POO), tiveram mais facilidade na implementação, enquanto aqueles com menos experiência nessa abordagem enfrentaram desafios adicionais. Como os padrões de projeto são fortemente baseados em POO, a falta de uma base sólida nesse paradigma pode ter dificultado a compreensão e aplicação eficaz dos conceitos.

Outra limitação está na complexidade dos padrões abordados, que exigem um tempo maior para assimilação e prática. A adoção de uma abordagem progressiva, como propõe o modelo 4C/ID, pode ajudar a estruturar melhor o aprendizado, permitindo que os alunos avancem de forma mais gradual, consolidem melhor os princípios da POO e desenvolvam uma compreensão mais profunda dos padrões antes de aplicá-los em projetos mais complexos.

5.6 Implicações para o Ensino de Padrões de Projeto

Os resultados desta pesquisa trazem insights importantes para tornar o ensino de padrões de projeto mais eficiente e acessível. A prática contínua se mostrou essencial para a assimilação dos conceitos, reforçando a importância de metodologias que incentivem a experimentação e a aplicação em cenários reais. Isso está alinhado com o modelo 4C/ID, que sugere dividir o aprendizado em etapas menores e progressivas, facilitando o desenvolvimento das habilidades necessárias para a aplicação dos padrões.

Com base nas sugestões dos participantes e na literatura existente, algumas melhorias podem ser implementadas no curso:

- **Exercícios práticos curtos e frequentes**, para reforçar a compreensão dos conceitos antes da aplicação em projetos mais complexos;
- **Exemplos de código reais**, que mostrem como os padrões de projeto melhoram a estrutura e a manutenção do software;
- **Checkpoints mais frequentes**, garantindo que os alunos recebam feedback contínuo e possam ajustar seu aprendizado ao longo do processo;
- **Ritmo de aprendizado ajustável**, permitindo mais tempo para a absorção dos conceitos mais abstratos.

Dessa forma, o ensino de padrões de projeto pode se tornar mais acessível e eficiente, preparando melhor os alunos para desafios práticos no desenvolvimento de software.

6 Considerações Finais

Este estudo teve como objetivo avaliar a eficácia de uma abordagem baseada na aplicação prática para o ensino de padrões de projeto e princípios SOLID, utilizando o modelo 4C/ID como base para a estruturação do aprendizado. A pesquisa analisou a percepção dos alunos em relação à metodologia utilizada, identificando os principais desafios enfrentados, bem como sugestões para aprimorar o ensino desses conceitos.

Os resultados demonstraram que uma maior ênfase na prática foi um fator determinante para o entendimento dos padrões de projeto, permitindo que os alunos compreendessem melhor sua aplicação no desenvolvimento de software. A necessidade de implementar os padrões diretamente no código e justificá-los durante o desenvolvimento do projeto proporcionou uma experiência mais concreta e alinhada às necessidades do mercado. Além disso, a interação em equipe e as apresentações dos projetos contribuíram para a troca de conhecimento e o reforço do aprendizado, o que está em de acordo com os achados de Silveira (2020) e Gómez-Martín et al. (2009), que destacam a importância do aprendizado ativo e da colaboração para a fixação dos conceitos.

Apesar da aceitação geral da metodologia, algumas dificuldades foram relatadas pelos participantes, principalmente em relação à compreensão de certos princípios SOLID e à aplicação estruturada dos padrões de projeto. A falta de uma base sólida em Programação Orientada a Objetos foi um dos fatores que influenciaram esses desafios, destacando a importância de um ensino progressivo que fortaleça esse conhecimento antes da introdução dos padrões. Esse achado está alinhado com Silva (2023a), que ressalta que a compreensão dos princípios SOLID depende diretamente do domínio de conceitos fundamentais da POO.

A análise dos dados reforça a eficácia do modelo 4C/ID para o ensino de padrões de projeto, pois sua abordagem estruturada permite que os alunos desenvolvam habilidades de forma progressiva, consolidando conceitos fundamentais antes de avançar para aplicações mais complexas. Estudos como os de Vahldick, Schoeffel e Moser (2020) e Vahldick, Pablo e Paolo (2019) sustentam que o ensino baseado no modelo 4C/ID possibilita um aprendizado mais eficiente, uma vez que fragmenta o conteúdo em níveis de complexidade crescente. Sugere-se que futuras implementações da metodologia incluam atividades práticas menores ao longo do curso, um acompanhamento mais próximo do desenvolvimento dos alunos por meio de checkpoints e um cronograma maior e mais flexível para a assimilação dos conceitos.

Como limitações do estudo, destaca-se o curto período de avaliação, que impossibilita medir os efeitos da metodologia no longo prazo. Além disso, a diversidade

de níveis de experiência entre os participantes pode ter influenciado a forma como os conceitos foram absorvidos, tornando essencial um acompanhamento mais individualizado para garantir um aprendizado uniforme. Trabalhos como os de Silva, Miranda e Felski (2016) indicam que o ensino de padrões de projeto se torna mais efetivo quando há um acompanhamento contínuo da evolução dos alunos, o que reforça a necessidade de ajustes na metodologia utilizada.

Para estudos futuros, sugere-se a ampliação do tempo dedicado à prática dos padrões de projeto ao longo do curso, permitindo que os alunos tenham mais oportunidades de experimentar e consolidar os conceitos estudados. Além disso, a adoção de atividades progressivas, seguindo os princípios do modelo 4C/ID, pode facilitar a assimilação dos padrões, garantindo que a base em Programação Orientada a Objetos seja fortalecida antes da introdução de conceitos mais avançados.

Outra recomendação é o aprimoramento das estratégias de acompanhamento e suporte aos alunos, garantindo que dificuldades sejam identificadas e corrigidas ao longo do processo. A adoção de checkpoints mais frequentes e a oferta de momentos específicos para revisões e ajustes no código podem tornar o aprendizado mais eficiente e acessível para todos os níveis de experiência.

Também para estudos futuros, é recomendável investigar a aplicação de práticas estruturadas em outras abordagens de ensino. Isso pode incluir, por exemplo, a utilização de jogos ou ferramentas de aprendizagem, a fim de determinar como essas diferentes estratégias podem complementar o modelo 4C/ID e potencializar o ensino de padrões de projeto e princípios SOLID.

Com isso, espera-se que este estudo contribua para o aprimoramento do ensino de padrões de projeto, fornecendo subsídios para metodologias mais eficazes na formação de desenvolvedores de software mais preparados para os desafios do mercado.

Referências

CRUZ, C. T. da; PAZOTI, M. A.; MARACCI, F. V. Lepattern - uma ferramenta para ensino de padrões de projeto. *Revista de Ensino de Engenharia de Software*, 2020. Citado 3 vezes nas páginas 6, 7 e 13.

FREEMAN, E.; FREEMAN, E.; SIERRA, K.; BATES, B. *Head First Design Patterns*. [S.l.]: O'Reilly Media, 2004. Citado 4 vezes nas páginas 8, 9, 10 e 11.

GAMMA, E.; HELM, R.; JOHNSON, R.; VLISSIDES, J. *Padrões de Projeto: Soluções reutilizáveis de software orientado a objetos*. [S.l.]: Bookman, 2000. v. 11. Citado 4 vezes nas páginas 6, 8, 9 e 10.

GóMEZ-MARTíN; ANTONIO, M.; JIMéNEZ-DíAZ; GUILLERMO; ARROYO; JAVIER. Teaching design patterns using a family of games. 2009. Citado 3 vezes nas páginas 6, 27 e 31.

KöPPE, C. A pattern language for teaching design patterns. *Revista de Ensino de Engenharia de Software*, 2020. Citado na página 13.

MARTIN, R. C. *Agile Software Development: Principles, Patterns, and Practices*. Upper Saddle River, NJ: Prentice Hall, 2000. ISBN 9780135974445. Disponível em: <https://www.oreilly.com/library/view/agile-principles-patterns/9780135974445/>. Citado 2 vezes nas páginas 11 e 12.

MARTIN, R. C. *Clean Architecture: A Craftsman's Guide to Software Structure and Design*. [S.l.]: Pearson Education, 2018. Citado 3 vezes nas páginas 6, 11 e 12.

MERRIëNBOER, J. J. G. van. *Four-Component Instructional Design (4C/ID) Model*. 2023. Accessed: 2024-06-17. Disponível em: <https://www.4cid.org/>. Citado 2 vezes nas páginas 14 e 28.

PILLAY, N. Teaching design patterns. 2010. Citado na página 6.

SHVETS, A. *Composite Design Pattern*. 2020. <https://refactoring.guru/design-patterns/composite>. Citado 2 vezes nas páginas 9 e 10.

SHVETS, A. *Facade Design Pattern*. 2020. <https://refactoring.guru/design-patterns/facade>. Citado na página 9.

SHVETS, A. *Factory Method Design Pattern*. 2020. <https://refactoring.guru/design-patterns/factory-method>. Citado na página 8.

SHVETS, A. *Interpreter Design Pattern*. 2020. <https://refactoring.guru/design-patterns/interpreter>. Citado na página 10.

SHVETS, A. *Prototype Design Pattern*. 2020. <https://refactoring.guru/design-patterns/prototype>. Citado na página 8.

SHVETS, A. *Proxy Design Pattern*. 2020. <https://refactoring.guru/design-patterns/proxy>. Citado na página 9.

SHVETS, A. *Strategy Design Pattern*. 2020. <https://refactoring.guru/design-patterns/strategy>. Citado 2 vezes nas páginas 9 e 11.

SHVETS, A. *Visitor Design Pattern*. 2020. <https://refactoring.guru/design-patterns/visitor>. Citado na página 10.

SILVA, L. Introdução aos princípios do solid. 2023. Disponível em: <https://www.dio.me/articles/introducao-aos-principios-do-solid>. Citado 3 vezes nas páginas 6, 28 e 31.

SILVA, L. Introdução aos princípios do solid. *Digital Innovation One*, 2023. Disponível em: <https://www.dio.me/articles/introducao-aos-principios-do-solid>. Citado 2 vezes nas páginas 11 e 12.

SILVA, R. C. da; MIRANDA, E. M. de; FELSKI, P. T. Uma experiência no ensino de padrões de projeto utilizando jogo no processo de ensino-aprendizagem. 2016. Citado 2 vezes nas páginas 7 e 32.

SILVA, R. C. da; MIRANDA, E. M. de; PEREIRA, T. F. Uma experiência no ensino de padrões de projeto. *Revista de Ensino de Engenharia de Software*, 2020. Citado 4 vezes nas páginas 6, 13, 19 e 20.

SILVEIRA, I. F. A game development-based strategy for teaching software. *Revista de Ensino de Engenharia de Software*, 2020. Citado 4 vezes nas páginas 13, 27, 28 e 31.

VAHLDICK, A.; PABLO, S.; PAOLO, M. Metodologia de ensino de padrões de projeto baseado no modelo 4c/id. 2019. Citado 4 vezes nas páginas 6, 27, 28 e 31.

VAHLDICK, A.; SCHOEFFEL, P.; MOSER, P. Metodologia de ensino de padrões de projeto baseado no 4cid. *Revista de Ensino de Engenharia de Software*, 2020. Citado 7 vezes nas páginas 7, 13, 14, 15, 21, 28 e 31.

Anexos

Anexo A: Respostas ao Formulário de Avaliação

Data/Hora: 21/02/2025 11:51:59
Trabalha com desenvolvimento de software? Não, nunca
Frequência de uso de POO: 2 - Raramente
Utiliza padrões de projeto? Não, nunca
Conhecimento sobre princípios SOLID: Não, nunca ouvi falar
Metodologia eficiente para ensino? 4 - Eficiente
Explicações teóricas suficientes? 4 - Suficientes
Complexidade da primeira atividade: 2 - Moderadas
Projeto prático ajudou no entendimento? 4 - Ajudaram bastante
Princípios SOLID ajudaram na criação de código? 4 - Bastante
Confiança em aplicar padrões de projeto: 3 - Moderadamente confiante
Aspectos que mais contribuíram: Acredito que o incentivo à parte prática fez com que o conhecimento se fixasse melhor.
Sugestões para melhorias: Acho que buscar uma forma mais lúdica de ensinar a parte teórica ajudaria no aprendizado.
Observações gerais: —

Data/Hora: 21/02/2025 11:58:59
Trabalha com desenvolvimento de software? Não, nunca
Frequência de uso de POO: 2 - Raramente
Utiliza padrões de projeto? Não, nunca
Conhecimento sobre princípios SOLID: Não, nunca ouvi falar
Metodologia eficiente para ensino? 4 - Eficiente
Explicações teóricas suficientes? 4 - Suficientes
Complexidade da primeira atividade: 1 - Difíceis
Projeto prático ajudou no entendimento? 4 - Ajudaram bastante
Princípios SOLID ajudaram na criação de código? 3 - Moderadamente
Confiança em aplicar padrões de projeto: 3 - Moderadamente confiante
Aspectos que mais contribuíram: Construção de diagramas de classes com ou sem os padrões.
Sugestões para melhorias: Mais comparações de códigos usando e n usando
Observações gerais: —

Data/Hora: 21/02/2025 12:08:00
Trabalha com desenvolvimento de software? Não, nunca
Frequência de uso de POO: 3 - Às vezes
Utiliza padrões de projeto? Não, nunca
Conhecimento sobre princípios SOLID: Não, nunca ouvi falar

Metodologia eficiente para ensino? 4 - Eficiente
Explicações teóricas suficientes? 3 - Moderadamente suficientes
Complexidade da primeira atividade: 2 - Moderadas
Projeto prático ajudou no entendimento? 4 - Ajudaram bastante
Princípios SOLID ajudaram na criação de código? 4 - Bastante
Confiança em aplicar padrões de projeto: 2 - Pouco confiante
Aspectos que mais contribuíram: A parte prática e o momento antes da apresentação final me ajudaram a rever algumas coisas do código
Sugestões para melhorias: Acredito que além de mostrar alguns exemplos simples, um exemplo mais elaborado (onde não seria tão óbvio aplicar os princípios) poderia ajudar a ver sob uma outra perspectiva os conceitos.
Observações gerais: Foi bem legal a gente pegar um código já pronto e mexer nele. Iniciar um projeto é sempre fácil, sempre tem uma explosão de ideias do que e como implementar. Porém melhorar um código já pronto precisa de uma certa paciência. Trabalhar em equipe também foi bem interessante.

Data/Hora: 21/02/2025 12:08:35
Trabalha com desenvolvimento de software? Sim, atualmente
Frequência de uso de POO: 4 - Frequentemente
Utiliza padrões de projeto? Sim, ocasionalmente
Conhecimento sobre princípios SOLID: Sim, mas não sabia em detalhes
Metodologia eficiente para ensino? 4 - Eficiente
Explicações teóricas suficientes? 3 - Moderadamente suficientes
Complexidade da primeira atividade: 3 - Fáceis
Projeto prático ajudou no entendimento? 4 - Ajudaram bastante
Princípios SOLID ajudaram na criação de código? 4 - Bastante
Confiança em aplicar padrões de projeto: 3 - Moderadamente confiante
Aspectos que mais contribuíram: Eu sempre fui melhor em aprender na prática, então ter que fazer a pesquisa ao mesmo tempo que eu aplicava essas questões ajudou muito.
Sugestões para melhorias: A aula explicando os padrões não foi tão eficiente. Na minha visão deveria ter uma prática básica relacionada a cada padrão z de forma bem rápida, só para poder fixar na mente do aluno como de fato funciona.
Observações gerais: —

Data/Hora: 21/02/2025 12:10:26
Trabalha com desenvolvimento de software? Não, nunca
Frequência de uso de POO: 4 - Frequentemente
Utiliza padrões de projeto? Não, nunca
Conhecimento sobre princípios SOLID: Não, nunca ouvi falar
Metodologia eficiente para ensino? 4 - Eficiente

Explicações teóricas suficientes? 2 - Pouco suficientes
Complexidade da primeira atividade: 2 - Moderadas
Projeto prático ajudou no entendimento? 3 - Ajudaram moderadamente
Princípios SOLID ajudaram na criação de código? 3 - moderadamente
Confiança em aplicar padrões de projeto: 2 - Pouco confiante
Aspectos que mais contribuíram: Dinâmica em grupo
Sugestões para melhorias: A disciplina em si já se trata de um conteúdo muito maçante, no sentido que são muito conteúdo teórico, o que não se distancia das demais disciplinas. O problema em si é que se trata de uma abstração muito grande, pq em Cálculo, por exemplo, conceitos são aplicados da maneira que devem ser. Diferente dos padrões de projeto que de uma certa forma é aplicado na visão que a pessoa acha que é correto. Então a minha sugestão é aumenta a dinâmica e reduzir conceitos que de certa forma só se aplicam na teoria, e que não tem impacto na vida real.
Observações gerais: Obrigado

Data/Hora: 21/02/2025 12:52:40
Trabalha com desenvolvimento de software? Sim, no passado
Frequência de uso de POO: 4 - Frequentemente
Utiliza padrões de projeto? Sim, ocasionalmente
Conhecimento sobre princípios SOLID: Não, nunca ouvi falar
Metodologia eficiente para ensino? 3 - Moderadamente eficiente
Explicações teóricas suficientes? 2 - Pouco suficientes
Complexidade da primeira atividade: 2 - Moderadas
Projeto prático ajudou no entendimento? 4 - Ajudaram bastante
Princípios SOLID ajudaram na criação de código? 3 - Moderadamente
Confiança em aplicar padrões de projeto: 4 - Confiante
Aspectos que mais contribuíram: A criação do diagrama de classes, e também a confecção do código nos padrões como singleton que facilitam a implementação de novos dados e modificações no código.
Sugestões para melhorias: Fazer projetos menores mais simples com frequência, para ajudar a gravar esses padrões.
Observações gerais: Foi uma ótima experiência, dentro do âmbito de programação, pois pouco existe dessas práticas no decorrer da vida acadêmica de muitos alunos, é importante para não desvincularmos os estudos do ramo profissional.

Data/Hora: 21/02/2025 13:01:37
Trabalha com desenvolvimento de software? Não, nunca
Frequência de uso de POO: 4 - Frequentemente
Utiliza padrões de projeto? Não, nunca
Conhecimento sobre princípios SOLID: Não, nunca ouvi falar

Metodologia eficiente para ensino? 4 - Eficiente
Explicações teóricas suficientes? 2 - Pouco suficientes
Complexidade da primeira atividade: 2 - Moderadas
Projeto prático ajudou no entendimento? 4 - Ajudaram bastante
Princípios SOLID ajudaram na criação de código? 4 - Bastante
Confiança em aplicar padrões de projeto: 3 - Moderadamente confiante
Aspectos que mais contribuíram: Ter que colocar os conhecimentos adquiridos em prática.
Sugestões para melhorias: Mais tempo
Observações gerais: —

Data/Hora: 21/02/2025 13:12:45
Trabalha com desenvolvimento de software? Não, nunca
Frequência de uso de POO: 2 - Raramente
Utiliza padrões de projeto? Não, nunca
Conhecimento sobre princípios SOLID: Não, nunca ouvi falar
Metodologia eficiente para ensino? 3 - Moderadamente eficiente
Explicações teóricas suficientes? 3 - Moderadamente suficientes
Complexidade da primeira atividade: 2 - Moderadas
Projeto prático ajudou no entendimento? 3 - Ajudaram moderadamente
Princípios SOLID ajudaram na criação de código? 4 - Bastante
Confiança em aplicar padrões de projeto: 3 - Moderadamente confiante
Aspectos que mais contribuíram: A parte da prática e explicação de como usar as classes abstratas.
Sugestões para melhorias: Mais exemplos, códigos de antes vs depois
Observações gerais: —

Data/Hora: 21/02/2025 14:11:21
Trabalha com desenvolvimento de software? Não, nunca
Frequência de uso de POO: 2 - Raramente
Utiliza padrões de projeto? Não, nunca
Conhecimento sobre princípios SOLID: Não, nunca ouvi falar
Metodologia eficiente para ensino? 4 - Eficiente
Explicações teóricas suficientes? 4 - Suficientes
Complexidade da primeira atividade: 2 - Moderadas
Projeto prático ajudou no entendimento? 4 - Ajudaram bastante
Princípios SOLID ajudaram na criação de código? 3 - Moderadamente
Confiança em aplicar padrões de projeto: 3 - Moderadamente confiante
Aspectos que mais contribuíram: colocar em pratica oque eu vi na teoria
Sugestões para melhorias: fazer alguns mini projetos

Observações gerais: —

Data/Hora: 21/02/2025 14:40:43
Trabalha com desenvolvimento de software? Não, nunca
Frequência de uso de POO: 2 - Raramente
Utiliza padrões de projeto? Não, nunca
Conhecimento sobre princípios SOLID: Não, nunca ouvi falar
Metodologia eficiente para ensino? 4 - Eficiente
Explicações teóricas suficientes? 3 - Moderadamente suficientes
Complexidade da primeira atividade: 2 - Moderadas
Projeto prático ajudou no entendimento? 4 - Ajudaram bastante
Princípios SOLID ajudaram na criação de código? 4 - Bastante
Confiança em aplicar padrões de projeto: 3 - Moderadamente confiante
Aspectos que mais contribuíram: O fato de a gente ter estudado um assunto, ter que implementar e explicar ele na prática, por conta própria, foi muito interessante pra mim, porque só estudar um conteúdo e responder uma prova normalmente não me prende tanto, a forma que a gente estuda também é diferente, quando temos que aplicar na prática eu sinto que é muito mais fácil entender o conteúdo. Gostei muito to projeto, desde o princípio quando foi formulado pra gente, eu já achei muito interessante.
Sugestões para melhorias: Tenho algumas ressalvas que eu acho que melhorariam o projeto, caso seja aplicado em turmas futuras. - Deixar mais claro que a gente poderia mudar o código original: Eu fiquei com medo de implementar outras funcionalidades no meu código, como por exemplo, a funcionalidade de ExcluirUsuário, ou DevolverReserva. Eu acho que melhorar o código que foi formulado seria uma boa prática pra gente, fiquei com receio de fazer e acabar errando algo no meio do caminho, ou errar algo na mudança do Banco de Dados e acabar diminuindo minha nota tentando fazer algo mais excepcional do que o necessário. Por isso acabei não saindo muito do básico. - Talvez criar uma competitividade entre os alunos também seria interessante nesse caso, por exemplo: o grupo que tiver conseguido fazer o melhor código, contando as aplicações dos padrões de projeto, melhorias feitas no código, e até mesmo fazer interface, que um grupo fez, ganhar um ponto extra de bônus. Acho que isso também faria com que os alunos se esforçassem ainda mais no projeto.
Observações gerais: Como eu já falei, foi uma ótima experiência pra mim ter participado desse projeto. Espero que seja implementados para turmas futuras terem essa mesma experiência.

Data/Hora: 21/02/2025 14:40:50
Trabalha com desenvolvimento de software? Não, nunca
Frequência de uso de POO: 3 - Às vezes
Utiliza padrões de projeto? Sim, ocasionalmente

Conhecimento sobre princípios SOLID: Sim, mas não sabia em detalhes
Metodologia eficiente para ensino? 4 - Eficiente
Explicações teóricas suficientes? 4 - Suficientes
Complexidade da primeira atividade: 3 - Fáceis
Projeto prático ajudou no entendimento? 4 - Ajudaram bastante
Princípios SOLID ajudaram na criação de código? 4 - Bastante
Confiança em aplicar padrões de projeto: 3 - Moderadamente confiante
Aspectos que mais contribuíram: Foram claras e objetivas.
Sugestões para melhorias: Nao tenho
Observações gerais: —

Data/Hora: 21/02/2025 14:56:34
Trabalha com desenvolvimento de software? Sim, atualmente
Frequência de uso de POO: 4 - Frequentemente
Utiliza padrões de projeto? Não, nunca
Conhecimento sobre princípios SOLID: Sim, mas não sabia em detalhes
Metodologia eficiente para ensino? 4 - Eficiente
Explicações teóricas suficientes? 4 - Suficientes
Complexidade da primeira atividade: 2 - Moderadas
Projeto prático ajudou no entendimento? 4 - Ajudaram bastante
Princípios SOLID ajudaram na criação de código? 4 - Bastante
Confiança em aplicar padrões de projeto: 3 - Moderadamente confiante
Aspectos que mais contribuíram: Dividir o código em módulos e classes, cada um com sua responsabilidade. Definir as interfaces para desprender as classes de outras classes concretas, a implementação das estratégias para o problema proposto, entre outros.
Sugestões para melhorias: Uso de mais exemplos, ou mais aulas sobre o assunto, visto que um código bem estruturado e organizado pode ser chave para um bom desenvolvedor.
Observações gerais: —

Data/Hora: 21/02/2025 15:11:57
Trabalha com desenvolvimento de software? Não, nunca
Frequência de uso de POO: 3 - Às vezes
Utiliza padrões de projeto? Não, nunca
Conhecimento sobre princípios SOLID: Sim, mas não sabia em detalhes
Metodologia eficiente para ensino? 4 - Eficiente
Explicações teóricas suficientes? 3 - Moderadamente suficientes
Complexidade da primeira atividade: 2 - Moderadas
Projeto prático ajudou no entendimento? 4 - Ajudaram bastante
Princípios SOLID ajudaram na criação de código? 4 - Bastante
Confiança em aplicar padrões de projeto: 3 - Moderadamente confiante

Aspectos que mais contribuíram: Colaboração em equipe e apresentação do projeto, criando a necessidade de saber explicar sobre os padrões de projetos utilizados.
Sugestões para melhorias: Faz com que o aluno crie do zero o próprio projeto, implementando os padrões de projeto.
Observações gerais: —

Data/Hora: 21/02/2025 15:36:30
Trabalha com desenvolvimento de software? Sim, atualmente
Frequência de uso de POO: 2 - Raramente
Utiliza padrões de projeto? Não, nunca
Conhecimento sobre princípios SOLID: Sim, mas não sabia em detalhes
Metodologia eficiente para ensino? 4 - Eficiente
Explicações teóricas suficientes? 4 - Suficientes
Complexidade da primeira atividade: 2 - Moderadas
Projeto prático ajudou no entendimento? 4 - Ajudaram bastante
Princípios SOLID ajudaram na criação de código? 4 - Bastante
Confiança em aplicar padrões de projeto: 4 - Confiante
Aspectos que mais contribuíram: a interaçao com os colegas em sala
Sugestões para melhorias: talvez mais exemplos
Observações gerais: excelente didatica do professor e do joao

Data/Hora: 21/02/2025 15:50:36
Trabalha com desenvolvimento de software? Não, nunca
Frequência de uso de POO: 3 - Às vezes
Utiliza padrões de projeto? Não, nunca
Conhecimento sobre princípios SOLID: Não, nunca ouvi falar
Metodologia eficiente para ensino? 4 - Eficiente
Explicações teóricas suficientes? 3 - Moderadamente suficientes
Complexidade da primeira atividade: 3 - Fáceis
Projeto prático ajudou no entendimento? 3 - Ajudaram moderadamente
Princípios SOLID ajudaram na criação de código? 3 - Moderadamente
Confiança em aplicar padrões de projeto: 3 - Moderadamente confiante
Aspectos que mais contribuíram: ter a oportunidade de criar um código aplicando na prática as regras
Sugestões para melhorias: tentar dar mais alusões nas explicações pra facilitar o entendimento
Observações gerais: —

Data/Hora: 21/02/2025 18:36:23
Trabalha com desenvolvimento de software? Não, nunca

Frequência de uso de POO: 4 - Frequentemente
Utiliza padrões de projeto? Sim, ocasionalmente
Conhecimento sobre princípios SOLID: Sim, mas não sabia em detalhes
Metodologia eficiente para ensino? 4 - Eficiente
Explicações teóricas suficientes? 4 - Suficientes
Complexidade da primeira atividade: 2 - Moderadas
Projeto prático ajudou no entendimento? 4 - Ajudaram bastante
Princípios SOLID ajudaram na criação de código? 4 - Bastante
Confiança em aplicar padrões de projeto: 3 - Moderadamente confiante
Aspectos que mais contribuíram: Desenvolver e separar os códigos, de uma maneira que ficassem mais simples de serem entendidos. Comparar as diferentes tipos de abordagens que eu consegui desenvolver ao longo do projeto, o que fez com que eu percebesse o por que era tão necessário aprender sobre os padrões , e de que forma eu posso estruturar meus códigos de agora em diante. Além disso , as explicações que tivemos na etapa parcial ajudaram bastante, uma vez que tive a oportunidade de testar a estrutura de código e vez os pontos no qual eu poderia melhorar.
Sugestões para melhorias: A aplicações de mais projetos práticos, onde poderia ser aplicados ainda mais exemplos , com casos de uso reais no mercado, assim aplicando frameworks conhecidos, para conseguir entender o lado dos padrões de uma forma profissional, e como aplicamos nos códigos e em seus respectivos ambientes.
Observações gerais: Gostei muito do projeto proposto, ajudou bastante a ampliar meus conhecimentos, principalmente em elaboração e estruturação dos códigos.

Data/Hora: 21/02/2025 22:21:45
Trabalha com desenvolvimento de software? Não, nunca
Frequência de uso de POO: 1 - Nunca
Utiliza padrões de projeto? Não, nunca
Conhecimento sobre princípios SOLID: Sim, mas não sabia em detalhes
Metodologia eficiente para ensino? 3 - Moderadamente eficiente
Explicações teóricas suficientes? 3 - Moderadamente suficientes
Complexidade da primeira atividade: 3 - Fáceis
Projeto prático ajudou no entendimento? 3 - Ajudaram moderadamente
Princípios SOLID ajudaram na criação de código? 3 - Moderadamente
Confiança em aplicar padrões de projeto: 3 - Moderadamente confiante
Aspectos que mais contribuíram: Projetos aplicados diretamente em atividades.
Sugestões para melhorias: Envolver mais projetos em grupos.
Observações gerais: —

Data/Hora: 22/02/2025 17:30:38
Trabalha com desenvolvimento de software? Sim, atualmente

Frequência de uso de POO: 4 - Frequentemente
Utiliza padrões de projeto? Sim, ocasionalmente
Conhecimento sobre princípios SOLID: Sim, mas não sabia em detalhes
Metodologia eficiente para ensino? 3 - Moderadamente eficiente
Explicações teóricas suficientes? 3 - Moderadamente suficientes
Complexidade da primeira atividade: 1 - Difíceis
Projeto prático ajudou no entendimento? 3 - Ajudaram moderadamente
Princípios SOLID ajudaram na criação de código? 4 - Bastante
Confiança em aplicar padrões de projeto: 2 - Pouco confiante
Aspectos que mais contribuíram: A parte prática associada ao estudo da parte teórica repassada anteriormente
Sugestões para melhorias: Acho que vale a pena aprofundar mais em cada padrão de projeto e nos princípios de clean code, sendo pelo menos 2 por aula. Pois alguns deles podem ser difíceis de aplicar na prática. Seria útil tratar da parte teórica no começo da aula e depois fazer uma prática com tira dúvidas junto aos alunos. Mesmo que o progresso fique mais lento, vai ser muito melhor para solidificar o esses princípios que são muito importantes nos desenvolvimento de software.
Observações gerais: Achei muito interessante a aplicação de um projeto prático sobre padrões de projeto e princípios de clean code como o SOLID. Apenas endosso a necessidade de ir mais devagar e ter mais práticas e tira dúvidas em sala de aula. No geral foi uma ótima experiência com muito potencial aproveitado.

Data/Hora: 23/02/2025 14:32:02
Trabalha com desenvolvimento de software? Não, nunca
Frequência de uso de POO: 3 - Às vezes
Utiliza padrões de projeto? Sim, ocasionalmente
Conhecimento sobre princípios SOLID: Sim, mas não sabia em detalhes
Metodologia eficiente para ensino? 3 - Moderadamente eficiente
Explicações teóricas suficientes? 3 - Moderadamente suficientes
Complexidade da primeira atividade: 2 - Moderadas
Projeto prático ajudou no entendimento? 4 - Ajudaram bastante
Princípios SOLID ajudaram na criação de código? 4 - Bastante
Confiança em aplicar padrões de projeto: 3 - Moderadamente confiante
Aspectos que mais contribuíram: O desenvolvimento do projeto em si foi o maior contribuinte, testar e aplicar os padrões de projeto no sistema, geraram maior absorção de conhecimento, que apenas uma aula expositiva sobre o tema.
Sugestões para melhorias: Uma explicação mais detalhada do padrões de projeto mais complexos, o entendimento e aplicação dos padrões S e I, foram bem simples, entretanto, houve maior dificuldade em compreender os outros três padrões (O,L,D).

Observações gerais: Em conclusão, o projeto foi de grande ajuda, aplicar os padrões, podem ajudar na manutenção de projetos com maior complexidade a partir de agora, sendo de grande ajuda para o desenvolvimento profissional.

Data/Hora: 23/02/2025 14:37:02
Trabalha com desenvolvimento de software? Não, nunca
Frequência de uso de POO: 3 - Às vezes
Utiliza padrões de projeto? Não, nunca
Conhecimento sobre princípios SOLID: Não, nunca ouvi falar
Metodologia eficiente para ensino? 4 - Eficiente
Explicações teóricas suficientes? 3 - Moderadamente suficientes
Complexidade da primeira atividade: 2 - Moderadas
Projeto prático ajudou no entendimento? 4 - Ajudaram bastante
Princípios SOLID ajudaram na criação de código? 4 - Bastante
Confiança em aplicar padrões de projeto: 3 - Moderadamente confiante
Aspectos que mais contribuíram: Acho que de fato colocar a mão no projeto foi a aparte mais importante, sair da teoria é muito importante, ainda mais com o nível de abstração abstração dos assuntos !
Sugestões para melhorias: talvez a colocação de mais alguns checkpoints para o acompanhamento do projeto!
Observações gerais: —

Data/Hora: 24/02/2025 00:07:08
Trabalha com desenvolvimento de software? Não, nunca
Frequência de uso de POO: 2 - Raramente
Utiliza padrões de projeto? Sim, ocasionalmente
Conhecimento sobre princípios SOLID: Não, nunca ouvi falar
Metodologia eficiente para ensino? 4 - Eficiente
Explicações teóricas suficientes? 4 - Suficientes
Complexidade da primeira atividade: 2 - Moderadas
Projeto prático ajudou no entendimento? 4 - Ajudaram bastante
Princípios SOLID ajudaram na criação de código? 3 - Moderadamente
Confiança em aplicar padrões de projeto: 4 - Confiante
Aspectos que mais contribuíram: Aplicar os padrões de projeto na prática, principalmente na hora de fazer o código e ver onde cada padrão é ou pode ser aplicado.
Sugestões para melhorias: O professor Davi e João Batista foram muito empenhados no ensino dos padrões de projeto e na hora de prestar apoio e dar dicas aos grupos, gostei bastante.
Observações gerais: Muito bom o projeto, espero aprender e aplicar ainda mais isso no futuro.

Data/Hora: 24/02/2025 15:00:15
Trabalha com desenvolvimento de software? Não, nunca
Frequência de uso de POO: 3 - Às vezes
Utiliza padrões de projeto? Sim, ocasionalmente
Conhecimento sobre princípios SOLID: Não, nunca ouvi falar
Metodologia eficiente para ensino? 4 - Eficiente
Explicações teóricas suficientes? 4 - Suficientes
Complexidade da primeira atividade: 3 - Fáceis
Projeto prático ajudou no entendimento? 4 - Ajudaram bastante
Princípios SOLID ajudaram na criação de código? 4 - Bastante
Confiança em aplicar padrões de projeto: 3 - Moderadamente confiante
Aspectos que mais contribuíram: A colocar em prática a teoria.
Sugestões para melhorias: Nenhuma sugestão.
Observações gerais: —

Data/Hora: 24/02/2025 18:41:51
Trabalha com desenvolvimento de software? Não, nunca
Frequência de uso de POO: 3 - Às vezes
Utiliza padrões de projeto? Não, nunca
Conhecimento sobre princípios SOLID: Não, nunca ouvi falar
Metodologia eficiente para ensino? 4 - Eficiente
Explicações teóricas suficientes? 3 - Moderadamente suficientes
Complexidade da primeira atividade: 2 - Moderadas
Projeto prático ajudou no entendimento? 4 - Ajudaram bastante
Princípios SOLID ajudaram na criação de código? 4 - Bastante
Confiança em aplicar padrões de projeto: 3 - Moderadamente confiante
Aspectos que mais contribuíram: A parte de implementar um padrão de projeto em código já pronto, onde tínhamos que remodelar ele para se adequar aos padrões foi muito importante para nós aprender sobre eles.
Sugestões para melhorias: De acordo com o que foi ministrado em aula e no projeto deixou o assunto bem compreensível.
Observações gerais: —