USANDO CHATGPT PARA APOIO AO EMPREGO DE PADRÕES DE PROJETO EM UMA APLICAÇÃO DE CAD PARA CIRCUITOS LÓGICOS

VÍTOR MONTEIRO COLOMBO[1]; WESLEN SCHIAVON DE SOUZA[1]; BRENDA SALENAVE SANTANA[1]; TATIANA AIRES TAVARES[1]; LISANE BRISOLARA DE BRISOLARA[1]

[1]UFPEL - {vmcolombo, wsdsouza, bssalenave, tatiana, lisane}@inf.ufpel.edu.br

1. INTRODUÇÃO

Grandes modelos de linguagem (LLMs, do inglês, *Large Language Models*) são exemplos de Inteligência Artificial Generativa, que se destacam na geração de texto, de conteúdo e no processamento de linguagem natural, baseados em aprendizado de máquina (NAVEED et al., 2025). Evoluções recentes nesses modelos têm motivado sua aplicação em diversas tarefas.

Conforme apontado por ZHANG et al. (2024), na área de Engenharia de Software, estes modelos têm sido usados para geração e refatoração de código, geração de documentação e teste de software. Entretanto, etapas iniciais do processo de desenvolvimento de software, como a engenharia de requisitos e o projeto de software, também podem se beneficiar do uso de tais abordagens.

Na etapa de projeto, a adoção de padrões de projeto de software exige conhecimento especializado, e sua escolha e aplicação influenciam diretamente à qualidade e a estrutura do sistema. Esses padrões oferecem soluções reutilizáveis para problemas recorrentes, especialmente no paradigma orientado a objetos, destacando-se os padrões GoF (*Gang of Four*) (GAMMA, 1995). Estudos recentes reportados por PAN et al. (2025) e PANDEY et al. (2025) já demonstraram a capacidade de LLMs em identificar padrões de projeto diretamente no código-fonte. No entanto, não foram encontrados estudos que estudem a capacidade destes modelos de sugerirem padrões de projeto em etapas iniciais do desenvolvimento, quando código ainda não foi escrito.

Diante do exposto, este trabalho tem dois objetivos: (i) avaliar empiricamente o ChatGPT na recomendação de padrões de projeto a partir de descrições textuais do software; e (ii) disponibilizar o conjunto de dados gerado — incluindo prompt, descrições e respostas — para apoiar pesquisas futuras.

2. METODOLOGIA

Esta seção apresenta a metodologia empregada para o experimento. O primeiro passo foi definir o prompt a ser utilizado, seguido da elaboração da descrição do projeto. Na sequência, essas informações (prompt e descrição do software), foram submetidos ao GPT-4 Turbo. O resultado foi coletado e avaliado por três especialistas. A Figura 1 ilustra e sintetiza a metodologia empregada no estudo de caso.

Com base em alguns experimentos iniciais, foi otimizado o prompt a ser empregado. Na definição do prompt, também considerou-se orientações apresentadas em (WHITE et al., 2024) para exploração de padrões arquiteturais. O prompt empregado está ilustrado na Figura 2 e delimita o formato e escopo das respostas, de forma a orientar a ferramenta a produzir saídas mais alinhadas com o experimento.

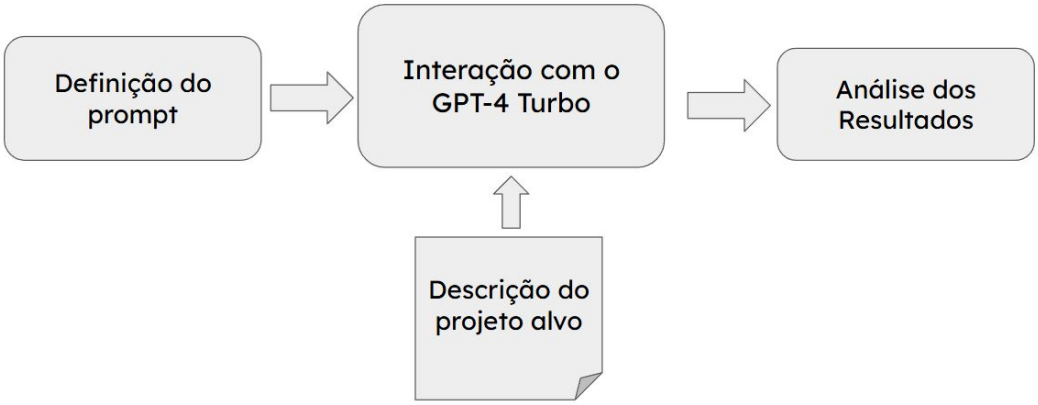

Figura 1: Metodologia utilizada no experimento

Este prompt limitou o modelo a sugerir três padrões somente, pois a análise de um número elevado de padrões seria inviável no contexto de um projeto real.

Considere que sou um projetista de software atuando no projeto **"título do projeto"**

Gostaria de auxílio para identificar e explorar o emprego de padrões de projeto orientado a objeto (design patterns)

Gostaria de sugestões de padrões até três padrões de projeto que melhor atendam às características, funcionalidades e objetivos mencionados na descrição do software.

Para cada padrão sugerido, apresente
Padrão: [nome]
Categoria: [Criacional / Estrutural / Comportamental]
Aplicação no projeto: [...]
Benefícios: [...]

Considerando que padrões também podem trazer complexidade ao código, indique apenas os padrões mais benéficos no contexto.

"Descrição textual detalhada:"

Figura 2: Prompt utilizado no experimento

O experimento considerou o desenvolvimento de um software desktop voltado ao projeto de Circuitos Lógicos, oferecendo funcionalidades de edição e simulação. Esse tipo de aplicação se enquadra na categoria de ferramentas de Computer-Aided Design (CAD), ou projeto assistido por computador, comumente utilizadas para facilitar a criação, modificação e visualização de sistemas técnicos. O objetivo no uso de padrões de projeto nesta aplicação é aprimorar a manutenibilidade da aplicação, facilitando futuras modificações e garantindo maior escalabilidade.

3. RESULTADOS EXPERIMENTAIS

A Tabela 1 apresenta um resumo das sugestões fornecidas pelo GPT-4 Turbo, especificando, para cada padrão indicado, sua categoria, uma síntese de sua aplicação no contexto proposto e os benefícios esperados. O texto empregado como entrada, o texto usado como prompt na interação, bem como o retorno gerado pelo mo-

delo estão disponibilizados no link https://drive.google.com/drive/folders/106LSo3vJELlsAoPWOQBVIzsxjw5bvLq?usp=drive_link.

Tabela 1: Padrões de Projeto, Categoria, Aplicação e Benefícios

Padrão Sugerido	**Categoria**	**Aplicação**	**Benefícios**
Composite	Estrutural	Modelagem dos circuitos hierárquicos	Manipulação uniforme, fácil expansão, simplifica recursão para tratar geração de código, simulação, etc.
Command	Comportamental	Gerenciamento das ações do usuário na interface gráfica	Facilita a implementação robusta de undo/redo, aumentando muito a usabilidade.
Factory Method	Criacional	Centraliza a criação dos componentes do circuito	Organiza a criação dos objetos, melhorando a coesão e facilitando a manutenção.

No intuito de avaliar os resultados obtidos, utilizou-se como referência o método de inspeção de software, tradicionalmente empregado para analisar a conformidade de artefatos de software. De acordo com KALINOWSKI (2008), a inspeção clássica segue etapas como planejamento, revisão individual, reunião em equipe, registro de defeitos, correção pelo autor e avaliação final. No experimento, esse método foi adaptado para a análise de sugestões geradas por um LLM.

Na etapa de validação, foram convidados dois especialistas, do gênero feminino, com 20 e 10 anos de experiência em projeto de software orientado a objetos para avaliar as recomendações. Os especialistas foram responsáveis por inspecionar as sugestões fornecidas pela ferramenta, avaliando sua adequação ao contexto e aos seus requisitos. Em casos de divergência, uma terceira especialista, com experiência equivalente, atuou como avaliadora de desempate.

O padrão *Composite* foi considerado totalmente aderente, considerando a descrição textual do projeto, uma vez que sua aplicação facilita a construção de hierarquias estruturais de componentes, característica essencial em sistemas que manipulam circuitos compostos por subsistemas.

O padrão *Command* foi considerado parcialmente aderente ao contexto do experimento. Tal padrão poderia auxiliar no gerenciamento das ações do usuário na interface gráfica, facilitando a implementação de funcionalidades como desfazer e refazer. Embora estas funcionalidades sejam interessantes em softwares de edição gráfica, elas não haviam sido mencionadas explicitamente na descrição do software, e por isso os especialistas avaliaram como parcial.

O padrão *Factory Method* não foi considerado necessário, já que a instância dos componentes a partir da barra de ferramentas não requer encapsulamento adicional da lógica de criação, ou seja, o componente a ser instanciado já é definido a

priori quando o usuário o escolhe a partir de uma biblioteca.

4. CONCLUSÕES

Este artigo avaliou a habilidade da ferramenta GPT-4 Turbo de sugerir o emprego de padrões de projeto, apenas baseado em uma descrição textual do software a ser desenvolvido. A descrição empregada no experimento é um artefato construído para o Ensino de Engenharia de Software e já havia sido aplicado em trabalhos práticos com foco em Análise e Projeto de Software orientado a objetos. O experimento reportado demonstrou que a ferramenta pode ser útil a projetistas, sobretudo aos menos experientes, sugerindo e enriquecendo a discussão sobre a aplicabilidade de padrões de projeto de software orientado a objetos e auxiliando na tomada de decisão.

A discussão apresentada neste trabalho limita-se a um único experimento. No futuro, pretendemos estender os experimentos para outras aplicações, bem como avaliar diferentes ferramentas baseadas em LLMs. Devido aos resultados não determinísticos, é necessária a realização de experimentos com maior número de interações e análise estatística dos resultados para conclusões mais sólidas. O grupo de pesquisa já está trabalhando em experimentos adicionais de forma a tratar estas limitações mencionadas.

5. REFERÊNCIAS BIBLIOGRÁFICAS

GAMMA, E. **Design patterns: elements of reusable object-oriented software**. Reading: Addison-Wesley, 1995.

KALINOWSKI, M. **Introdução à inspeção de software**. **Revista Engenharia de Software: Qualidade de software** 1, vol. 1, n., p. pp. 68–74, 2008.

NAVEED, H.; KHAN, A. U.; QIU, S.; SAQIB, M.; ANWAR, S.; USMAN, M.; AKHTAR, N.; BARNES, N.; MIAN, A. **A Comprehensive Overview of Large Language Models**. **ACM Trans. Intell. Syst. Technol.** 16.5, New York, vol. 16, n. 5, p. pp. 1–72, 2025.

PAN, Z.; SONG, X.; WANG, Y.; CAO, R.; LI, B.; LI, Y.; LIU, H. (2025). **Do Code LLMs Understand Design Patterns?** arXiv preprint arXiv:2312.15223.

PANDEY, S. K.; CHAND, S.; HORKOFF, J.; STARON, M.; OCHODEK, M.; DURISIC, D. **Design pattern recognition: a study of large language models**. **Empirical Software Engineering** 30, vol. 30, n., p. p. 69, 2025.

WHITE, J.; HAYS, S.; FU, Q.; SPENCER-SMITH, J.; SCHMIDT, D. C. **ChatGPT Prompt Patterns for Improving Code Quality, Refactoring, Requirements Elicitation, and Software Design**. Cham: Springer Nature Switzerland, 2024.

ZHANG, Q.; FANG, C.; XIE, Y.; ZHANG, Y.; YANG, Y.; SUN, W.; YU, S.; CHEN, Z. (2024). **A Survey on Large Language Models for Software Engineering**. arXiv preprint 2312.15223.