UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS QUIXADÁ

CURSO DE GRADUAÇÃO EM ENGENHARIA DE SOFTWARE

GABRIEL DE SOUZA MENDES

INVESTIGANDO ESTRATÉGIAS DE ENGENHARIA DE *PROMPT* PARA MELHORAR A QUALIDADE DE REQUISITOS GERADOS POR LLMS

QUIXADÁ

2026

GABRIEL DE SOUZA MENDES

INVESTIGANDO ESTRATÉGIAS DE ENGENHARIA DE *PROMPT* PARA MELHORAR A QUALIDADE DE REQUISITOS GERADOS POR LLMS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Software do Campus Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia de Software.

Orientadora: Profa. Dra. Carla Ilane Moreira Bezerra.

QUIXADÁ

2026

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

M491i Mendes, Gabriel de Souza.
Investigando Estratégias de Engenharia de prompt para Melhorar a Qualidade de Requisitos Gerados por LLMs / Gabriel de Souza Mendes. – 2026.
63 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Engenharia de Software, Quixadá, 2026.
Orientação: Profa. Dra. Carla Ilane Moreira Bezerra.

1. Engenharia de requisitos. 2. Large language models. 3. Engenharia de Prompt. 4. Qualidade de software. I. Título.

CDD 005.1

GABRIEL DE SOUZA MENDES

INVESTIGANDO ESTRATÉGIAS DE ENGENHARIA DE *PROMPT* PARA MELHORAR A QUALIDADE DE REQUISITOS GERADOS POR LLMS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Software do Campus Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia de Software.

Aprovada em: 22/01/2026.

BANCA EXAMINADORA

Profa. Dra. Carla Ilane Moreira Bezerra (Orientadora)
Universidade Federal do Ceará (UFC)

Profa. Dra. Carla Taciana Lima Lourenço Silva Schuenemann
Universidade Federal de Pernambuco (UFPE)

Profa. Dra. Rainara Maia Carvalho
Universidade Federal do Ceará (UFC)

À minha mãe, por todo apoio. Ao meu pai por sempre ter acreditaco em mim. À minha madrinha, por todo acolhimento e suporte.

AGRADECIMENTOS

Primeiramente agradeço a Deus, por sempre dar força para seguir nessa caminhada, sem o dom dado por Ele jamais chegaria onde estou. Ter passado por todos os percalços ao longo da gradução não foi fácil, mas tenho certeza que todos os perrengues foram necessários para meu crescimento enquanto aluno.

Aos meus pais por todo apoio financeiro e emocional, sobretudo, por sempre me motivarem ao estudo e me deixarem a vontade para decidir meu destino.

À minha madrinha, a pessoa que me acolheu ao longo desses 4 anos, e quem mais esteve presente em meu dia a dia ao longo da faculdade, principalmente me dando os melhores conselhos que poderia receber.

Aos meus amigos, por todas as conversas, trocas de ideias, risadas. De modo especial Felipe, Caio e Emanuel, por todos atividades e trabalhos em conjunto.

À minha orientadora Profa. Dra. Carla Ilane, por todos os ensinamentos, orientações que foram cruciais para o desenvolvimento desse trabalho.

À minha namorada, por ter estado presente e me ajudado e incentivado nessa reta final de escrita.

À Universidade Federal do Ceará e todo o corpo docente tão competente e dedicado ao ensino e aprendizado dos alunos.

RESUMO

A qualidade dos requisitos de software é um fator determinante para o sucesso de projetos, visto que falhas de especificação geram altos custos de retrabalho e inconsistências no produto final. Os *Large Language Models* (LLMs) têm sido explorados para auxiliar na elicitação e documentação de requisitos, embora ainda apresentem limitações quanto à precisão de regras de negócio e alucinações. Este estudo investiga empiricamente o impacto de diferentes estratégias de Engenharia de *Prompt* (*Input Only*, *Chain of Thought* e *Tree of Thought*), na qualidade de requisitos funcionais gerados pelos modelos GPT-5, Gemini 3 e DeepSeek-R1. A metodologia adotou uma abordagem híbrida, combinando uma análise quantitativa da completude estrutural (ator, ação, objeto e condição) com uma avaliação qualitativa baseada em clareza, completude, consistência e relevância, finalizando com uma validação prática junto a *stakeholders*. Os resultados indicam que, embora os modelos apresentem alto desempenho em atributos básicos, a especificação de Condições representa o maior desafio, registrando taxas de presença inferiores a 60% na estratégia *Input Only*. A combinação do modelo GPT-5 com a técnica *Chain of Thought* produziu os requisitos com a maior qualidade geral, atingindo médias superiores a 4,90 nos critérios citados na análise qualitativa. Entretanto, a validação com especialistas revelou que apenas 33% dos requisitos gerados foram plenamente aceitos sem restrições, sendo a Omissão de informações o defeito mais prevalente (50%). Além disso, observou-se uma forte correlação positiva entre a completude estrutural das condições e a percepção humana de qualidade. Conclui-se que o uso de estratégias avançadas de raciocínio é essencial para mitigar deficiências dos LLMs, servindo como uma base sólida para a engenharia de requisitos, mas não elimina a necessidade de revisão humana especializada.

Palavras-chave: Engenharia de requisitos; *Large language models*; Engenharia de *prompt*; Qualidade de software.

ABSTRACT

The quality of software requirements is a determining factor for the success of projects, since specification failures generate high rework costs and inconsistencies in the final product. Large Language Models (LLMs) have been explored to support requirements elicitation and documentation; however, they still present limitations regarding the accuracy of business rules and hallucinations. This study empirically investigates the impact of different Prompt Engineering strategies (*Input Only*, *Chain of Thought*, and *Tree of Thought*) on the quality of functional requirements generated by the GPT-5, Gemini 3, and DeepSeek-R1 models. The adopted methodology followed a hybrid approach, combining a quantitative analysis of structural completeness (actor, action, object, and condition) with a qualitative evaluation based on clarity, completeness, consistency, and relevance, concluding with practical validation involving stakeholders. The results indicate that, although the models demonstrate high performance in basic attributes, the specification of Conditions represents the greatest challenge, with presence rates below 60% under the *Input Only* strategy. The combination of the GPT-5 model with the *Chain of Thought* technique produced the highest overall quality requirements, achieving average scores above 4.90 in the criteria considered in the qualitative analysis. However, validation with experts revealed that only 33% of the generated requirements were fully accepted without restrictions, with Information Omission being the most prevalent defect (50%). Furthermore, a strong positive correlation was observed between the structural completeness of conditions and the human perception of quality. It is concluded that the use of advanced reasoning strategies is essential to mitigate LLM deficiencies, serving as a solid foundation for requirements engineering; however, it does not eliminate the need for specialized human review.

Keywords: Requirements engineering; Large language models; Prompt engineering; Software quality.

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

CoT	*Chain of Thought*
CSI	*Cardio Surgery Illustrar*
EP	Engenharia de *Prompt*
ER	Engenharia de Requisitos
ES	Engenharia de Software
IA	Inteligência Artificial
IO	*Input Only*
LLM	*Large Language Model*
LLMs	*Large Language Models*
PLN	Processamento de Linguagem Natural
RF	Requisitos Funcionais
RNF	Requisitos Não Funcionais
ToT	*Tree of Thought*

SUMÁRIO

1 INTRODUÇÃO

A qualidade dos requisitos tem impacto direto na confiabilidade e sucesso de um sistema de *software*. Estudos mostram que uma parcela significativa das falhas em sistemas complexos decorre de erros presentes na fase de requisitos, sendo responsáveis por altos custos de correção e retrabalho (Sommerville, 2011). Documentos de requisitos mal formulados podem introduzir ambiguidades, inconsistências e lacunas que afetam a compressão e clarezas dos mesmos, por parte dos stakeholders. Conforme destacado por Mahbub *et al.* (2024), quando esses defeitos não são identificados nas fases iniciais, resultam em custos elevados de retrabalho e desalinhamento com os objetivos do sistema. Esses problemas afetam diretamente a eficiência e a eficácia no desenvolvimento de *software* (Vogelsang, 2024).

Com o avanço recente dos *Large Language Models* (LLMs), como o GPT-4, muitos profissionais da engenharia de *software* têm usado e explorado todo o poder e capacidade desses modelos, sobretudo, tanto para auxiliar na geração, como também na validação e refinamento dos requisitos escritos em linguagem natural (Vogelsang, 2024). Apesar disso, os LLMs têm se mostrado limitadas para identificar de maneira precisa alguns defeitos, como inconsistências e ambiguidades, frequentemente gerando falsos positivos e tendo dificuldades na compreensão de contexto mais amplos dos documentos (Mahbub *et al.*, 2024; Masoudifard *et al.*, 2024)

Dessa forma, a Engenharia de *Prompt* (EP) surge como uma abordagem estratégica para melhorar e otimizar a interação com os LLMs, sendo capaz de direcionar o raciocínio dos modelos e melhorar a qualidade das respostas obtidas. Assim, a construção dos *prompts* influencia diretamente a clareza, completude e relevância dos requisitos gerados. Segundo (Chen *et al.*, 2025), estamos passando por uma "crise de *promptware*", onde os *prompts* são frequentemente desenvolvidos sem suporte técnico ou científico, gerando assim respostas frágeis e inconsistentes. (Vogelsang, 2024) deixa claro que é essencial explorar métodos sistemáticos e científicos no desenvolvimento de prompts para que estes sejam tratados como artefatos de *software*.

Sendo assim, algumas técnicas com CoT e ToT têm mostrado eficácia significativa no aprimoramento da capacidade de raciocínio dos LLMs, possibilitando assim a geração de requisitos mais robustos e confiáveis. Masoudifard *et al.* (2024) demonstraram que o uso combinado de CoT e ToT, juntamento com o mecanismo *Graph-RAG* para recuperação contextualizada de informações, melhora consideravelmente a precisão e a rastreabilidade, com desempenhos superiores a 85% em tarefas complexas de conformidade e verificação automatizada. Além

disso, algumas técnicas como *few-shot prompting* e *retrieval augmented generation* também têm apresentado bons resultados na melhoria da qualidade dos requisitos gerados (Vogelsang, 2024).

Outra abordagem para melhorar a qualidade dos requisitos gerados, foi proposta por Ma *et al.* (2025), sendo a mesma voltada para melhorar a interação entre humanos e modelos de linguagem, denominada *Requirement-Oriented Prompt Engineering* (ROPE), com o objetivo de preencher uma lacuna existente na literatura sobre a importância da articulação clara e completa de requisitos no prompts. Esse paradigma foca em treinar humanos para que os mesmos sejam capazes de formular de maneira mais eficaz, garantindo uma interação mais precisa com modelos de linguagem. Os autores destacam que a qualidade das respostas dos LLMs está diretamente relacionada à clareza das instruções fornecidas aos modelos, mostrando melhorias consideráveis no desempenho de usuários novatos que receberam o treinamento em ROPE em comparação com o treinamento tradicional em EP (Ma *et al.*, 2025).

Nesse contexto, se faz necessário avaliar a qualidade dos requisitos gerados por LLMs a partir de critérios objetivos e consistentes. Para isso, Masoudifard *et al.* (2024) utiliza métricas como precisão, *recall* e *F1-score* para mensurar o desempenho de técnicas como ToT e *Graph-RAG* na detecção de discordâncias nos requisitos. Essas métricas foram anotadas manualmente, permitindo analisar a presença de defeitos, além da gravidade e relevância dos mesmos, de acordo com os padrões normativos. Além disso, avaliacões qualitativas feitas por especialistas também foram levadas em consideração, reforçando assim a confiabilidade das análises, o que sugere uma abordagem híbrida para mensurar a qualidade em contextos gerais de Engenharia de Requisitos (ER).

Dessa forma, esta pesquisa busca investigar como diferentes estratégias de engenharia de *prompt* podem ser capazes de impactar na qualidade de requisitos gerados por LLMs. O propósito é avaliar experimentos empíricos, comparando abordagens de *prompts* simples, CoT e ToT, quanto à clareza, completude e alinhamento com objetivos e normas estabelecidas. Para alcançar esse objetivo, serão realizados experimentos empíricos com cada técnica de *prompt* sendo aplicada a contextos específicos de geração de requisitos. Após isso os resultados obtidos serão analisados, tanto através de métricas objetivas como também com avaliações quantitativas, para assim ter uma maior compreensão dos impactos práticos de cada abordagem testada. O foco desse estudo é fornecer evidências a respeito da vantagem e das limitações de estratégias de *prompting* aplicadas à ER, contribuindo assim para a sistematização de boas práticas no uso de LLMs nesse domínio.

1.1 Objetivos

1.1.1 Objetivo geral

Investigar empiricamente a influência de estratégias de EP (IO, CoT e ToT) na geração de requisitos por múltiplos LLMs, propondo um *framework* de avaliação híbrida que integra a análise da completude estrutural, a qualidade intrínseca e a validação prática dos artefatos.

1.1.2 Objetivos específicos

a) Avaliar a completude estrutural dos requisitos gerados, verificando a presença dos atributos: Ator, Ação, Objeto e Condição, por meio de uma análise quantitativa.
b) Comparar o desempenho das diferentes combinações de técnica (IO, CoT, ToT) e *Large Language Model* (LLM), com base na análise qualitativa sobre os critérios de Clareza, Completude, Consistência e Relevância.
c) Identificar, a partir da síntese das análises quantitativa e qualitativa, qual combinação de técnica de *prompt* e LLM produz os requisitos de *software* com a maior qualidade geral.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção, serão apresentados os conceitos essenciais para a compreensão deste trabalho. Na Seção 2.1, serão abordados os fundamentos da ER, com foco nas definições e práticas fundamentais para garantir a qualidade e a precisão na documentação de requisitos de software. Na Seção 2.2, serão explorados os LLMs, com destaque para ChatGPT, Deepseek e Gemini. Por fim, na Seção 2.3, serão discutidos os conceitos relacionados à EP, destacando as técnicas de EP, e sua aplicação para otimizar a geração de requisitos com LLMs.

2.1 Engenharia de requisitos

A ER é tanto a parte responsável pelo processo de especificação de requisitos por meio do estudo das necessidades das partes interessadas como também pelo processo de analisar e refinar sistematicamente essas especificações (Hofmann; Lehner, 2001). Ela possui um papel fundamental em garantir que as necessidades do usuário sejam compreendidas, documentadas e comunicadas, assegurando a construção de sistemas eficazes e eficientes (Vasudevan; Reddivari, 2025). Além disso, Essa etapa é crucial para a evolução e construção de um software, pois é partir do requisitos elicitados que se dá o desenvolvimento do software, com o objetivo de garantir que seus objetivos desejados pelas partes interessadas sejam refletidos no produto final (Vasudevan; Reddivari, 2025).

O processo de ER é comumente dividido em quatro etapas principais: elicitação, modelagem, validação e verificação (Umar *et al.*, 2025). A elicitação não se limita apenas a entrevistas e análise documental, mas também envolve métodos avançados como sessões de grupo, técnicas específicas para sistemas baseados em conhecimento e a utilização de ferramentas automatizadas para apoiar o levantamento dos requisitos (Umar *et al.*, 2025; Ferrari; Spoletini, 2025). No entanto, um desafio comum durante a elicitação é a falta de clareza nos requisitos. Por exemplo, um requisito como “O sistema deve ser rápido” é vago e não especifica o que significa “rápido”. Uma versão mais precisa seria “O sistema deve processar até 100 transações por segundo”, o que proporciona uma métrica clara e mensurável.

Já modelagem envolve o uso de diferentes notações, variando entre informais, semi-formais e formais, para representar as soluções percebidas no contexto do domínio de aplicação. Modelos são refinados progressivamente até chegar a uma especificação satisfatória. Com o uso de LLMs, esse processo pode ser acelerado, permitindo a geração de modelos de requisitos

iniciais que podem ser facilmente refinados, aumentando a eficiência e reduzindo o tempo gasto em iterações (Ferrari; Spoletini, 2025). Após a modelagem, os requisitos gerados passam pelos processos de validação e verificação. Um exemplo de modelagem incompleta seria a descrição de um sistema como "O sistema deve permitir a entrada de dados". Essa descrição se mostra incompleta, e para melhora-la podemos especificar melhor as condições de entrada, como por exemplo: "O sistema deve permitir a entrada de dados de até 10.000 registros simultâneos", garantindo maior clareza quanto ao requisito.

A validação verifica se os requisitos capturam corretamente as intenções e necessidades das partes interessadas, enquanto a verificação assegura que a especificação seja consistente e correta internamente. Para execução desses processos, podemos usar técnicas como revisões por pares, inspeções técnicas e testes de cenários, essenciais para minimizar retrabalho e custos adicionais no desenvolvimento (Umar *et al.*, 2025; Hofmann; Lehner, 2001). Além disso, a priorização contínua dos requisitos ao longo do projeto permite garantir que os aspectos mais críticos sejam tratados primeiramente, ajudando na gestão eficaz de recursos e tempo (Ferrari; Spoletini, 2025).

Embora a elicitação de requisitos com stakeholders seja fundamental na ER, ela enfrenta alguns desafios. Conforme apontado por Daun *et al.* (2023), a falta de envolvimento contínuo dos stakeholders e a dificuldade em comunicar necessidades complexas são questões comuns que limitam a eficácia dessa abordagem. Muitas vezes, os stakeholders têm dificuldades em articular de forma clara e precisa as necessidades do sistema, especialmente em projetos mais complexos, o que pode resultar em requisitos vagos ou incompletos. Além disso, o tempo disponível para a elicitação pode ser restrito, especialmente em ambientes de desenvolvimento ágil, onde as mudanças são frequentes e os requisitos podem evoluir rapidamente (Hofmann; Lehner, 2001). Esses desafios são ampliados pela falta de experiência prática dos stakeholders em traduzir suas necessidades para requisitos técnicos, o que pode gerar lacunas ou ambiguidades na documentação dos requisitos (Vasudevan; Reddivari, 2025). Embora técnicas como workshops e entrevistas possam mitigar essas limitações, elas ainda dependem fortemente da participação ativa e da habilidade de comunicação de todos os envolvidos. Dessa forma, ao aplicarmos os LLMs juntamente com a ER pode ser uma ótimo solução, automatizando assim parte do processo de elicitação e fornecendo um ponto de partida mais estruturado e claro para os requisitos.

Nos contextos de desenvolvimento ágil, a ER desempenha um papel crucial, devido à necessidade constante de adaptação e flexibilidade frente às mudanças contínuas. O uso de

técnicas como observações, entrevistas e *workshops*, associadas ao envolvimento frequente dos clientes, garante uma gestão iterativa eficaz, garantindo assim um alinhamento constante das entregas com as expectativas dos usuários finais (Umar *et al.*, 2025; Vasudevan; Reddivari, 2025).

Além disso, a ER não se restringe aos modelos tradicionais de desenvolvimento de software, também estando presente nas abordagens ágeis, onde desempenha um papel fundamental no refinamento contínuo e na adaptação dos requisitos aos objetivos em constante evolução do projeto (Umar *et al.*, 2025).

Portanto, o entendimento aprofundado das práticas de ER e a correta implementação das suas etapas são determinantes para garantir a qualidade e o sucesso contínuo dos projetos de software, além atender as necessidades do stakeholders (Hofmann; Lehner, 2001; Umar *et al.*, 2025).

2.1.1 Especificação e Estrutura de Requisitos

A qualidade da documentação de requisitos não depende apenas da elicitação correta das necessidades, mas também da rigidez sintática com que essas necessidades são redigidas. Embora a linguagem natural ofereça flexibilidade, seu uso sem restrições em processos automatizados frequentemente resulta em ambiguidades que comprometem a engenharia de software baseada em modelos (Umar *et al.*, 2025). Nesse contexto, a literatura recente reforça o uso de estruturas padronizadas como um pré-requisito para mitigar inconsistências e viabilizar o processamento eficaz por algoritmos de Inteligência Artificial (Rocha *et al.*, 2025).

Para garantir essa precisão, Umar *et al.* (2025) estabeleceram que a descrição de uma funcionalidade deve seguir uma decomposição semântica rigorosa. Essa estrutura fundamental contempla, minimamente, quatro elementos sintáticos: o ator (ou sujeito), a ação, o objeto e a condição (ou contexto).

O Ator refere-se à entidade ativa, humana ou sistêmica, responsável por iniciar a interação. Em arquiteturas modernas, a definição precisa do ator é crucial não apenas para o controle de acesso, mas para delimitar as fronteiras de responsabilidade entre microserviços e usuários finais (Umar *et al.*, 2025).

A Ação constitui o núcleo operacional do requisito, descrevendo o comportamento observável que o sistema deve executar. A literatura de Processamento de Linguagem Natural (PLN) aplicada a requisitos sugere que a ação seja expressa por verbos transitivos diretos, evitando termos vagos que possam gerar alucinações ou interpretações dúbias por modelos

generativos (Rocha *et al.*, 2025).

O Objeto indica a entidade passiva ou o artefato de dados sobre o qual a ação incide. A clareza na definição do objeto é vital para a modelagem de domínio, assegurando que as operações do sistema manipulem as entidades corretas do banco de dados ou da interface (Umar *et al.*, 2025).

Por fim, a Condição estabelece as restrições, gatilhos ou estados prévios necessários para a execução da funcionalidade. Conforme destacado por Gheorghe (2025) no contexto de sistemas complexos como IoT, a especificação explícita de condições é o principal diferenciador entre um requisito genérico e uma regra de negócio testável, sendo frequentemente a fonte de maior complexidade na validação automática.

2.2 *Large Language Models*

Os LLMs emergiram como uma inovação significativa no campo do PLN. A arquitetura *Transformer*, responsável por sua base, permite que esses modelos processem grandes volumes de dados de maneira eficiente, o que resulta em uma capacidade aprimorada de entender e gerar texto (Yang *et al.*, 2024). O aumento no número de parâmetros nos LLMs, que pode variar de bilhões a trilhões, proporciona um desempenho superior em uma ampla gama de tarefas de PLN, incluindo tradução, resumo e respostas a perguntas (Yao *et al.*, 2024).

A grande característica dos LLMs é sua capacidade de aprender e gerar texto contextualizado, o que os torna poderosos para uma variedade de aplicações. Eles são treinados com grandes conjuntos de texto e, uma vez treinados, conseguem aplicar esse conhecimento de maneira eficiente para tarefas que exigem compreensão linguística e raciocínio complexo, sem necessitar de re-treinamento explícito (Zhao *et al.*, 2025). O GPT-3, por exemplo, demonstro-se altamente capaz de aprender com exemplos fornecidos nos *prompts*, sem precisar de grandes quantidades de dados rotulados (Yang *et al.*, 2024). Isso permite que LLMs realizem tarefas de maneira mais dinâmica e adaptável, o que é um avanço em relação aos modelos tradicionais.

No entanto, conforme observado por Wei *et al.* (2023), os LLMs também enfrentam desafios, como a geração de respostas incoerentes ou alucinações, que podem ocorrer devido a falhas no processamento contextual ou treinamento em dados imprecisos. Essas limitações são um obstáculo significativo quando se busca gerar texto confiável e preciso, como é o caso de tarefas que exigem alta especialização, como raciocínio médico ou jurídico (Wei *et al.*, 2023). Além disso, a grande quantidade de parâmetros em LLMs pode resultar em altos

custos computacionais, tornando o acesso e a implementação desses modelos desafiadores para pesquisadores com recursos limitados (Rostam *et al.*, 2024).

Para melhorar a eficácia dos modelos, técnicas de EP têm sido desenvolvidas, permitindo que os usuários orientem o comportamento do modelo por meio de instruções detalhadas. Dessa forma, uso de técnicas como CoT pode melhorar significativamente o raciocínio dos modelos, especialmente em tarefas de múltiplas etapas (Wei *et al.*, 2023). Além disso, Zhao *et al.* (2025) mostrou que o *prompting* pode ser ajustado para otimizar tarefas específicas de PLN, promovendo respostas mais precisas e coerentes. Por fim, Rostam *et al.* (2024) observou que o uso da técnica ToT também tem se mostrado promissor, ajudando a estruturar o raciocínio de forma mais eficaz em problemas complexos.

2.2.1 ChatGPT

O GPT-5 marca uma mudança de paradigma na evolução dos LLMs, deixando de ser apenas um modelo mais potente para se tornar um sistema adaptativo de inteligência. Lançado pela OpenAI[1] em agosto de 2025, sua principal inovação está na arquitetura de roteamento em tempo real (*Real-time Router*), que avalia a complexidade de cada solicitação e escolhe dinamicamente entre dois caminhos: o *gpt-5-main*, otimizado para respostas rápidas e diretas, ou o *gpt-5-thinking*, voltado para raciocínio profundo através de cadeias de pensamento explícitas antes da geração da resposta final (OpenAI, 2025).

Diferente das versões anteriores, o GPT-5 foi desenvolvido como modelo verdadeiramente multimodal desde sua concepção, processando texto, imagem, áudio e vídeo nativamente em uma única rede neural. O treinamento incorporou aprendizado por reforço focado em raciocínio e dados sintéticos de alta qualidade para reduzir alucinações. Com janela de contexto expandida para 256.000 *tokens*, o modelo é capaz de analisar documentos extensos e repositórios completos de código. Nos *benchmarks*, estabeleceu novos recordes: 74,9% no SWE-bench Verified (resolução autônoma de problemas de engenharia de software) e 94,6% no AIME 2025 (matemática avançada) (OpenAI, 2025).

Apesar dos avanços, o modelo apresenta limitações própria, principalmente em seu modo de raciocínio profudo. Em determinados momentos, algumas respostas podem parecer lentas, compromentendo, assim, conversas casuais. Além disso, em comparação ao modelo anterior (GPT-4o), o atual pode parece menos empático ou mais “frio”. Ademais, a ocorrência de

[1] https://openai.com/

alucinações diminuiu consideravelmente em tarefas lógicas, mas ainda ocorrem ocasionalmente em domínios mais específicos ou quando há informações contraditórias.

Quanto a segurança, a OpenAI substituiu o sistema de recusas categóricas pelo paradigma de *Safe-Completions*, garantindo que o modelo exiba informações educacionais apropridades, ao invés de negar responder a tópicos sensíveis. Outro avanço foi a redução da tendência à adulação, onde agora o modelo questiona premissas incorretas, priorizando veracidade sobre concordância passiva (OpenAI, 2025).

Portanto, GPT-5 vai além da função de assistente conversacional, representando um avanço na automação de tarefas cognitivas. O modelo demonstra capacidade de utilizar diferentes ferramentas de forma coordenada e manter a coerência em processos de trabalho extensos. Enquanto o GPT-4o focava na interação humanizada em múltiplos formatos, o GPT-5 tem como foco a resolução efetiva de problemas e a execução autônoma de tarefas, evidenciando uma nova abordagem e um avanço nos LLMs.

2.2.2 DeepSeek

A família de modelos DeepSeek-R1 representa um avanço no desenvolvimento de LLMs especializados em raciocínio complexo, utilizando uma abordagem inovadora baseada em *Reinforcement Learning* (RL) para desenvolver capacidades avançadas de raciocínio (DeepSeek-AI, 2025). O DeepSeek-R1-Zero foi o primeiro modelo desta família, desenvolvido através de uma estratégia única: treinamento exclusivamente com aprendizado por reforço em larga escala, sem passar pela etapa tradicional de ajuste fino supervisionado. Apesar dessa abordagem não convencional, o modelo demonstrou ótima capacidade de raciocínio, alcançando 71% de acurácia no *benchmark* AIME 2024 (DeepSeek-AI, 2025).

Para este treinamento, foi utilizado o algoritmo GRPO aplicado diretamente sobre o modelo base DeepSeek-V3O GRPO apresenta importantes vantagens por melhorar o desempenho do modelo através de recompensas baseadas na precisão das respostas e na formatação adequada. Além disso, esse algoritmo evita problemas de *reward hacking* e elimina a necessidade de usar um modelo crítico separado, o que reduz os custos computacionais (DeepSeek-AI, 2025). O treinamento seguiu um formato estruturado específico, onde o raciocínio deve aparecer entre as marcações <*think*> e a resposta final dentro da *tag* <*answer*>. Essa estrutura permitiu que o modelo desenvolvesse estratégias próprias de raciocínio, como auto-verificação e reflexão (DeepSeek-AI, 2025).

O DeepSeek-R1 surgiu como uma evolução do modelo anterior, abordando algumas limitações identificadas no DeepSeek-R1-Zero. O modelo predecessor apresentava problemas frequentes como baixa legibilidade dos processos de raciocínio, além de misturar o idioma em certos momentos(DeepSeek-AI, 2025). Para resolver essas questões, o DeepSeek-R1 incorporou um estágio de *cold start* com dados supervisionados, tornando os processos de raciocínio mais legíveis e acessíveis. Essa inclusão foi fundamental para o sucesso do modelo, pois o conjunto de dados supervisionados contendo exemplos de CoT melhorou a legibilidade e ampliou o potencial de respostas do modelo (DeepSeek-AI, 2025). Os resultados demonstram a eficácia dessa abordagem: o DeepSeek-R1 alcançou desempenho comparável ao OpenAI-o1-1217 em tarefas matemáticas (97,3% no MATH-500) e de programação (DeepSeek-AI, 2025).

Além disso, o como DeepSeek-R1-Distill-Qwen-1.5B também se mostrou capaz em solucionar problemas, conseguindo bons resultados ao superar o GPT-4o em *benchmarks* matemáticos, atingindo 28,9% no AIME (DeepSeek-AI, 2025). Esse resultado comprova a escalabilidade da abordagem e sua aplicabilidade em diferentes tamanhos de modelo. Apesar desses avanços, o DeepSeek-R1 apresenta algumas limitações que precisam ser consideradas. O modelo demonstrou sensibilidade à formulação dos *prompts*, com desempenho reduzido em cenários *few-shot* (DeepSeek-AI, 2025). Ademais, em comparação com o DeepSeek-V3, o modelo mostrou-se inferior em tarefas específicas que exigem múltiplas interações, simulações complexas de papéis e geração de saídas estruturadas em formato JSON (DeepSeek-AI, 2025). Atualmente, o modelo está otimizado principalmente para chinês e inglês, podendo apresentar dificuldades quando utilizado com outros idiomas (DeepSeek-AI, 2025).

2.2.3 Gemini

O Gemini 3 marca uma evolução significativa na família de modelos do Google, representando sua mais recente aposta em sistemas de IA nativamente multimodais. Lançado em novembro de 2025, este modelo foi construído sobre uma arquitetura de Mistura de Especialistas Esparsa (*Sparse Mixture-of-Experts*) baseada em *Transformers*, capaz de processar de forma integrada diferentes tipos de entrada, como texto, áudio, imagem e vídeo, sem depender de componentes modulares separados (Google, 2025).

Entre as novidades mais relevantes desta geração está o modo *Deep Think*, uma configuração opcional que permite ao modelo intensificar seu raciocínio em problemas complexos. Esse modo funciona alocando mais recursos computacionais durante a inferência, o

que possibilita uma análise mais profunda antes da geração de respostas. Outro diferencial está na janela de contexto expandida para até 1 milhão de *tokens*, viabilizando o processamento de documentos extensos, grandes bases de dados e até repositórios completos de código em uma única sessão.

Nos testes de desempenho, o Gemini 3 Pro apresentou resultados expressivos quando comparado às versões anteriores e a outros modelos concorrentes. Na avaliação matemática AIME 2025, por exemplo, alcançou 95,0% de precisão. Já no *SWE-bench Verified*, voltado para tarefas de engenharia de software, obteve 76,2%, demonstrando capacidade robusta como agente autônomo em atividades de codificação (Google, 2025).

Ademais, o modelo passou por extensas rodadas de *Red Teaming*, isto é, simulações de ataques conduzidas por equipes especializadas, além de avaliações automatizadas focadas em mitigar riscos como discurso de ódio, conteúdo violento e manipulação. Segundo o relatório de *Frontier Safety*, embora o Gemini 3 Pro apresente capacidades avançadas em domínios como cibersegurança e raciocínio lógico, ele não atingiu os chamados Níveis de Capacidade Crítica (CCL), o que significa que não representa ameaças como proliferação de armas biológicas ou químicas (Google, 2025).

Apesar dos avanços, o modelo ainda carrega limitações comuns aos *Foundation Models*, como a tendência a gerar alucinações — respostas plausíveis, mas factualmente incorretas. Há também relatos ocasionais de lentidão ou *timeouts* ao processar tarefas especialmente complexas. Para mitigar esses e outros problemas, a Google implementou políticas rigorosas de uso, incluindo filtros de segurança que bloqueiam conteúdo sexualmente explícito, violento ou que possa violar direitos autorais e privacidade.

2.3 Engenharia de *Prompt*

A EP refere-se ao processo de projetar e otimizar instruções ou solicitações específicas para LLMs, com o objetivo de melhorar a qualidade e relevância das respostas geradas por esses modelos (Rawson; Reddivari, 2025; Koyuncu, 2025). Com a popularização dos LLMs, como o GPT, *Gemini* e outros, a EP surge como uma prática crítica para aproveitar efetivamente as capacidades desses sistemas em diversas aplicações, desde a categorização automática de relatórios de bugs até a geração de critérios de aceitação de software (Rawson; Reddivari, 2025; Koyuncu, 2025).

Diante dessa perpesctiva, Arawjo *et al.* (2023) proprôs uma uma plataforma visual de

programação projetada para simplificar a EP, denominda *ChainForge*, que permite aos usuários rapidamente experimentar e avaliar diferentes estratégias de prompts e seus efeitos sobre os resultados gerados pelos modelos. Tal ferramenta apoia a experimentação visual e facilita a identificação das melhores práticas de formulação de *prompts*, reduzindo a necessidade de conhecimento avançado em programação (Arawjo *et al.*, 2023).

Além disso, pesquisas destacam que a qualidade e a efetividade dos *prompts* estão diretamente relacionadas ao contexto em que são utilizados, além da especificidade das tarefas pretendidas (Henrickson; Meroño-Peñuela, 2025). Conforme evidenciado por Mao *et al.* (2025), pequenas mudanças no design dos *prompts* podem levar a variações significativas nos resultados obtidos pelos LLMs, enfatizando a necessessidade de sistematizar os templates de *prompts* para aplicações reais. Este aspecto torna evidente a necessidade de uma abordagem rigorosa e metódica no desenvolvimento de *prompts* para obter resultados consistentes e alinhados com objetivos específicos (Mao *et al.*, 2025).

Em uma abordagem hermenêutica, os *prompts* podem ser otimizados não apenas para obter respostas factualmente precisas, mas também para maximizar o valor interpretativo e hermenêutico das respostas geradas pelos LLMs (Henrickson; Meroño-Peñuela, 2025). Isso se deve, pois, a clareza das instruções e a especificidade do contexto fornecido ao modelo são cruciais para a produção de textos que sejam semanticamente ricos e interpretativamente relevantes, ampliando assim as possibilidades de aplicação prática desses modelos em contextos que exigem maior profundidade interpretativa (Henrickson; Meroño-Peñuela, 2025).

Técnicas avançadas e estruturadas de EP, têm mostrado resultados promissores, melhorando a capacidade de raciocínio e a geração de respostas mais robustas pelos LLMs (Masoudifard *et al.*, 2024). Esses métodos estruturados facilitam a geração de textos com maior coerência lógica e alinhamento com os requisitos desejados, evidenciando o potencial da EP para maximizar a eficácia e a precisão em tarefas complexas (Masoudifard *et al.*, 2024; Rawson; Reddivari, 2025).

Nesse cenário de profissionalização da interação com LLMs, emerge o conceito de *prompts* como artefatos de *software*. Diferentemente de consultas descartáveis em mecanismos de busca, os *prompts* em sistemas de Engenharia de Software (ES) complexos devem ser tratados com o mesmo rigor aplicado ao código-fonte. Isso implica que um *prompt* deve possuir ciclo de vida, ser passível de versionamento, testes de regressão e manutenção evolutiva (Vogelsang, 2024).

Tratar o *prompt* como um artefato engenheirado permite a rastreabilidade das decisões de projeto. Ao registrar as versões do *prompt*, o engenheiro de requisitos consegue isolar variáveis e atribuir a melhoria ou degradação da qualidade da saída especificamente à mudança na instrução, e não apenas à aleatoriedade do modelo. Essa abordagem transforma a EP de uma atividade exploratória *ad-hoc* para um processo sistemático e reprodutível.

2.3.1 Chain-of-Thought prompting

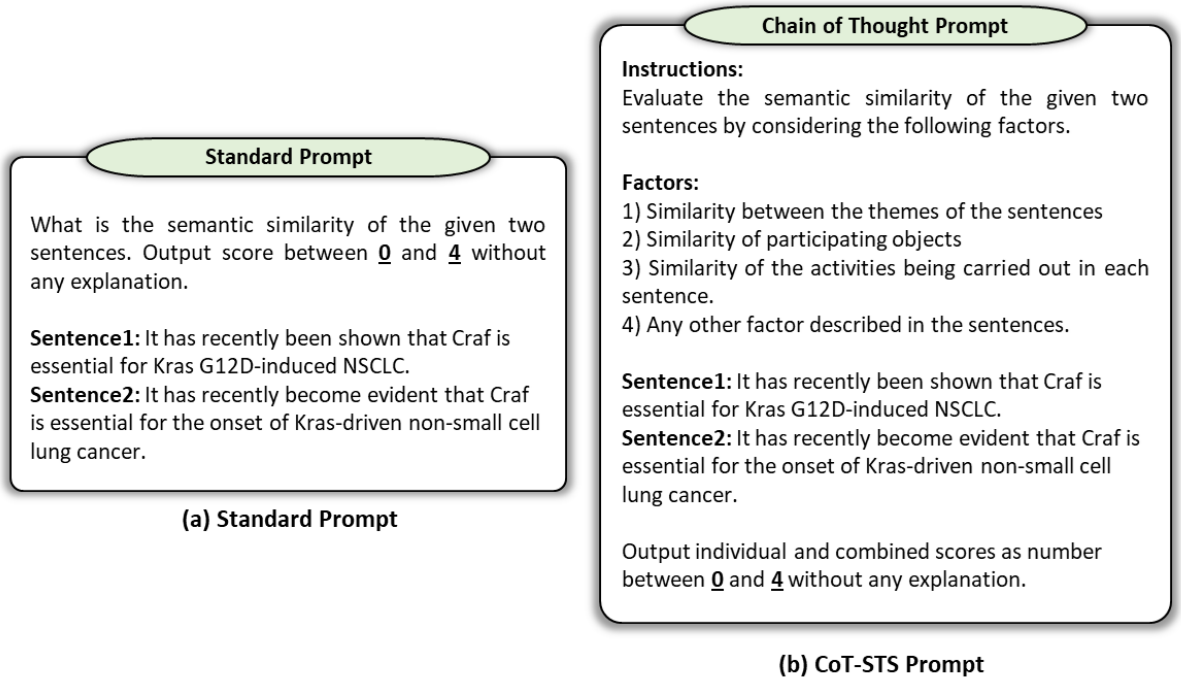
O método CoT tem se mostrado como uma abordagem promissora para aprimoramento da capacidade de raciocínio dos modelos dos LLMs, sobretudo, em tarefas que exigem múltiplas etapas de inferência e processamento lógico (Sheokand *et al.*, 2025; Wei *et al.*, 2023). Diferentemente das abordagens tradicionais que solicitam respostas diretas dos modelos, a técnica CoT estimula os LLMs a desenvolverem suas respostas de forma encadeada, isto é, explicitando o raciocínio intermediário necessário para então chegar à solução final (Hussain *et al.*, 2024; Wei *et al.*, 2023). Essa estruturação do raciocínio tem demonstrado ganhos relevantes em tarefas que envolvem interpretação textual, raciocínio matemático e inferência semântica, diminuindo problemas de alucinação e respostas incoerentes frequentemente geradas por modelos que não seguem essa estratégia (Rukmono *et al.*, 2024; Sheokand *et al.*, 2025).

Conforme evidenciado por Hussain *et al.* (2024), o uso da técnica CoT aplicada à tarefa de similaridade textual semântica elevou a correlação de Pearson de 0.45 para 0.72, quando comparada ao *prompt* padrão, demostrando como o encadeamento de pensamento melhora a performance em tarefas de comparação textual ao considerar aspectos como similaridade temática, objetos envolvidos, atividades realizadas e julgamento final com base em múltiplos fatores (Hussain *et al.*, 2024). A Figura 1 exemplifica a comparação entre um *prompt* padrão e a utilização da técnica CoT-STS, onde a tarefa foi desmembrada em quatro subtarefas, como parte do *framework Chain-of-Thought* (Hussain *et al.*, 2024).

2.3.2 Tree-of-Thought prompting

A Árvore de Pensamentos, ou ToT, é uma estratégia de EP que visa aprimorar a capacidade de resolução de problemas dos LLMs, especialmente em tarefas que exigem planejamento, exploração ou raciocínio estratégico (Long, 2023; Yao *et al.*, 2023). A técnica se inspira na abordagem da mente humana para resolver desafios por meio de tentativa e erro, explorando um espaço de soluções como uma árvore de pensamentos e permitindo retroceder

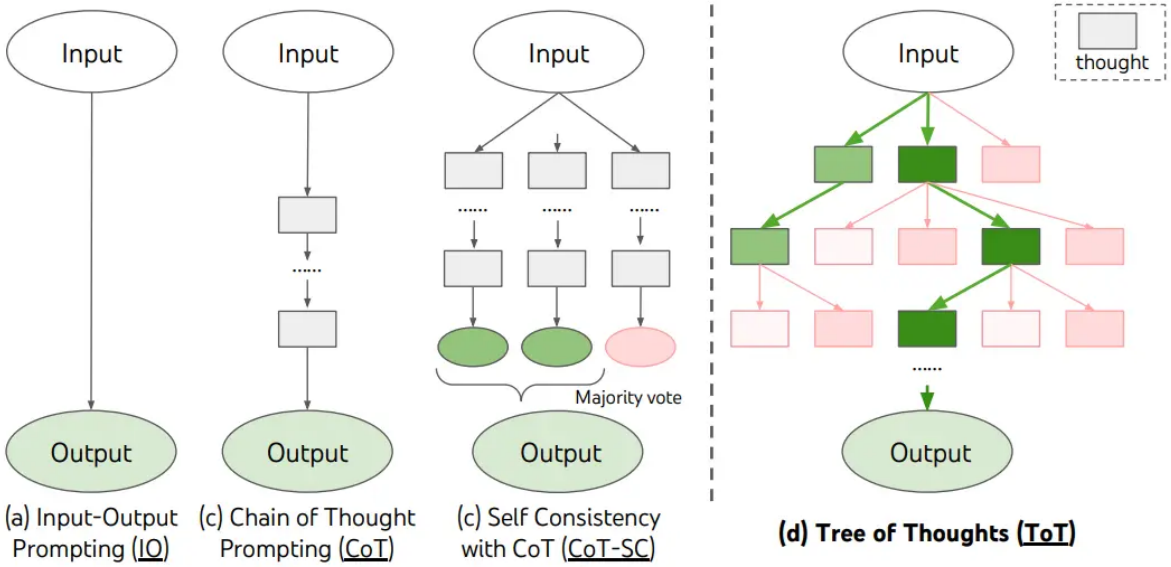
Figura 1 – *Prompt* padrão versus CoT-STS *prompt*.



Fonte: Hussain *et al.* (2024)

(*backtracking*) quando um caminho se mostra desfavorável (Long, 2023; Yao *et al.*, 2023). Diferente do CoT, que gera uma única cadeia de raciocínio, o ToT generaliza essa ideia ao permitir que o modelo explore múltiplos caminhos de raciocínio de forma paralela (Yao *et al.*, 2023).

Figura 2 – *Prompt* padrão versus CoT-STS *prompt*.



Fonte: Yao *et al.* (2023)

O funcionamento do ToT enquadra a resolução de um problema como uma busca

em uma árvore, onde cada nó representa um "pensamento", isto é, uma etapa intermediária e coerente para a solução (Long, 2023; Yao *et al.*, 2023). A principal inovação do ToT é o uso do próprio LLM não apenas para gerar os próximos pensamentos possíveis, mas também para deliberar e avaliar o potencial de cada um, agindo como uma heurística para guiar a busca (Yao *et al.*, 2023). Para operacionalizar esse processo, o LLM é frequentemente aumentado com módulos externos, como um agente de *prompt*, um módulo de verificação para checar a validade de cada passo, um módulo de memória para registrar o histórico e um controlador ToT que gerencia o processo e pode explicitamente comandar o retrocesso para explorar alternativas (Long, 2023).

A eficácia do ToT foi demonstrada em tarefas que desafiam os métodos lineares de raciocínio (Long, 2023; Yao *et al.*, 2023). Por exemplo, Long (2023) aplicou o *framework* para resolver o quebra-cabeça Sudoku, onde o ToT alcançou uma taxa de sucesso maior em comparação com abordagens de *prompts* mais simples. De forma complementar, o estudo de Yao *et al.* (2023) utilizou o "Jogo de 24"como um de seus principais experimentos, mostrando que a abordagem ToT elevou a taxa de sucesso para 74%, em contraste com os 4% obtidos pelo método de CoT.

A Figura 2, compara a técnica ToT com os outras técnicas existente de EP. É possível notar que, enquanto outras técnicas focam em uma abordagem mais sequencial e contínua para resolver problemas, a técninca ToT opta por uma árvore de pensamentos, onde cada pensamento uma sequência de linguagem coerente que serve como passo intermediário na solução do problema (Yao *et al.*, 2023).

3 TRABALHOS RELACIONADOS

Ao examinar a literatura atual, foram encontrados trabalhos relevantes que exploram diferentes perspectivas sobre a geração e avaliação da qualidade de artefatos produzidos por LLMs. Esta seção apresenta esses estudos, ressaltando suas contribuições e estabelecendo conexões com os objetivos deste projeto.

3.1 *ChatGPT Prompt Patterns for Improving Code Quality, Refactoring, Requirements Elicitation, and Software Design*

O trabalho de White *et al.* (2023) introduz o conceito de *prompt patterns* aplicados à Engenharia de *Software*. O objetivo principal do estudo é criar um catálogo de projetos de *prompt* reutilizáveis que servem como soluções para problemas comuns ao usar LLMs como o *ChatGPT*. O artigo foca especificamente em como esses padrões podem ser aplicados para automatizar e melhorar a qualidade de diversas atividades do ciclo de vida do *software*, incluindo a elicitação de requisitos, o projeto de sistemas e a refatoração de código.

A metodologia adotada pelos autores foi a de identificar, documentar e categorizar um conjunto de 13 padrões de *prompt*. A identificação desses padrões combinou a exploração de *prompts* já existentes na comunidade com a criação de novos para apoiar suas próprias tarefas de engenharia de *software*. Cada padrão é documentado de forma estruturada, similar aos padrões de projeto de *software*, contendo seções como intenção, motivação, estrutura e exemplo. Para a ER, eles detalham padrões específicos como o *Requirements Simulator Pattern*, que usa o LLM para simular um sistema e ajudar a identificar requisitos faltantes, e o *Specification Disambiguation Pattern*, que usa o LLM para encontrar ambiguidades em especificações.

Como resultado, o estudo apresenta um catálogo funcional de padrões que podem ser aplicados para mitigar erros comuns dos LLMs e para aproveitar suas capacidades de formas que seriam difíceis de automatizar com tecnologias tradicionais. Os autores concluem que, embora a intervenção humana qualificada ainda seja essencial, o uso desses padrões de *prompt* pode melhorar a qualidade dos artefatos gerados, como código e especificações, e acelerar o processo de desenvolvimento.

Este artigo se relaciona diretamente com o presente trabalho, pois ambos se concentram no uso da EP para aprimorar tarefas da ER. A principal diferença reside na abordagem, visto que White *et al.* (2023) focam na criação de um catálogo de padrões de conteúdo, ou seja,

maneiras de como estruturar as instruções para guiar o comportamento do LLM em alto nível. Já o presente trabalho, por outro lado, foca na avaliação empírica de estratégias de raciocínio, investigando como a maneira com que o LLM ”pensa“ e processa a informação impacta a qualidade do requisito gerado.

3.2 *Requirements Engineering using Generative AI: Prompts and Prompting Patterns*

O estudo de Ronanki *et al.* (2023b) aborda a necessidade de avaliar sistematicamente a eficácia de diferentes padrões de *prompt* para tarefas de ER. O objetivo principal do trabalho é determinar quais técinas de EP são mais eficazes para tarefas específicas de ER, preenchendo uma lacuna na literatura onde, apesar do reconhecimento da importância da EP, faltam métodos para selecionar a melhor abordagem para um determinado contexto.

Para atingir esse objetivo, os autores conduziram um estudo empírico utilizando a API do *GPT-3.5 turbo*. A metodologia consistiu em selecionar cinco padrões de *prompt* da literatura, como *Cognitive Verifier*, *Persona* e *Question Refinement*, e aplicá-los a duas tarefas de ER: classificação de requisitos em Requisitos Funcionais (RF) e Requisitos Não Funcionais (RNF), além das rastreabilidade de requisitos. A performance de cada padrão foi medida de forma quantitativa, utilizando métricas de *precision*, *Recall*, *F1-Score* e *Accuracy*, com os experimentos sendo executados em diferentes configurações de "temperatura"do modelo para avaliar a consistência dos resultados.

Os resultados da pesquisa indicaram que não existe um único padrão de *prompt* que seja o melhor para todas as tarefas. O padrão *Question Refinement* demonstrou ser o mais consistente e com melhor desempenho geral com a maioria as tarefas. Contudo, padrões como o *Context Manager* mostraram alta variabilidade e baixa performance. Os autores concluem oferecendo um *ranking* de recomendação dos padrões para as tarefas analisadas e propõem sua metodologia como um *framework* geral para que outros pesquisadores e praticantes possam avaliar a eficácia de *prompts* em outros contextos de ER.

Este artigo possui relação com o presente trabalho, pois ambos realizam um estudo empírico para comparar diferentes estratégias de EP no domínio da ER. Apesar disso, existem algumas diferenças, pois Ronanki *et al.* (2023b) focam na análise e classificação de requisitos já existentes, enquanto o presente estudo foca na geração de novos requisitos. Além disso, Ronanki *et al.* (2023a) abordam outras técnicas de EP como *Context Manager* e *Persona*, que definem "o que"e "como"o LLM deve se apresentar, enquanto o presente trabalho avalia estratégias de

raciocínio que definem "como"o LLM deve processar a informação para chegar a uma solução.

3.3 *Leveraging Graph-RAG and Prompt Engineering to Enhance LLM-Based Automated Requirement Traceability and Compliance Checks*

Masoudifard *et al.* (2024) propõem e avaliam um *framework* automatizado para verificar a conformidade de Requisitos de *Software* com regulações e documentos de mais alto nível. O objetivo principal do estudo é aprimorar a precisão da rastreabilidade e da verificação de conformidade em domínios altamente regulados, como o financeiro e o aeroespacial, onde a aderência a padrões é crítica.

A metodologia empregada combina uma abordagem de *Graph-RAG* jutamente com técnicas avançadas de Engenharia de *Prompt*. O *Graph-RAG* é utilizado para extrair de forma mais precisa o contexto relevante de documentos normativos. Em seguida, estratégias como CoT e ToT são aplicadas para guiar o raciocínio do LLM (*GPT-4o*) na tarefa de detectar se um requisito de baixo nível viola ou não a norma de referência recuperada. O estudo utiliza dois conjuntos de dados reais para a sua avaliação.

Os resultados da pesquisa demonstram que a combinação de *Graph-RAG* com *prompts* avançados, especialmente ToT, supera as abordagens de *RAG* tradicionais. A *F1-Score* na detecção de requisitos não conformes foi notavelmente maior, validando a eficácia do *framework*. Os autores concluem que, embora poderosa, a solução possui alto custo computacional e uma complexidade de implementação que deve ser levado em consideração.

Este artigo se relaciona diretamente com o presente trabalho pelo uso compartilhado das estratégias de CoT e ToT para aprimorar o raciocínio de LLMs no domínio de requisitos. A principal diferença, no entanto, reside no objetivo da aplicação. Enquanto Masoudifard *et al.* (2024) focam na validação e verificação de conformidade de requisitos já existentes, o presente trabalho foca na geração de novos requisitos, avaliando como as diferentes estratégias de *prompt* influenciam a qualidade do artefato gerado, em vez de sua conformidade com um padrão externo.

3.4 *A ChatGPT-powered Prompt Engineering Framework for Generating Software Acceptance Criteria*

O estudo de Rawson e Reddivari (2025) aborda a criação de Critérios de Aceitação para Histórias de Usuário. O objetivo do trabalho é propor e avaliar um *framework* que utiliza

a engenharia de *prompts* no *ChatGPT* para automatizar a geração desses critérios, que são essenciais para validar se uma funcionalidade foi implementada corretamente.

A metodologia da pesquisa consistiu no desenvolvimento de um *framework* de *prompting* estruturado. Neste *framework*, uma História de Usuário é fornecida como entrada para o LLM, e o *prompt* é cuidadosamente projetado para guiar o modelo a gerar uma lista de critérios de aceitação relevantes, claros e testáveis. A avaliação da eficácia do *framework* foi conduzida analisando a qualidade dos critérios gerados em comparação com critérios de qualidade predefinidos para este tipo de artefato de requisito.

Os resultados indicaram que o uso de um *framework* de *prompting* bem definido pode ser capaz auxiliar na geração de Critérios de Aceitação a partir de Histórias de Usuário. Contudo, a intervenção humana ainda seja necessária para refinar e validar a saída final. Apesar disso, essa abordagem automatizada é capaz de acelerar o processo, reduzindo assim o esforço manual e servindo como um excelente ponto de partida para a equipe de desenvolvimento e teste.

A relação do trabalho de Rawson e Reddivari (2025) com o presente trabalho é clara, pois ambos utilizam a engenharia de *prompts* e LLMs para gerar artefatos no domínio da ER. No entanto, o foco e a abrangência são distintos. A maior diferença está no artefato que será gerado ao final, poist Rawson e Reddivari (2025) foca na geração de Critérios de Aceitação a partir de Histórias de Usuário. Já o presente estudo investiga a geração de RF e RNF a partir de fontes menos estruturadas e, além de comparar o impacto de diferentes estratégias de EP na qualidade geral dos requisitos gerados.

3.5 Análise Comparativa

O Quadro 1 apresenta uma análise comparativa detalhada entre o presente estudo e os trabalhos relacionados. A comparação foi estruturada em quatro dimensões principais para evidenciar as especificidades metodológicas de cada pesquisa: (i) Objeto de Estudo, identificando o foco central da investigação; (ii) Técnicas de Avaliação, descrevendo como os resultados foram mensurados; (iii) Participantes/Fonte, indicando a origem dos dados ou quem realizou os experimentos; e (iv) Objetivo do Estudo, clarificando a meta final de cada trabalho. Essa estruturação permite situar este TCC no contexto da literatura, destacando sua abordagem específica na geração de requisitos via LLMs.

Dessa forma, ao analisar a interação entre os estudos, é possível notar um diferença de foco entre os mesmos. O presente trabalho, juntamento com White *et al.* (2023) e Rawson

e Reddivari (2025), foca na geração de novos requisitos, porém, usando outras técnicas de EP. Já estudos como os de Ronanki *et al.* (2023b) e Masoudifard *et al.* (2024) focam na análise de requisitos já existentes, como classificação e verificação de conformidade, diferente do presente trabalho que foca na geração de novos requisitos. Além disso, este estudo utiliza as mesmas técnicas de raciocínio avançado (CoT e ToT) de Masoudifard *et al.* (2024), porém, aplicando essa técnicas para a geração de novos requisitos. Já avaliação da qualidade intrínseca do artefato, também pode ser observado no estudo de Rawson e Reddivari (2025), que priorizam métricas de desempenho de tarefa.

Quadro 1 – Quadro comparativo entre trabalhos relacionados e este estudo

TRABALHOS	**OBJETO DE ESTUDO**	**TÉCNICAS DE AVALIAÇÃO**	**PARTICIPANTES / FONTE**	**OBJETIVO DO ESTUDO**
White *et al.* (2023)	Catálogo de Padrões de Prompt para Engenharia de Software.	Definição estrutural de padrões e validação por exemplificação.	Autores (Análise de padrões existentes).	Criar um catálogo de soluções reutilizáveis para automatizar elicitação e refatoração.
Ronanki *et al.* (2023b)	Eficácia de diferentes padrões (Persona, etc.) na ER.	Análise Quantitativa (Precision, Recall, F1-Score).	Datasets de requisitos pré-existentes.	Identificar qual técnica de prompt é mais eficaz para tarefas de classificação e rastreabilidade.
Masoudifard *et al.* (2024)	Framework automatizado (Graph-RAG + CoT/ToT).	Métricas de desempenho (F1-score) comparadas ao RAG tradicional.	Datasets de domínios regulados (Financeiro e Aeroespacial).	Melhorar a precisão na verificação de conformidade e rastreabilidade de requisitos.
Rawson e Reddivari (2025)	Framework para geração de Critérios de Aceitação.	Avaliação da qualidade baseada em critérios pré-definidos (testabilidade).	Estudo de caso conduzido pelos autores.	Automatizar a criação de critérios de aceitação a partir de Histórias de Usuário.
ESTE TRABALHO	**Estratégias de Raciocínio (IO, CoT, ToT) na geração de novos requisitos.**	**Abordagem Híbrida: Quantitativa (Estrutural) e Qualitativa (Clareza, Completude, Consistência).**	**Autor e Múltiplos LLMs (GPT, Gemini, DeepSeek).**	**Investigar o impacto das estratégias de raciocínio na qualidade intrínseca de RF e RNF gerados do zero.**

Fonte: Elaborado pelo autor.

4 PROCEDIMENTOS METODOLÓGICOS

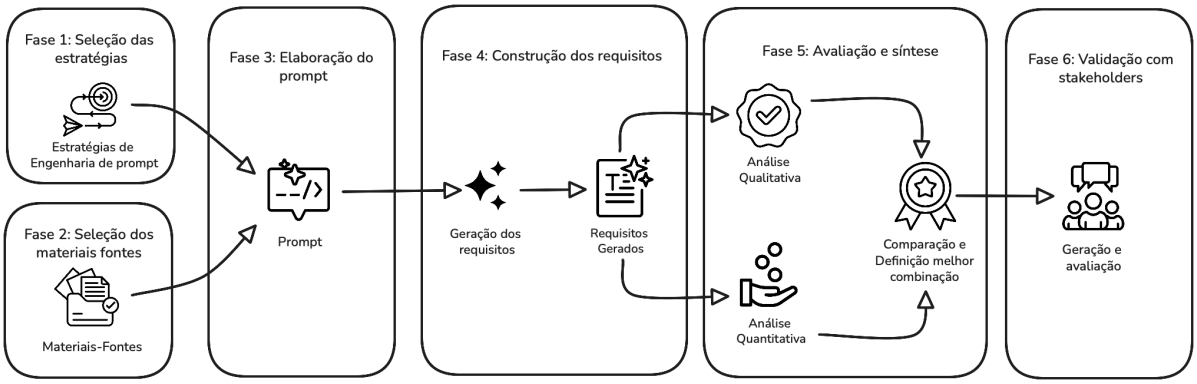
Nesta seção, serão apresentados os passos necessários para a realização deste trabalho. O estudo está estruturado em etapas específicas para avaliar a qualidade dos requisitos gerados por LLMs a partir de diferentes estratégias de EP. Os procedimentos metodológicos adotados são descritos a seguir, detalhando cada etapa do estudo. Para realizar a avaliação da qualidade dos requisitos, o estudo foi estruturado em seis etapas: (i) Seleção e categorização das estratégias de *prompt*; (ii) Materiais-Fonte para a Construção dos *Prompts*; (iii) Construção dos *prompts* a partir dos materias selecionados; (iv) Geração dos Requisitos com os LLMs; (v) Estratégia de Avaliação e Análise dos Requisitos Gerados; (vi) Validação dos requisitos com *stakeholders*.

Quadro 2 – Questões de pesquisa

ID	Questão de Pesquisa	Objetivo
QP1	Como as LLMs podem melhorar a qualidade dos requisitos funcionais gerados, através de uma análise quantitativa e qualitativa.	Esta questão busca compreender como as LLMs influenciam a qualidade dos requisitos gerados, analisando os critérios de clareza, completude, consistência e relevância
QP2	Como as técnicas de engenharia de *prompt* podem otimizar as respostas geradas pelas LLMs, melhorando a geração de requisitos em contextos de desenvolvimento de software?	Esta questão busca entender como diferentes estratégias de engenharia de *prompt* influenciam a qualidade das respostas das LLMs, especialmente na geração de requisitos de software.
QP3	Qual combinação de técnica de engenharia de *prompt* e LLM produz os requisitos de software com a maior qualidade geral?	Esta questão busca entender, a partir dos resultados das análises quantitativa e qualitativa, qual a combinação mais eficaz entre as técnicas e os LLMs avaliados.
QP4	Em que medida os requisitos gerados pela melhor combinação de LLM e técnica de *prompt* são aceitos por *stakeholders* em cenários reais de uso?	Esta questão busca validar a utilidade prática dos artefatos gerados, mensurando a taxa de aceitação por especialistas do domínio e identificando lacunas de regras de negócio não detectadas na análise estrutural.

Fonte: Elaborado pelo autor.

Figura 3 – Fluxo procedimentos metodológicos



Fonte: Criado pelo autor.

4.1 Seleção e Categorização das Estratégias de Engenharia de *prompt*

A primeira etapa da nossa metodologia é a seleção das estratégias de *prompt* que serão comparadas. Para este estudo, foram escolhidas três abordagens que representam uma escala de complexidade e capacidade de raciocínio: (IO), (CoT) e (ToT). A ideia é comparar como cada um desses métodos, do mais simples ao mais avançado, influencia a qualidade dos requisitos gerados por um LLM.

Cada técnica funciona de uma maneira única. O IO é o método mais básico e servirá como nosso ponto de comparação; ele consiste em uma pergunta direta esperando uma resposta final, sem detalhes do processo. O CoT é um avanço, instruindo o modelo a pensar passo a passo, ou seja, a detalhar sua linha de raciocínio antes de chegar à conclusão. Por fim, o ToT é a abordagem mais sofisticada, permitindo que o LLM explore múltiplos caminhos de raciocínio simultaneamente, avalie qual é o mais promissor e até mesmo retroceda para tentar uma nova abordagem, simulando um processo de resolução de problemas mais humano e deliberado.

4.2 Seleção de Materiais-Fonte para a Construção dos *Prompts*

A base para a construção dos *prompts* fundamenta-se em materiais extraídos de documentos de especificação de requisitos e projetos de código aberto abrangendo cinco domínios distintos, os quais estão detalhados no Quadro 3. Esses materiais foram selecionados pela sua completude, oferecendo uma visão clara das necessidades de cada sistema. Para mitigar potenciais vieses na seleção e extração das informações, a análise dos domínios contou com a participação de estudantes da área de computação.

Os requisitos existentes são cruciais, pois demonstram às LLMs qual o formato que está sendo utilizado no projeto. Eles servem como exemplos diretos, instruindo a LLM a gerar novos requisitos que sejam condizentes e mantenham o padrão de qualidade e detalhamento já estabelecido previamente.

Já a descrição do contexto do sistema fornece uma ideia geral da aplicação, incluindo seus principais objetivos de negócio e os perfis de usuário, permitindo, assim, que o LLM seja capaz de gerar os requisitos faltantes de maneira alinhada ao escopo total e à funcionalidade central estipulada.

Dessa forma, os dois materiais em conjunto permitem a construção de *prompts* mais elaborados e alinhados ao escopo real do projeto, fornecendo uma base sólida para a geração de

requisitos de alta qualidade pelas LLMs.

Quadro 3 – Caracterização dos domínios e sistemas utilizados como fonte para a geração de requisitos

ID	Domínio / Sistema	Descrição do Escopo e Objetivo
D1	FakeNet-NG (Cibersegurança)	Ferramenta de análise dinâmica de malware que intercepta tráfego de rede, desviando-o para servidores falsos (Listeners) para impedir a conexão real com a internet e permitir análise segura.
D2	Financinhas (Educação)	Plataforma gamificada de educação financeira para o ambiente escolar. Utiliza quizzes e desafios ("Aventuras do Dinheirinho") para ensinar crianças e adolescentes a lidar com dinheiro.
D3	Abrace (Assistência Social)	Plataforma web para a "Casa da Caridade de Quixadá", focada em centralizar a divulgação de ações sociais, gerenciar doações via Pix e aumentar a credibilidade da instituição.
D4	Restaurante Universitário (Gestão Alimentar)	Sistema de gerenciamento para RUs, focado em controle de demanda, reservas antecipadas de refeições, gestão de cardápios e redução de desperdício alimentar.
D5	Cardio Surgery Illustrator (Saúde)	Solução móvel para auxiliar pacientes e estudantes no acompanhamento e educação sobre saúde cardiovascular. Integra triagem inteligente, visualização interativa 2D de condições cardíacas.

Fonte: Elaborado pelo autor.

4.3 Construção dos *prompts* a partir dos materias selecionados

Uma vez definidos os materiais-fonte, o passo seguinte é a construção efetiva dos *prompts*. Este processo é uma seleção manual e criteriosa dos trechos mais relevantes de cada documento.

Para os requisitos, o processo envolve selecionar em torno de 25% a 30% dos requisitos existentes no documento original. Esta seleção busca selecionar aqueles requisitos que possuem todas os critérios objetivos(Ator, Ação, Objeto, Condição) além de uma linguagem clara e concisa. O objetivo é fornecer à LLM um conjunto de exemplos de alta qualidade e com valor.

Para o contexto do sistema, a seleção foca em extrair a descrição da motivação do *software*, isto é, qual problema ela soluciona, além da definição dos atores do sistema. Esses elementos são cruciais para garantir que os novos requisitos gerados pela LLM sejam

funcionalmente relevantes e não apenas estruturalmente corretos.

Este processo de seleção detalhada possibilita aplicar as diferentes técnicas de EP de forma eficaz, direcionando o retorno do LLM para que ele atenda a um requisito ou a uma demanda específica com maior precisão. A Figura 4 exemplifica um exemplo de *prompt* submetido à LLM, com algumas informações omitidas, para melhor entendimento da imagem.

Figura 4 – Exemplos de *prompt* com a técnica IO

```
# CONTEXTO DO SISTEMA

O projeto é o "Cardio Surgery Illustrator", uma solução móvel inovadora para saúde e bem-estar. O objetivo principal é auxiliar estudantes e
pacientes no acompanhamento de condições de saúde cardiovascular, conectando informações complexas de maneira intuitiva e acessível. O aplicativo comb
ina funcionalidades de monitoramento de saúde, educação e localização de serviços médicos. Seus principais stakeholders são os pacientes, que desejam
acompanhar sua saúde cardíaca, receber recomendações e entender suas condições, e os estudantes, que buscam aprimorar conhecimentos sobre saúde cardio
vascular através de materiais e quizzes. O sistema inclui funcionalidades chave como formulários inteligentes para diagnóstico preliminar, visualizaçã
o interativa em 2D de condições cardíacas, mapas de clínicas, assistente virtual (chatbot), módulos educativos e uma comunidade para troca de experiên
cias.

# EXEMPLOS DE REQUISITOS (30%)

Abaixo estão alguns exemplos de requisitos funcionais e não-funcionais do sistema, derivados das histórias de usuário iniciais:

## RF001 - Realizar Triagem Inteligente
O sistema deve oferecer um formulário inteligente ("Como vai a sua saúde cardiovascular?") que coleta dados sobre hábitos e fatores de risco do pacien
te para gerar uma avaliação preliminar e direcionar possíveis diagnósticos.

...

## RF016 - Realizar Quiz Educacional
O sistema deve oferecer quizzes interativos sobre saúde cardiovascular voltados para estudantes, fornecendo feedback imediato sobre as respostas para
testar e aprimorar o conhecimento.

# INSTRUÇÕES

Com base no Contexto do sistema e nos requisitos fornecidos, gere os demais requisitos funcionais faltantes do sistema.
```

Fonte: Criado pelo autor.

4.4 Geração dos Requisitos com os LLMs

Nesta etapa, inicia-se o processo prático de geração dos requisitos. Os *prompts*, construídos a partir dos materiais-fonte, são inseridos nas ferramentas de LLM selecionadas: *ChatGPT*, *DeepSeek* e *Gemini*. O objetivo é aplicar as diferentes estratégias de EP para obter as versões dos requisitos. A geração buscará contemplar tanto RF, que descrevem as funcionalidades do sistema, como também RNF, que abordam os atributos de qualidade, como desempenho e segurança, e como o sistema deve se comportar. Para isso, os *prompts* foram elaborados para extrair esses diferentes tipos de informação dos trechos das entrevistas e anotações.

Ao final desta fase, o resultado consistiu em um conjunto de requisitos de *software* gerados por cada modelo e estratégia. O volume total de artefatos produzidos oscilou entre 10 e 20 requisitos por execução, dependendo da verbosidade de cada LLM. Para fins de padronização da amostra comparativa e viabilidade da análise manual, estabeleceu-se um recorte metodológico nos 10 primeiros requisitos retornados. Essa decisão fundamenta-se no comportamento típico

de uso de ferramentas de IA generativa, onde a qualidade e relevância dos primeiros resultados apresentados são determinantes para a utilidade da ferramenta sob a ótica do usuário.

4.5 Estratégia de Avaliação e Análise dos Requisitos Gerados

Por fim, será feito a avaliação empírica da qualidade dos requisitos produzidos. O conjunto de dados para esta fase consiste em uma base estruturada contendo cada requisito gerado e seus metadados associados: o LLM de origem e estratégia de EP utilizada. Será realizado tanto uma análise qualitativa, como também uma análise quantitativa.

4.5.1 Análise Qualitativa

A análise qualitativa consiste na avaliação da qualidade semântica e a adequação de cada requisito ao contexto do projeto. Para isso, serão utilizados quatro critérios de qualidade:

- **Clareza:** Avalia se o requisito é compreensível, direto e livre de ambiguidades que possam dificultar o entendimento por parte dos *stakeholders*.
- **Completude:** Verifica se o requisito contém todas as informações necessárias para sua implementação, sem omissões ou lacunas importantes.
- **Consistência:** Analisa a coesão interna do conjunto, assegurando que não existem contradições lógicas dentro de um mesmo requisito ou entre requisitos distintos.
- **Relevância:** Examina a pertinência do requisito no contexto do sistema, garantindo que ele representa uma necessidade válida e alinhada aos objetivos do projeto.

Será atribuída uma nota para cada critério em uma escala de 1 (insatisfatório) a 5 (excelente). Essa etapa garante uma análise detalhada das particularidades de cada linguagem e cada LLM utilizada durante o procedimento.

4.5.2 Análise Quantitativa

Para a análise quantitativa, será realizada uma análise de percentual, para entender qual a proporção de requisitos que atende a critérios estruturais pré-estabelecidos. A metodologia, inspirada em trabalhos como os de Masoudifard *et al.* (2024), se baseia na verificação de atributos estruturais.

Será estabelecido um *checklist* com os quatro atributos esperados:

- **Ator:** Refere-se a quem ou o que executa a ação, como um tipo de usuário ou outro

sistema. É a entidade que inicia ou interage com o requisito.

- **Ação:** Descreve o processo ou a operação específica que deverá ser realizada pelo sistema. Corresponde ao verbo principal do requisito, indicando assim a funcionalidade, como por exemplo, criar, manter, validar.
- **Objeto:** Indica a entidade sobre a qual a ação é executada. É o alvo da operação descrita no requisito. Por exemplo, relatória de venda, conta do cliente.
- **Condição:** Define as restrições, pré-condições ou circunstâncias sob as quais a ação deve ocorrer. Ou seja, quais regras definem a execução do requisito.

Os requisitos gerados serão anotados manualmente para checar a presença dos quatro atributos estruturais. Com base nisso, será calculado o percentual de conformidade para cada atributo, oferecendo uma medida objetiva da capacidade de cada método.

4.5.3 Análise Comparativa e Síntese dos Resultados

A etapa final da metodologia consiste na análise comparativa dos dados coletados. Os escores médios da análise qualitativa e os percentuais da análise quantitativa serão cruzados para cada uma das combinações de LLM e técnica de EP. O objetivo desta comparação é ranquear o desempenho das diferentes abordagens e determinar qual delas se mostrou mais eficaz na geração de requisitos de alta qualidade, respondendo assim às questões de pesquisa deste trabalho.

4.6 Validação de Requisitos com Stakeholders

Após a etapa analítica, que identifica a combinação de LLM e técnica de EP com melhor desempenho estrutural e qualitativo, o estudo avança para uma análise e validação prática com *stakeholders* reais. O objetivo é verificar se a técnica selecionada é capaz de gerar requisitos semanticamente corretos e aderentes às regras de negócio, sob a ótica de especialistas do domínio.

Os domínios selecionados para esta validação foram Nutrição e Engenharia de Produção, representados por estudantes de graduação destas respectivas áreas, atuando como clientes reais para a elicitação e validação dos requisitos.

4.6.1 Caracterização dos Participantes

A população-alvo desta etapa da pesquisa consiste em potenciais usuários com domínio sobre as regras de negócio abordadas pelo sistema. Para a validação, foram selecionados

dois participantes, ambos acadêmicos em fase de conclusão de curso nas áreas de Nutrição e Engenharia de Produção.

Esse perfil foi escolhido para unir a teoria que eles veem na faculdade com a prática do mercado de trabalho. Como ambos já possuem experiência de estágio, eles conseguem trazer uma visão mais real para a validação. O perfil detalhado de cada participante está no Quadro 4.

O Quadro 4 resume o perfil dos validadores que atuaram como *stakeholders* nesta etapa.

Quadro 4 – Perfil dos Participantes

Cód	**Perfil**
P1	Graduando em Nutrição, em fase de conclusão de curso, com experiência profissional prática adquirida através de estágios na área.
P2	Graduando em Engenharia de Produção, em fase de conclusão de curso, com experiência profissional prática adquirida através de estágios na área.

Fonte: Elaborado pelo autor.

4.6.2 Protocolo de Validação: Leitura Baseada em Cenários

Para mitigar o viés de aquiescência, fenômeno comum em validações onde o usuário tende a concordar passivamente com a especificação apresentada pela "autoridade"do sistema, optou-se por não utilizar uma revisão *ad-hoc*. Em seu lugar, foi adotada uma adaptação da técnica de Leitura Baseada em Cenários, fundamentada nos estudos de Porter *et al.* (1995).

Esse estudo aponta que revisores detectam mais defeitos quando guiados por cenários de uso específicos do que através de leitura livre. Dessa forma, a entrevista segue as seguintes etapas:

1. **Preparação (Cenários de Falha):** Antes da entrevista, os participantes foram instruídos a trazer casos de uso complexos ou situações de exceção reais de seus domínios para cada um dos requisitos.
2. **Confronto:** Durante a sessão, cada requisito gerado pela melhor técnica é lido e comparadada com o cenário trazido pelo participante.
3. **Veredito:** O requisito é classificado em uma das três categorias a seguir, dependendo da gravidade dos problemas encontrados:
   - **Aceito:** O requisito cobre satisfatoriamente o cenário de teste, está alinhado às regras de negócio e não apresenta erros semânticos ou estruturais.
   - **Aceito com Restrições:** O requisito é pertinente e estruturalmente válido, mas

apresenta falhas pontuais que exigem ajustes para ser considerado pronto para implementação.

– **Rejeitado:** O requisito apresenta falhas graves ou não soluciona o problema proposto pelo cenário, devendo ser descartado.

4.6.3 Taxonomia de Defeitos e Análise

Caso o requisito não seja integralmente aceito, o participante deve justificar a falha classificando-a de acordo com uma taxonomia de defeitos adaptada de Wiegers e Beatty (2013). Esta categorização permite transformar a validação subjetiva em dados analíticos sobre as limitações da técnica, podendo ser observadas no Quadro 5.

Quadro 5 – Taxonomia de Defeitos para Validação com Stakeholders

Tipo de Defeito	**Definição e Critério de Identificação**
Omissão	O requisito falha em descrever uma regra de negócio, uma exceção ou uma condição necessária para o cenário real.
Incorreção Factual	O requisito descreve uma funcionalidade ou regra que é tecnicamente inviável ou falsa dentro do domínio de conhecimento.
Ambiguidade	O texto gerado permite mais de uma interpretação, podendo levar a implementações errôneas.
Inconsistência	O requisito entra em conflito com outro requisito gerado ou com uma regra de negócio pré-existente citada na entrevista.
Irrelevância	O requisito está estruturalmente correto, mas não agrega valor ao negócio ou descreve uma obviedade desnecessária (Baixo valor de negócio).

Fonte: Adaptado de Wiegers e Beatty (2013).

Ao final desta etapa, será possível quantificar não apenas a taxa de aceitação dos requisitos gerados por Inteligência Artificial (IA), mas identificar quais tipos de defeitos persistem mesmo com o uso de técnicas de EP, fornecendo uma resposta à QP4.

5 AVALIAÇÃO DOS REQUISITOS GERADOS POR LLMS

Neste capítulo, serão apresentados os resultados obtidos nesse estudo, em conjunto com as quatro questões de pesquisa que foram levantadas anteriormente. Cada questão de pesquisa é apresentada como uma seção, onde primeiro é dada a resposta e, em seguida, são apresentadas as análises que justificam essa resposta. As análises realizadas estão divididas em três tipos principais: (i) análise estrutural quantitativa; (ii) análise da qualidade intrínseca qualitativa; e (iii) validação com stakeholders. Foi criado um repositório[1] no Google Drive, com todos os requisitos gerados e suas respectivas notas e avaliações. Além no Apêndice A é possível visualizar as médias das análise quantitativa e qualitativa, filtradas por domínio.

5.1 QP_1: Como as LLMs podem melhorar a qualidade dos requisitos funcionais gerados?

Os dados indicam que todos os LLMs foram capazes de gerar requisitos com alta completude estrutural, atingindo constantemente 100% nos atributos de Ação e de Objeto. Porém, vale ressaltar que o atributo de Condição ainda se tornou um desafio para grande parte dos modelos. Na estratégia IO, tanto o GPT quanto o DeepSeek alcançaram apenas 60% de presença nesse atributo, enquanto o Gemini obteve 75,76%.

Assim, a aplicação de técnicas de EP foram fundamentais para melhorar esse aspecto. O modelo GPT apresentou a evolução mais expressiva, visto que, ao utilizar CoT e ToT, a presença de Condição foi elevada de 60% para 81,11%, demonstrando que estratégias estruturadas de raciocínio são capazes de estimular os modelos a explorarem aspectos mais complexos da especificação. Dessa forma, o GPT demonstrou ser o modelo que melhor desempenhou, sobretudo ao se utilizar de técnicas de EP, apresentando evolução nas métricas objetivas.

Além disso, o quadro 6 apresenta os resultados da análise quantitativa obtidos, sumarizando as médias de desempenho de cada LLM submetido às três estratégias de EP. Os dados foram unificados considerando a média aritmética dos cinco domínios analisados, permitindo uma visualização comparativa a respeito da completude estrutural.

A seguir, detalha-se o comportamento estrutural específico de cada modelo.

[1] https://drive.google.com/drive/folders/1I_aZ9nsG-34aBwncDFGmkSh_GGcf4nOn?usp=sharing

Quadro 6 – Resultados da Análise Quantitativa: Completude Estrutural por Modelo e Técnica

LLM	Técnica	Ator (%)	Ação (%)	Objeto (%)	Condição (%)
GPT	IO	97,78	96,67	100,00	60,00
	CoT	91,11	100,00	100,00	81,11
	ToT	92,22	100,00	100,00	81,11
Gemini	IO	96,67	100,00	100,00	75,76
	CoT	93,33	100,00	100,00	70,60
	ToT	91,67	100,00	100,00	75,83
DeepSeek	IO	97,78	100,00	100,00	60,00
	CoT	96,67	100,00	100,00	68,89
	ToT	100,00	100,00	100,00	74,40

Fonte: Elaborado pelo autor.

5.1.1 Gemini

O modelo Gemini destacou-se por apresentar o melhor desempenho sem engenharia de *prompt* avançada. Na estratégia IO, ele liderou isoladamente o atributo Condição com 75,76%, superando os demais modelos em quase 15%. Além disso, foi o único modelo a atingir 100% de precisão no atributo Ação em todas as estratégias testadas.

Além, os dados indicam que o Gemini é menos sensível às técnicas de *prompt* para fins estruturais. A aplicação da técnica CoT resultou em uma leve regressão na identificação de condições (caindo para 70,60%), e a técnica ToT manteve o desempenho praticamente estagnado em 75,83%. Isso sugere que o modelo possui uma ótimo capacidade para seguir instruções simples, mas não necessariamente expande seu raciocínio estrutural com *prompts* complexos.

5.1.2 ChatGPT

Diferente do Gemini, o GPT demonstrou ser altamente responsivo à estratégia de EP utilizada. Na abordagem IO, seu desempenho na detecção de condições foi de apenas 60%. No entanto, ele foi o modelo que apresentou o maior salto de qualidade ao ser submetido a estratégias de raciocínio.

Com a aplicação das técnicas CoT e ToT, o GPT elevou sua completude de condições para 81,11%, tornando-se o modelo com o maior teto de desempenho estrutural do estudo. Nos demais atributos(Ator, Ação e Objeto), o modelo manteve consistência acima de 91% em todas as técnicas, provando ser a ferramenta mais adaptável para refinamento via *prompt*.

5.1.3 DeepSeek

O modelo DeepSeek apresentou um comportamento não tão consistente. Ele se destacou na identificação do atributo Ator, sendo o único modelo a atingir 100% de presença neste quesito quando utilizada a técnica ToT. Isso indica uma facilidade particular do modelo em identificar as entidades responsáveis pelas ações do sistema.

No entanto, assim como o GPT, o DeepSeek sofreu na especificação de regras de negócio com prompts simples (IO), registrando apenas 60% no atributo Condição. Ademais, a introdução de técnicas avançadas trouxe melhorias, com o CoT elevando a métrica para 68,89% e o ToT alcançando 74,40%. Embora tenha havido evolução, o DeepSeek permaneceu com médias inferiores ao GPT nas estratégias avançadas.

5.1.4 Variação Semântica entre Modelos

Embora as técnicas avançadas de EP tendam a elevar a qualidade geral, a análise qualitativa revelou que a escolha do LLM permanece determinante. Para ilustrar esse fenômeno, o Quadro 7 compara o requisito de "Cadastro de Aluno" no domínio *Financinhas*, gerado por DeepSeek e GPT, ambos utilizando a técnica CoT.

Observa-se que, mesmo com o CoT instruindo o passo-a-passo, o modelo DeepSeek gerou um requisito genérico, omitindo quais dados seriam necessários, resultando em uma nota de Completude mediana (3). Porém, ao aplicarmos a mesma técnica ao GPT, houve uma evolução no requisito gerado incluindo uma regra de negócio crucial para o domínio educacional infantil: o consentimento do responsável. Isso evidencia que a técnica de EP, por si só, não é capaz de mitigar totalmente as lacunas de conhecimento ou capacidade de inferência de determinados modelos.

Quadro 7 – Comparativo de Completude Semântica (Domínio Financinhas - Técnica CoT)

Modelo	**Requisito Gerado**	**Nota Completude**
DeepSeek	"O sistema deve permitir que um Aluno realize seu cadastro na plataforma."	3
GPT	"O sistema deve permitir o cadastro de alunos com dados básicos (nome, e-mail, turma/grupo opcional, data de nascimento, consentimento de responsável quando necessário)."	5

Fonte: Elaborado pelo autor.

5.1.5 Variação da Completude Estrutural entre Domínios

Além da análise realizada por modelo, também foi realizada uma análise por domínio. Os resultados evidecideram que regras de negócio específicas, como os domínios *Cardio Surgery Illustrar* (CSI) e Abrace, impuseram dificuldades aos modelos com uso da estratégia IO, que só foram mitigadas com técnicas avançadas.

No domínio Abrace, apesar de ser um sistema de gestão, quando aplicado a técnica IO ao modelos Gemini e DeepSeek, as médias obtidas no aspecto completude foram de 65% e 40%, respectivamente. Isso indica que, sem orientação direta no *prompt*, os modelos falham em capturar regras sensíveis e específicas. Já a aplicação da técnica ToT foi capaz a média do DeepSeek neste domínio para 75%, enquanto a técnica CoT levou o GPT a uma média de 95%, demonstrando a eficácia do raciocínio estruturado para regras de negócio críticas.

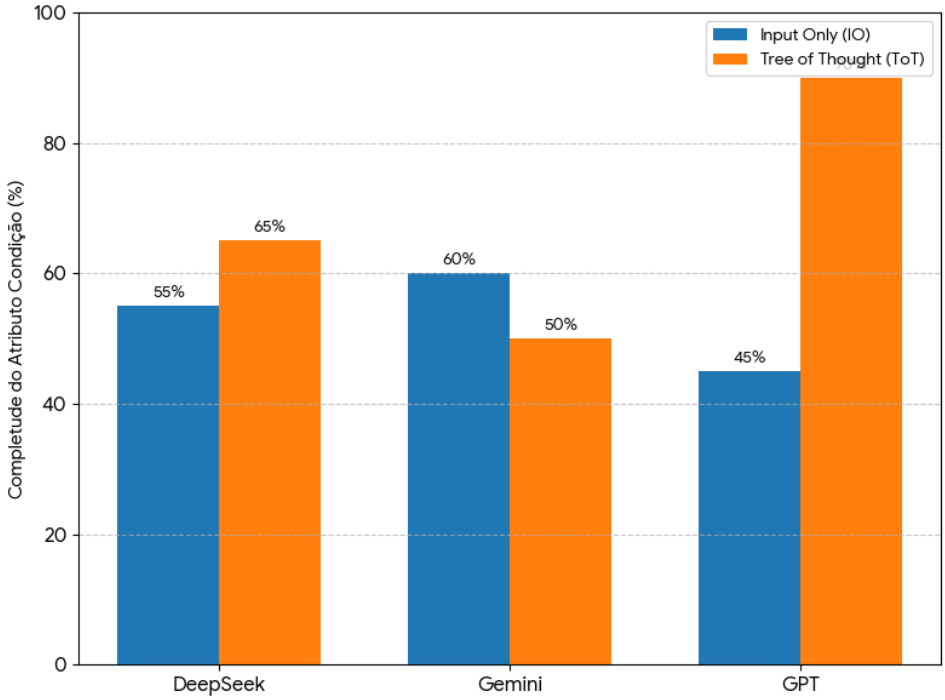
Os domínios Financinhas e CSI confirmaram ser os mais desafiadores para a especificação de regras de negócio. No domínio CSI, ao analisar o atributo Condição na estratégia IO, o modelo DeepSeek obteve uma média de apenas 55%. Com o uso da técnica ToT, essa média subiu para 65%. A figura 5 evidencia que o modelo GPT com ToT atigiu alta consistência ao atingir 90% de presença no atributo Condição, se mostrando um modelo compentente para o domínio da saúde. Além disso, conforme elucidado na figura 5, o Gemini teve um desempenho superior ao utilizar a técnica IO, mostrando que nem algumas alguns modelos em determinados domínios, podem desempenhar melhor ao não utilizarem técnicas avançadas de EP.

Já no domínio Financinhas, a análise do atributo Condição demonstrou a eficácia das estratégias de raciocínio estruturado para a especificação de regras de negócio em contextos gamificados. Na estratégia IO, o modelo DeepSeek apresentou uma média de completude de 70%. Com a aplicação da técnica ToT, essa média elevou-se para 80%, indicando que a decomposição do problema em uma árvore de pensamentos permitiu ao modelo aprofundar a compreensão das mecânicas do sistema, resultando em uma melhor cobertura das condições e restrições necessárias para o projeto.

5.1.5.1 Impacto do Domínio na Qualidade Intrínseca

Sob a ótica qualitativa, o domínio FakeNet (Cibersegurança) destacou-se pela consistência. Mesmo sendo altamente técnico, o uso de CoT permitiu ao DeepSeek atingir 100% de completude estrutural e notas máximas de qualidade. Isso sugere que domínios técnicos reduzem

Figura 5 – Comparativo aspecto Condição no Domínio da Saúde (CSI)



Fonte: Elaborado pelo autor.

a ambiguidade para o modelo. Outro aspecto que contribuiu para ótimo resultando foram o requisitos utilizados para construir o *prompt*, visto que, os mesmo possuiam detalhes técnicos e regras de negócio do sistema.

Em contrapartida, domínios que exigem abstração, como o Financinhas, apresentaram maior oscilação na qualidade semântica durante a estratégia IO, exigindo o passo-a-passo do CoT para garantir que a lógica do jogo fosse traduzida corretamente em requisitos funcionais claros.

5.2 QP_2: Como as técnicas de engenharia de *prompt* podem otimizar as respostas geradas pelas LLMs?

As técnicas de EP demonstraram impacto significativo na qualidade semântica dos requisitos gerados. A análise qualitativa revelou que as estratégias CoT e ToT não apenas melhoram a completude estrutural, mas também elevam substancialmente as médias de Clareza, Completude, Consistência e Relevância. O GPT, quando combinado com CoT, atingiu as maiores médias absolutas do estudo: 4,93 em Consistência, 4,90 em Clareza e 4,90 em Relevância. Essa evolução evidencia que o raciocínio explicitado passo a passo reduz ambiguidades e garante maior coerência interna nos requisitos.

Além disso, a comparação entre os modelos indicou que a escolha da técnica deve considerar as características semânticas desejadas para o contexto. Enquanto o IO oferece uma base funcional, as técnicas avançadas atuam como mecanismos de refinamento e desambiguação.

5.2.1 Resultados da estratégia Input Only

A técnica IO revelou a capacidade intrínseca dos modelos, servindo como *baseline* para a avaliação de otimização. Nesta estratégia, o GPT liderou nos aspectos de Clareza e Consistência, com 4,70 e 4,77, respectivamente. Já o o Gemini destacou-se em Relevância (4,83) e Completude (4,59), evidenciando maior facilidade em contextualizar informações com *prompts* simples.

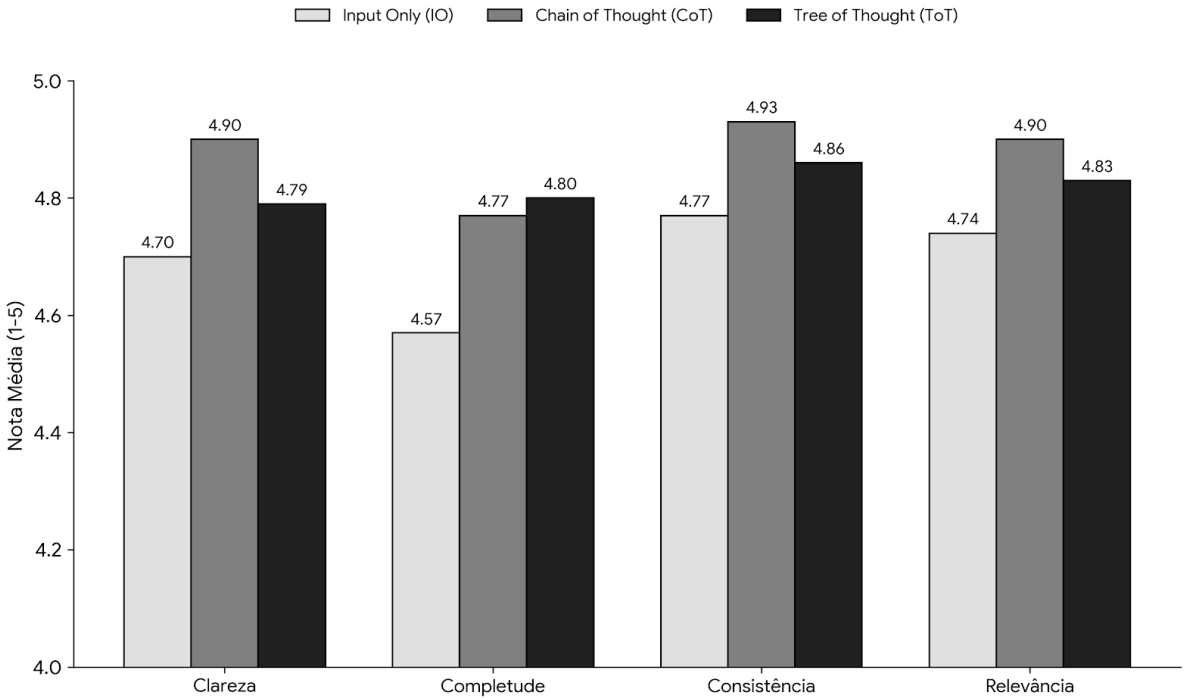
Entretando, as notas não atingiram notas próximas a 5 em nenhum critério. Dessa forma, os requisitos gerados tendem a ser funcionalmente corretos, mas genéricos, limitando-se a declarações simples de funcionalidade. Isso expõe as limitações da falta de orientação estruturada e confirma que a aplicação de técnicas avançadas de EP é indispensável para maximizar a qualidade dos aspectos qualitativos.

5.2.2 Resultados das estratégias Avançadas: Chain of Thought

A aplicação da técnica CoT atuou como o principal mecanismo de otimização para o modelo GPT. Ao aplicarmos a estratégia de EP, o GPT foi capaz de elevar tanto o aspecto de Consistência como o aspecto de Clareza, com médias de 4,93 e 4,90, respectivamente. A Figura 6 ilustra essa evolução clara em relação ao *baseline*.

Para ilustrar a evolução da aplicação da técnica, o Quadro 8 apresenta uma comparação direta de requisitos gerados para o mesmo objetivo no domínio CSI. Observa-se que, na estratégia IO, o requisito gerado é funcionalmente correto, mas genérico, limitando-se a declarar que o usuário deve visualizar o histórico. Já na estratégia CoT, o requisito torna-se mais robusto ao especificar as condições de visualização (gráficos por período diário, semanal, mensal) e os tipos de dados esperados (pressão arterial, frequência cardíaca). Essa diferença estrutural confirma os dados quantitativos, onde o uso de CoT elevou a presença do atributo Condição.

Figura 6 – Evolução da qualidade intrínseca do GPT por técnica



Fonte: Criado pelo autor.

Quadro 8 – Comparativo de Otimização: Input Only vs. Chain of Thought (GPT - CSI)

Estratégia	**Requisito Gerado**
Input Only (IO)	"O sistema deve permitir que o usuário visualize o histórico de suas avaliações, triagens e recomendações previamente geradas no aplicativo."
Chain of Thought (CoT)	"O sistema deve armazenar e exibir um histórico temporal das medições e eventos relacionados à saúde cardiovascular (ex.: pressão arterial, frequência cardíaca, sintomas reportados). Deve permitir visualização de gráficos por período (diário, semanal, mensal), filtrar por tipo de dado e gerar relatórios resumidos para o paciente."

Fonte: Elaborado pelo autor.

5.2.3 Resultados das estratégias Avançadas: Tree of Thought

A técnica ToT demonstrou eficácia na exploração de cenários, nivelando o desempenho dos modelos em aspectos críticos. Com o uso desta técnica, o Gemini obteve sua melhor nota de Clareza com valor 4,8. Além disso, também empatou com o GPT em Consistência e Relevância (4,80).

Esses resultados indicam que a abordagem de árvore de pensamentos otimiza as respostas ao permitir que o modelo avalie múltiplos caminhos lógicos antes da geração final, sendo particularmente útil para mitigar alucinações em modelos que apresentaram instabilidade na estratégia IO.

5.3 QP_3: Qual combinação de técnica de engenharia de *prompt* e LLM produz requisitos de software com maior qualidade?

Baseando-se na média global dos domínios, a combinação GPT + CoT produziu os requisitos com a maior qualidade geral, destacando-se pela excelência semântica, com notas acima de 4,90 nos critérios de Clareza, Consistência e Relevância. Na análise quantitativa, essa combinação alcançou 81,11% de presença do atributo Condição, empatando com GPT+ToT. Porém, foi na análise qualitativa que a técnica CoT se sobressaiu, atingindo as maiores médias absolutas do estudo em 3 dos 4 critérios avaliados.

No entanto, para cenários que exigem a exploração de múltiplos caminhos de regras de negócio, a combinação GPT + ToT mostrou ter resultados parelhos à técnica CoT. Em domínios com regras de negócio mais complexas ou com requisitos que envolvem condições interdependentes, o ToT pode ser uma alternativa viável. Apesar disso, para a maioria dos contextos práticos de ER, a combinação GPT + CoT é recomendada como estratégia padrão devido à sua maior estabilidade e consistência entre diferentes domínios.

5.3.1 Análise de Correlação: Atributo Ator

A análise do atributo Ator revelou um comportamento de saturação por parte dos modelos. Os dados indicam que, na grande maioria dos cenários, a presença do Ator oscilou entre 90% e 100%, independentemente da técnica de EP utilizada.

Consequentemente, não foi observada uma correlação linear significativa entre a presença do Ator e a variação na nota de Completude. Isso sugere que a identificação do "quem executa a ação"é uma competência basal já dominada pelos LLMs. Embora a presença do Ator seja um pré-requisito para a completude, sua alta frequência constante impede que ele atue como um elemento diferenciador para elevar a percepção de qualidade além do patamar básico.

5.3.2 Análise de Correlação: Atributo Ação

A análise estatística do atributo Ação demonstrou um padrão de constância ainda mais acentuado. Em praticamente todas as combinações de modelos e técnicas, a taxa de presença deste atributo atingiu 100%. Devido a essa ausência de variância nos dados estruturais, o cálculo de correlação torna-se estatisticamente irrelevante para explicar as flutuações na nota de Completude.

Este resultado evidencia que os modelos de linguagem, pela sua própria natureza instrucional, sempre estruturam o requisito em torno de um verbo ou comando. Portanto, a simples presença da Ação não garante uma nota alta de Completude; a variação na qualidade qualitativa (de 3,65 a 5,00) depende de outros fatores semânticos e não da mera existência do verbo de ação.

5.3.3 Análise de Correlação: Atributo Objeto

De forma análoga ao atributo Ação, o atributo Objeto apresentou 100% de presença em todas as amostras analisadas. A estabilidade absoluta desta métrica indica que os LLMs não falham em identificar o alvo da ação (o objeto direto ou indireto) durante a geração de requisitos.

Assim, conclui-se que não há correlação estatística direta que explique a variância da Completude através deste atributo isolado. A presença do Objeto é uma condição necessária, porém insuficiente, para atingir a excelência na qualidade. A percepção humana de um requisito "completo"exige que, além do objeto estar presente, ele esteja contextualizado por regras de negócio (Condições), confirmando a hipótese de que a estrutura básica (Ator, Ação, Objeto) é facilmente atendida pelas técnicas simples (IO), enquanto a sofisticação da Completude depende das variáveis mais complexas.

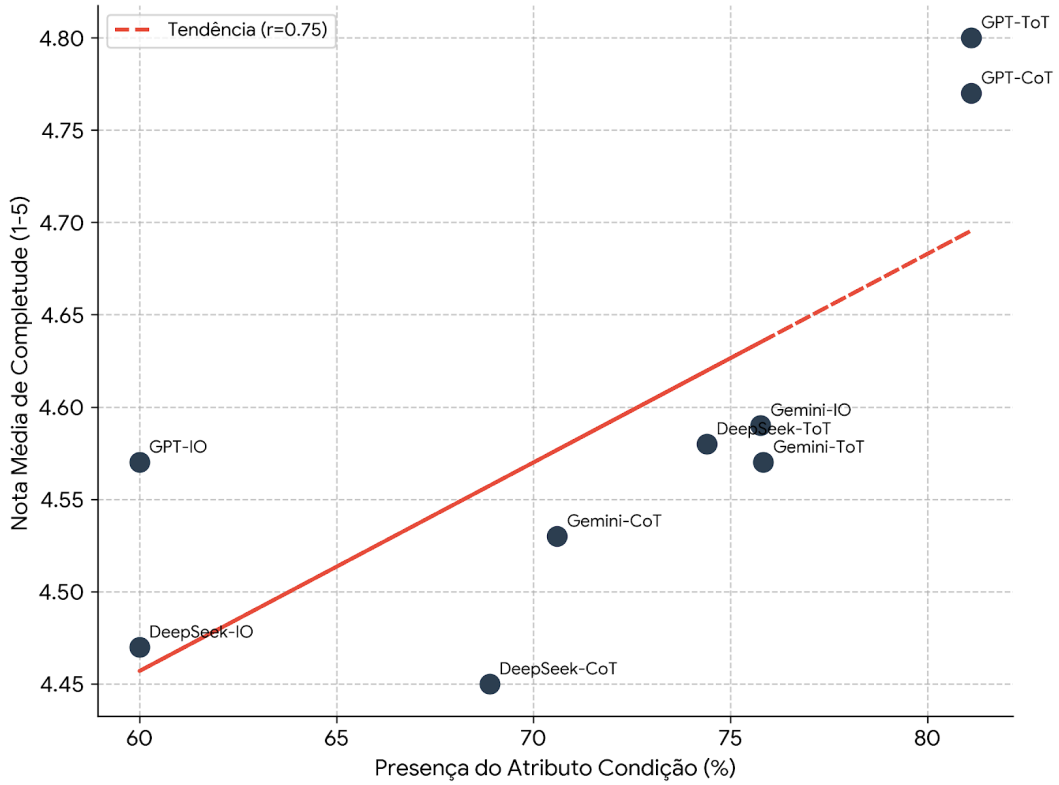
5.3.4 Análise de Correlação: Atributo Condição

Ao contrário dos atributos Ator, Ação e Objeto, que apresentaram um comportamento de saturação com taxas de presença próximas a 100% em quase todos os cenários, o atributo Condição demonstrou uma variabilidade significativa entre as diferentes combinações de modelos e estratégias. Essa oscilação estatística — variando de patamares inferiores a 30% em alguns casos da técnica IO até superiores a 90% nas técnicas ToT — permitiu isolar esta variável para verificar seu impacto direto na percepção humana de qualidade.

Para mensurar essa relação, foi realizado o cruzamento entre o percentual de presença de Condições e a nota média de Completude atribuída na avaliação qualitativa. O Gráfico 7 ilustra a dispersão dessas duas variáveis.

A análise estatística revelou um Coeficiente de Correlação de Pearson (r) de 0,74, indicando uma correlação positiva forte. Isso significa que há uma tendência clara, onde à medida que o percentual de presença de condições nos requisitos aumenta, a nota de completude atribuída também tende a subir.

Figura 7 – Correlação entre Presença de Condição e Nota de Completude



Fonte: Elaborado pelo autor.

Dessa forma, ao observar a técnica IO aplicada tanto ao modelo GPT quanto ao DeepSeek, onde em ambos os casos o atributo condição está presente em apenas 60% dos requisitos, nota-se que a nota de completude foi inferior a 4,60. Por outro lado, ao aplicar técnicas como CoT e ToT ao GPT, foi possível visualizar que, com o atributo Condição superior a 80%, houve um aumento direto na completude, alcançando notas superiores a 4,75.

Portanto, os dados sugerem que, para um requisito transmitir a percepção de completude a um leitor humano, é de suma importância que as regras de negócio e restrições estejam bem definidas. Requisitos limitados apenas a Ator, Ação e Objeto tendem a ser penalizados qualitativamente. Isso evidencia que as técnicas avançadas de EP foram capazes de amenizar esses problemas, ao estimular os LLMs a raciocinarem e validarem se os resultados finais estavam em conformidade com o esperado.

5.4 QP_4: Em que medida os requisitos gerados pela melhor combinação são aceitos por *stakeholders* em cenários reais?

A validação empírica com *stakeholders* revelou que a aceitação plena imediata dos requisitos gerados pela melhor combinação (GPT + CoT) é minoritária, ocorrendo em apenas 33% dos casos. No entanto, boa parte dos requisitos poderam ser aproveitados, em que 56% deles foram aceitos com restrições 89% dos artefatos gerados podem ser cosiderados são úteis como base para a engenharia de requisitos.

Apesar da alta utilidade, a necessidade de intervenção humana permanece obrigatória. A validação prática demonstrou que os LLMs tendem a falharem sistematicamente na especificidade das regras de negócio, exigindo a revisão de um especialista para preencher lacunas de contexto que o modelo ainda não é capaz de inferir autonomamente.

5.4.1 Taxa de Aceitação e Rejeição por Domínio

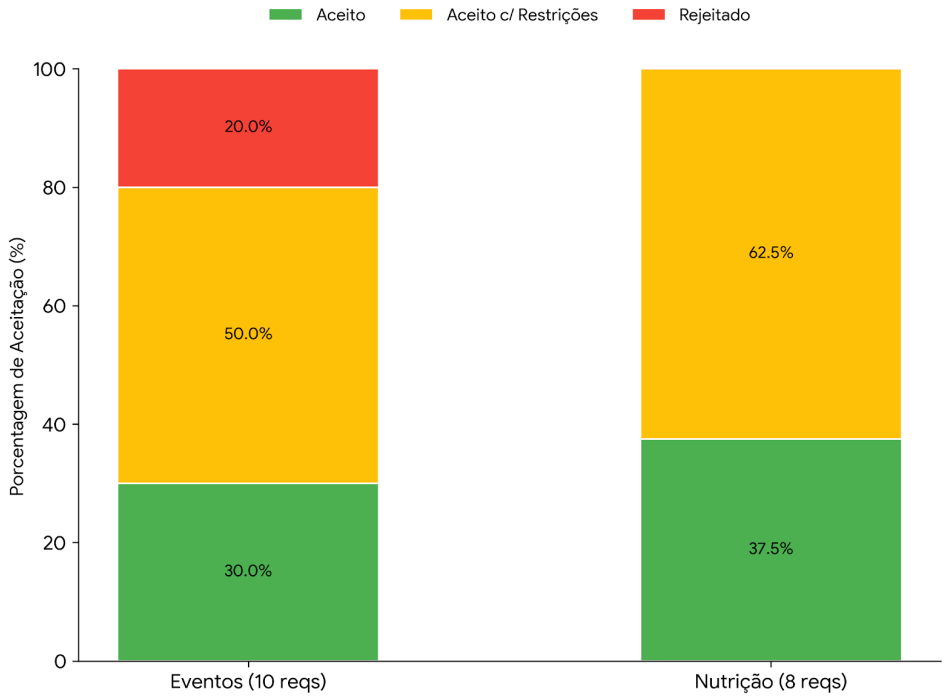
A análise dos dados revelou que a aceitação plena do requisito, isto é, sem nenhuma necessidade de mudança, ainda é minoritária, estando presente em apenas 33% dos casos. A maior parte dos requisitos se enquadrou na categoria de aceito com restrições, com cerca de 56%. Esse dado indica que os LLMs são capazes de gerar requisitos que estão de acordo com o domínio, porém falham em detalhes e especificações mais precisas.

Além disso, a Figura 8 evidencia uma distinção clara entre os domínios. No domínio de Eventos para agricultores, houve 20% de requisitos rejeitados por defeitos de Incorreção Factual, enquanto no domínio de Nutrição não houve casos de rejeição. Esse dado revela que os LLMs ainda possuem dificuldade em entender as nuances de domínios mais específicos. Um exemplo claro é o RF-05, que afirma que o sistema deve monitorar o número de participantes presentes no evento em tempo real. O motivo da rejeição desse requisito se deu devido à falta de estrutura de eventos de agricultura que, em geral, não fazem o descredenciamento dos participantes, inviabilizando, portanto, o monitoramento em tempo real de quantas pessoas estão presentes naquele momento no evento.

5.4.2 Análise Qualitativa dos Defeitos

A classificação detalhada das falhas apontadas pelos *stakeholders* evidenciou que a Omissão, caracterizada pela ausência de regras de negócio, condições de contorno ou informações

Figura 8 – Comparativo da Taxa de Aceitação por Domínio (Eventos vs. Nutrição)



Fonte: Elaborado pelo autor.

essenciais para a implementação, foi o defeito predominante. Conforme ilustrado na Figura 9, as omissões representaram 50% do total de falhas identificadas nos dois domínios analisados, seguidas pela Ambiguidade (25%) e Incorreção Factual (16,7%).

Esses dados sugerem que, embora a aplicação de técnicas avançadas de EP como CoT melhore a estrutura sintática dos requisitos, os modelos ainda enfrentam dificuldades em inferir a totalidade das regras de negócio implícitas em cenários de uso específicos sem o refinamento humano.

Para exemplificar a natureza dessas omissões, destacam-se dois casos onde a generalização do modelo falhou em capturar a especificidade exigida pelo processo real:

– **RF-09 (Domínio Eventos) - Falha por Omissão Técnica**

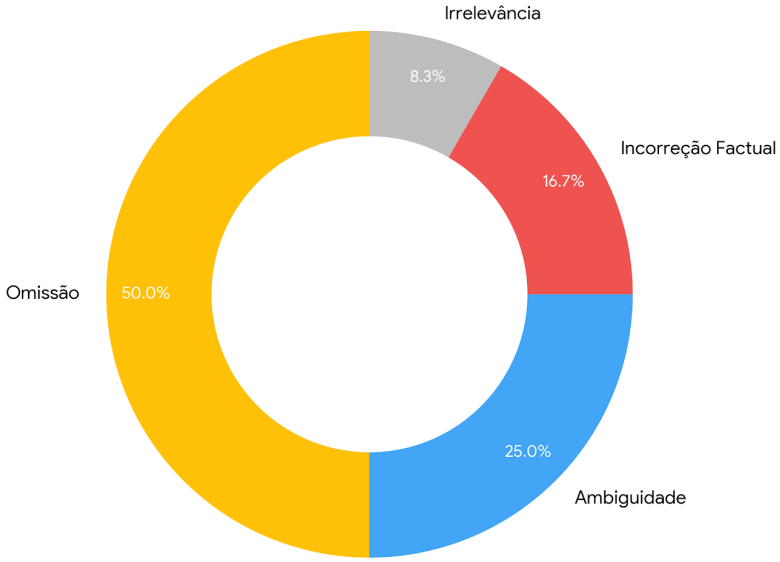
  Requisito Gerado: "O sistema deve permitir que participantes atualizem seus dados cadastrais, garantindo que informações como telefone, e-mail e local de origem estejam sempre atualizadas."

  Crítica do Stakeholder: "O requisito permite a atualização, mas não é claro quanto a validação dos dados e formatação dos mesmos."

– **RF-04 (Domínio Nutrição) - Falha por Omissão de Regra de Negócio**

  Requisito Gerado: "O sistema deve disponibilizar ao paciente a visualização do plano

Figura 9 – Distribuição dos Tipos de Defeitos Identificados na Validação



Fonte: Elaborado pelo autor.

alimentar prescrito, incluindo refeições, horários, opções de substituição e orientações nutricionais definidas pelo nutricionista."

Crítica do Stakeholder: "Embora cite opções de substituição, o requisito falha ao não especificar a porção exata de cada alimento."

Em ambos os casos, a falha reside na ausência de detalhamento das regras de validação e granularidade da informação. No RF-09, o requisito omite a necessidade de mecanismos de validação dos dados inseridos (ex: formato de e-mail, unicidade de cadastro), abrindo margem para inconsistências no banco de dados. Já no RF-04 o requisito falha em especificar atributos cruciais para o contexto nutricional, como a porção exata de cada alimento, dado essencial para que o paciente realize substituições corretas sem comprometer o plano dietético.

5.5 Discussão dos Resultados e Implicações Práticas

Portanto, os resultados desse estudo evideciam achados relevantes na literatura de ER, aplicado especialmente ao contexto de LLMs. Além disso, o presente estudo tem descobertas que também poderam ser encontrados e evidenciados em outros estudos, como Vogelsang (2024) ao também destacar a dificuldade dos LLMs em compreender o “mundo fechado” e as nuances específicas de um domínio sem um contexto exaustivo. Outro achado foram as regras de negócio do requisitos por vez muito vazias e simples, que também pode ser eviciado por Mahbub *et al.* (2024) ao detectar frequentemente omissões e ambiguidades, em requisitos gerados pelo

GPT-4. Ademais, os ótimos resultados da técnica CoT também já haviam sido observados por Masoudifard *et al.* (2024), demostrando que ao explicitar o racíocinio para o modelo e decompor o problema, houve uma evolução nas respostas geradas, deixando-as menos superficiais.

Sendo assim, os uso dos LLMs no contexto da indústria de ER pode ser válido e plenamente viável. Conforme a validação com *stakeholders*, quase 90% dos requisitos gerados poderam de algum maneira serem aproveitados, o que pode auxiliar na etapa inicial de geração de requisitos. Além disso, é possível notar que há um variação de desempenho ao variarmos de um domínio para outro. Assim, é necessario que haja maior supervisão humana, quando o domínio em questão é crítico, e pode gerar risco físicos ou financeiros.

5.6 Ameaças à Validade

Nesta seção, são discutidas as limitações identificadas durante a execução do estudo, bem como as estratégias adotadas para minimizar seus impactos nos resultados finais.

5.6.1 Identificação das Ameaças

A respeito davalidade interna, identificou-se que a subjetividade inerente ao processo de avaliação humana representa um risco, uma vez que tanto a análise qualitativa quanto a quantitativa são reféns da interpretação do avaliador no momento da verificação. Além disso, a qualidade da construção dos *prompts* pode ter introduzido vieses, visto que o nível de detalhes fornecido pode ter favorecido involuntariamente uma técnica em relação a outra. Outra ameaça relevante à validade interna foi a decisão metodológica de restringir a análise aos 10 primeiros requisitos gerados. Essa escolha configura um viés de amostragem, pois não se pode garantir que os requisitos subsequentes não apresentariam qualidade superior, ignorando o potencial dos modelos em janelas de contexto mais longas.

Pensando na validade externa, os resultados encontram-se limitados ao escopo dos modelos e domínios selecionados. Como a análise se concentrou em cinco domínios específicos, não se é possível afirmar que estes representam a totalidade dos problemas da ES, deixando de fora áreas críticas como sistemas embarcados ou de tempo real, que poderiam expor comportamentos distintos dos LLMs. Ademais, o perfil dos *stakeholders* participantes da validação constitui um fator limitante, pois, sendo estudantes de graduação, podem não possuir a mesma visão crítica e experiência de mercado que especialistas sêniores teriam para identificar

falhas sutis de negócio.

Por fim, quanto à validade de construto, as métricas utilizadas para avaliar a qualidade dos requisitos foram capazes de identificar elementos importantes, porém não capturam a qualidade como um todo. Conforme evidenciado por Sommerville (2011), a presença de atributos estruturais bem definidos garante que o requisito está formalmente correto, porém não assegura que ele seja o requisito adequado para resolver o problema real do cliente. Assim, modelos podem ter gerado requisitos úteis que foram penalizados simplesmente por divergirem da estrutura esperada.

5.6.2 Estratégias de Mitigação

Para contornar as ameaças à validade interna, foram adotados protocolos rigorosos de revisão. O risco da subjetividade foi mitigado através da realização de múltiplas rodadas de análise para os domínios, com o objetivo de eliminar o viés cognitivo de uma avaliação única. No que diz respeito aos *prompts*, buscou-se a padronização dos contextos e requisitos de entrada, variando apenas as instruções das técnicas, para garantir isonomia. Já sobre o recorte dos 10 primeiros requisitos, essa estratégia justifica-se pela necessidade de padronizar o tamanho da amostra entre modelos com verbosidades diferentes e, sobretudo, para simular o comportamento real do usuário, que tende a priorizar os primeiros resultados apresentados pela ferramenta de IA.

Visando fortalecer a validade externa, a seleção dos domínios buscou diversidade, cobrindo áreas distintas como Saúde, Educação e Gestão, para abranger diferentes tipos de regras de negócio. Da mesma maneira, para mitigar a inexperiência dos *stakeholders*, foram selecionados participantes que já estivessem em fase de conclusão de curso e com experiência comprovada de estágio na área, garantindo assim um nível adequado de conhecimento prático e teórico sobre os domínios avaliados.

Quanto à validade de construto, a mitigação baseou-se na adoção de uma abordagem híbrida de avaliação. Não se dependeu apenas da verificação estrutural automática, mas integrou-se uma avaliação qualitativa detalhada e a validação prática com humanos. Essa triangulação de métodos permitiu identificar falhas de negócio e nuances semânticas que a análise puramente estrutural não revelaria, assegurando que a qualidade fosse mensurada não apenas pela forma, mas também pela utilidade do artefato gerado.

6 CONCLUSÕES E TRABALHOS FUTUROS

Este estudo investigou empiricamente como diferentes estratégias de EP impactam a qualidade dos requisitos de software gerados por LLMs. A pesquisa adotou duas abordagens metodológicas complementares para avaliação dos artefatos gerados. Na primeira abordagem, documentos de especificação de requisitos de cinco domínios distintos foram utilizados como material-fonte para a construção dos *prompts*, permitindo avaliar a capacidade dos modelos de manter coerência com padrões estabelecidos. Na segunda abordagem, foram conduzidas entrevistas com *stakeholders* reais de dois domínios, cujo conteúdo foi utilizado como base para a elaboração dos *prompts*, permitindo validação prática dos requisitos gerados em cenários autênticos de uso.

A análise quantitativa evidenciou que os modelos ainda apresentam deficiências estruturais, especialmente na especificação de condições e regras de negócio dos requisitos, com o atributo Condição registrando taxas de presença inferiores a 60% quando aplicada a estratégia IO. Ademais, em domínios específicos como Financinhas e CSI, a completude semântica apresentou desempenho insatisfatório, com requisitos frequentemente omitindo informações cruciais para sua implementação correta.

A validação com *stakeholders* reforçou a necessidade de intervenção humana especializada para refinamento dos requisitos gerados, visto que apenas um terço dos artefatos foram aceitos integralmente, enquanto mais da metade demandaram ajustes antes de serem considerados prontos para implementação. A análise qualitativa dos defeitos identificados pelos especialistas revelou que o tipo de falha mais recorrente foi a Omissão, em metade dos casos, caracterizada pela ausência de regras de negócio, condições especificações essenciais para o requisito.

Portanto, apesar das estratégias avançadas de EP terem demostrado a capacidade de aprimorar a qualidade geral nos requisitos, os resultados indicam dificuldade dos LLMs de gerarem artefatos de completude estrutural e semântica, evidenciando a necessidade de passarem por validações e refinamentos conduzidos por engenheiros de requisitos e *stakeholders* do domínio.

6.1 Principais Contribuições

- Avaliação empírica de três estratégias de EP (IO, CoT e ToT) aplicadas a três LLMs distintos na geração de requisitos de software.
- Metodologia híbrida combinando análise quantitativa (presença de Ator, Ação, Objeto e Condição) e qualitativa (Clareza, Completude, Consistência e Relevância) para avaliação da qualidade dos requisitos.
- Identificação de que o atributo *Condição* representa o principal desafio para LLMs evidenciando dificuldade na especificação de regras de negócio.
- Demonstração de forte correlação entre completude estrutural e percepção humana de qualidade, comprovando que requisitos com condições bem definidas são avaliados como mais completos.
- Validação prática com *stakeholders* revelando que apenas um terço dos requisitos são plenamente aceitos, mais da metade requerem ajustes e metade das falhas são por Omissão de regras de negócio.
- Comprovação de que a combinação GPT + CoT produz requisitos com maior qualidade geral, atingindo notas próximas a máxima nos aspectos de Clareza, Consistência e Relevância.

6.2 Trabalhos Futuros

- Avaliação de modelos adicionais, incluindo Claude (Anthropic), Llama (Meta) e outros LLMs especializados, para verificar se os padrões identificados se mantêm em diferentes arquiteturas.
- Exploração de técnicas híbridas de EP, combinando estratégias como CoT com Retrieval-Augmented Generation (RAG) para melhorar a especificação de regras de negócio complexas.
- Desenvolvimento de métricas automatizadas para avaliação de qualidade semântica dos requisitos, reduzindo a necessidade de revisão manual e permitindo feedback iterativo.
- Investigação de como LLMs podem ser integrados a processos consolidados de ER, como entrevistas estruturadas, atuando como ferramentas de apoio ao engenheiro de requisitos.
- Proposta de *frameworks* e templates de EP específicos para ER, baseados nos achados deste trabalho, otimizados para diferentes tipos de requisitos e domínios de aplicação.

REFERÊNCIAS

ARAWJO, I.; VAITHILINGAM, P.; WATTENBERG, M.; GLASSMAN, E. Chainforge: An open-source visual programming environment for prompt engineering. In: ANNUAL ACM SYMPOSIUM ON USER INTERFACE SOFTWARE AND TECHNOLOGY, 36. **Proceedings [...]**. New York, NY, USA: Association for Computing Machinery, 2023. (UIST '23 Adjunct). ISBN 9798400700965. Disponível em: https://doi.org/10.1145/3586182.3616660. Acesso em: 04 maio 2025.

CHEN, Z.; WANG, C.; SUN, W.; YANG, G.; LIU, X.; ZHANG, J.; LIU, Y. Promptware engineering: Software engineering for llm prompt development. **arXiv e-prints**, 03 2025.

DAUN, M.; GRUBB, A. M.; STENKOVA, V.; TENBERGEN, B. A systematic literature review of requirements engineering education. **Requirements Engineering**, v. 28, n. 2, p. 145–175, 2023. ISSN 1432-010X. Disponível em: https://doi.org/10.1007/s00766-022-00381-9. Acesso em: 15 ago. 2025.

DEEPSEEK-AI. **DeepSeek-R1**: Incentivizing reasoning capability in llms via reinforcement learning. 2025. Disponível em: https://arxiv.org/abs/2501.12948. Acesso em: 15 abr. 2025.

FERRARI, A.; SPOLETINI, P. Formal requirements engineering and large language models: A two-way roadmap. **Inf. Softw. Technol.**, Butterworth-Heinemann, USA, v. 181, n. C, abr. 2025. ISSN 0950-5849. Disponível em: https://doi.org/10.1016/j.infsof.2025.107697. Acesso em: 10 ago. 2025.

GHEORGHE, S.-D. Designing internet of things systems for circular economy and digital product passports – a requirements engineering perspective. **ResearchGate**, Jul 2025. Full-text available.

GOOGLE. **Gemini 3 Pro Model Card**. [*S. l.*], 2025. Disponível em: https://deepmind.google/gemini/gemini-3-pro-model-card.pdf. Acesso em: 07 jan. 2026.

HENRICKSON, L.; MEROÑO-PEÑUELA, A. Prompting meaning: a hermeneutic approach to optimising prompt engineering with chatgpt. **AI Soc.**, v. 40, n. 2, p. 903–918, 2025. Disponível em: https://doi.org/10.1007/s00146-023-01752-8. Acesso em: 20 ago. 2025.

HOFMANN, H.; LEHNER, F. Requirements engineering as a success factor in software projects. **IEEE Software**, v. 18, n. 4, p. 58–66, 2001.

HUSSAIN, M.; REHMAN, U. U.; NGUYEN, T. D.; LEE, S. Cot-sts: A zero shot chain-of-thought prompting for semantic textual similarity. In: ARTIFICIAL INTELLIGENCE AND CLOUD COMPUTING CONFERENCE (AICCC), 6., 2023. **Proceedings[...]**. New York, NY, USA: Association for Computing Machinery, 2024. (AICCC '23), p. 135–139. ISBN 9798400716225. Disponível em: https://doi.org/10.1145/3639592.3639611.

KOYUNCU, A. Exploring fine-grained bug report categorization with large language models and prompt engineering: An empirical study. **ACM Trans. Softw. Eng. Methodol.**, Association for Computing Machinery, New York, NY, USA, maio 2025. ISSN 1049-331X. Disponível em: https://doi.org/10.1145/3736408. Acesso em: 20 jun. 2025.

LONG, J. **Large Language Model Guided Tree-of-Thought**. 2023. Disponível em: https://arxiv.org/abs/2305.08291. Acesso em: 01 abr. 2025.

MA, Q.; PENG, W.; YANG, C.; SHEN, H.; KOEDINGER, K.; WU, T. What should we engineer in prompts? training humans in requirement-driven llm use. **ACM Trans. Comput.-Hum. Interact.**, Association for Computing Machinery, New York, NY, USA, abr. 2025. ISSN 1073-0516. Just Accepted. Disponível em: https://doi.org/10.1145/3731756. Acesso em: 10 mar. 2025.

MAHBUB, T.; DGHAYM, D.; SHANKARNARAYANAN, A.; SYED, T.; SHAPSOUGH, S.; ZUALKERNAN, I. Can gpt-4 aid in detecting ambiguities, inconsistencies, and incompleteness in requirements analysis? a comprehensive case study. **IEEE Access**, v. 12, p. 171972–171992, 2024.

MAO, Y.; HE, J.; CHEN, C. From prompts to templates: A systematic prompt template analysis for real-world llmapps. **CoRR**, abs/2504.02052, 2025. Disponível em: https://doi.org/10.48550/arXiv.2504.02052. Acesso em: 15 ago. 2025.

MASOUDIFARD, A.; SOROND, M. M.; MADADI, M.; SABOKROU, M.; HABIBI, E. Leveraging graph-rag and prompt engineering to enhance llm-based automated requirement traceability and compliance checks. **arXiv e-prints**, p. arXiv–2412, 2024.

OPENAI. **GPT-5 System Card**. 2025. Disponível em: https://cdn.openai.com/gpt-5-system-card.pdf. Acesso em: 15 ago. 2025.

PORTER, A.; VOTTA, L.; BASILI, V. Comparing detection methods for software requirements inspections: a replicated experiment. **IEEE Transactions on Software Engineering**, v. 21, n. 6, p. 563–575, 1995.

RAWSON, J.; REDDIVARI, S. A chatgpt-powered prompt engineering framework for generating software acceptance criteria. In: **Proceedings[...]**. New York, NY, USA: Association for Computing Machinery, 2025. (ACMSE 2025). ISBN 9798400712777. Disponível em: https://doi.org/10.1145/3696673.3723078. Acesso em: 07 mar. 2025.

ROCHA, A. *et al.* Artificial intelligence techniques for requirements engineering: A comprehensive literature review. **Preprints**, MDPI, 2025. Preprint.

RONANKI, K.; BERGER, C.; HORKOFF, J. Investigating chatgpt's potential to assist in requirements elicitation processes. In: EUROMICRO CONFERENCE ON SOFTWARE ENGINEERING AND ADVANCED APPLICATIONS, 49., 2023. **Proceedings [...]**. [*S. l.*], 2023. p. 354–361.

RONANKI, K.; CABRERO-DANIEL, B.; HORKOFF, J.; BERGER, C. **Requirements Engineering using Generative AI**: Prompts and prompting patterns. 2023. Disponível em: https://arxiv.org/abs/2311.03832. Acesso em: 05 mar. 2025.

ROSTAM, Z. R. K.; SZéNáSI, S.; KERTéSZ, G. Achieving peak performance for large language models: A systematic review. **IEEE Access**, v. 12, p. 96017–96050, 2024.

RUKMONO, S. A.; OCHOA, L.; CHAUDRON, M. Deductive software architecture recovery via chain-of-thought prompting. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING: NEW IDEAS AND EMERGING RESULT ICSE-NIER. **Proceedings[...]**. New York, NY, USA: Association for Computing Machinery, 2024. (ICSE-NIER'24), p. 92–96. ISBN 9798400705007. Disponível em: https://doi.org/10.1145/3639476.3639776. Acesso em: 04 mar 2025.

SHEOKAND, T.; JAIN, G.; BAHGA, A.; MADISETTI, V. K. Enhancing llm reasoning capabilities through brokered multi-expert reflection. **IEEE Access**, v. 13, p. 67993–68019, 2025.

SOMMERVILLE, I. **Software Engineering**. 9. ed. Boston: Addison-Wesley, 2011.

UMAR, M. A.; LANO, K.; ABUBAKAR, A. K. Automated requirements engineering framework for agile model-driven development. **Frontiers in Computer Science**, Frontiers, v. 7, 2025. Acesso em: 28 mar. 2025.

VASUDEVAN, P.; REDDIVARI, S. The role of generative ai models in requirements engineering: A systematic literature review. In: PROCEEDINGS OF THE 2025 ACM SOUTHEAST CONFERENCE. **Proceedings[...]**. New York, NY, USA: Association for Computing Machinery, 2025. (ACMSE 2025), p. 188–194. ISBN 9798400712777. Disponível em: https://doi.org/10.1145/3696673.3723053. Acesso em: 03 maio 2025.

VOGELSANG, A. From specifications to prompts: On the future of generative large language models in requirements engineering. **IEEE Software**, v. 41, n. 5, p. 9–13, 2024.

WEI, J.; WANG, X.; SCHUURMANS, D.; BOSMA, M.; ICHTER, B.; XIA, F.; CHI, E.; LE, Q.; ZHOU, D. **Chain-of-Thought Prompting Elicits Reasoning in Large Language Models**. 2023. Disponível em: https://arxiv.org/abs/2201.11903. Acesso em: 03 maio 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**. 2023. Disponível em: https://arxiv.org/abs/2303.07839. Acesso em: 10 mar. 2025.

WIEGERS, K.; BEATTY, J. **Software Requirements**. 3. ed. Redmond, WA: Microsoft Press, 2013. ISBN 978-0735679665.

YANG, J.; JIN, H.; TANG, R.; HAN, X.; FENG, Q.; JIANG, H.; ZHONG, S.; YIN, B.; HU, X. Harnessing the power of llms in practice: A survey on chatgpt and beyond. **ACM**, New York, NY, USA, v. 18, n. 6, abr. 2024. ISSN 1556-4681. Disponível em: https://doi.org/10.1145/3649506. Acesso em: 10 maio 2025.

YAO, S.; YU, D.; ZHAO, J.; SHAFRAN, I.; GRIFFITHS, T. L.; CAO, Y.; NARASIMHAN, K. **Tree of Thoughts**: Deliberate problem solving with large language models. 2023. Disponível em: https://arxiv.org/abs/2305.10601. Acesso em: 03 maio 2025.

YAO, Y.; DUAN, J.; XU, K.; CAI, Y.; SUN, Z.; ZHANG, Y. A survey on large language model (llm) security and privacy: The good, the bad, and the ugly. **High-Confidence Computing**, v. 4, n. 2, p. 100211, 2024. ISSN 2667-2952. Disponível em: https://www.sciencedirect.com/science/article/pii/S266729522400014X. Acesso em: 02 abr. 2025.

ZHAO, W. X.; ZHOU, K.; LI, J.; TANG, T.; WANG, X.; HOU, Y.; MIN, Y.; ZHANG, B.; ZHANG, J.; DONG, Z.; DU, Y.; YANG, C.; CHEN, Y.; CHEN, Z.; JIANG, J.; REN, R.; LI, Y.; TANG, X.; LIU, Z.; LIU, P.; NIE, J.-Y.; WEN, J.-R. **A Survey of Large Language Models**. 2025. Disponível em: https://arxiv.org/abs/2303.18223. Acesso em: 10 jun. 2025.

APÊNDICE A – MÉDIAS DAS ANÁLISES QUANTITATIVAS E QUALITATIVAS POR DOMÍNIO

Figura 10 – Médias da combinação GPT + IO

Médias: IO + GPT

Domínio	# Média Ator	# Média Ação	# Média Objeto	# Média condição	# Média Clareza	# Média completude	# Média Consistência	# Média Relevância
Financinhas - 1	100%	100%	100%	20%	4.2	4.1	4.8	4.5
Financinhas - 2	100%	100%	100%	70%	5	5	4.9	5
Abrace - 1	100%	100%	100%	90%	4.95	4.85	4.9	4.7
Abrace - 1	100%	100%	100%	30%	4.5	4.8	4.5	4.6
FakeNet -1	100%	100%	100%	100%	4.45	4.45	4.7	5
CSI - 1	100%	100%	100%	40%	4.6	4.5	4.9	4.55
CSI - 2 OPEN	100%	90%	100%	50%	4.9	4.4	4.6	4.9
Rapidu - 1	100%	90%	100%	70%	4.9	4.5	4.6	4.65
Rapidu - 1	80%	90%	100%	70%	4.8	4.5	5	4.8
+ New page								
	AVERAGE 97.78%	AVERAGE 96.67%	AVERAGE 100.00%	AVERAGE 60.00%	AVERAGE 4.70	AVERAGE 4.57	AVERAGE 4.77	AVERAGE 4.74

Fonte: Criado pelo autor.

Figura 11 – Médias da combinação GPT + CoT

Médias: COT + GPT

Domínio	# Média Ator	# Média Ação	# Média Objeto	# Média condição	# Média Clareza	# Média completude	# Média Consistência	# Média Relevância
Financinhas - 1	100%	100%	100%	40%	4.8	4.6	4.9	4.7
Financinhas - 2	100%	100%	100%	50%	5	5	5	5
Abrace - 1	100%	100%	100%	90%	5	5	5	4.93
Abrace - 2	100%	100%	100%	100%	4.8	4.8	4.9	5
FakeNet - 1	100%	100%	100%	90%	4.7	4.75	4.9	5
CSI - 1	100%	100%	100%	100%	4.9	4.9	5	4.85
CSI - 2	100%	100%	100%	70%	4.9	4.8	4.7	5
Rapidu - 1	100%	100%	100%	90%	5	4.85	4.95	4.75
Rapidu - 2	20%	100%	100%	100%	5	4.2	5	4.9
+ New page								
	AVERAGE 91.11%	AVERAGE 100.00%	AVERAGE 100.00%	AVERAGE 81.11%	AVERAGE 4.90	AVERAGE 4.77	AVERAGE 4.93	AVERAGE 4.90

Fonte: Criado pelo autor.

Figura 12 – Médias da combinação GPT + ToT

Médias: TOT + GPT

Domínio	# Média Ator	# Média Ação	# Média Objeto	# Média condição	# Média Clareza	# Média completude	# Média Consistência	# Média Relevância
Financinhas -1	100%	100%	100%	50%	5	5	5	4.85
Financinhas - 2	100%	100%	100%	90%	5	5	5	5
Abrace - 1	100%	100%	100%	90%	4.85	5	4.85	4.75
Abrace - 2	100%	100%	100%	70%	4.6	4.5	4.9	4.9
FakeNet - 1	100%	100%	100%	80%	4.6	4.4	4.5	5
CSI - 1	100%	100%	100%	90%	5	5	5	4.95
CSI - 2	100%	100%	100%	90%	4.7	4.9	4.7	5
Rapidu - 1	100%	100%	100%	90%	4.85	4.9	4.9	4.7
Rapidu - 2	30%	100%	100%	80%	4.5	4.5	4.9	4.3
+ New page								
	AVERAGE 92.22%	AVERAGE 100.00%	AVERAGE 100.00%	AVERAGE 81.11%	AVERAGE 4.79	AVERAGE 4.80	AVERAGE 4.86	AVERAGE 4.83

Fonte: Criado pelo autor.

Figura 13 – Médias da combinação Gemini + IO

Médias: IO + Gemini

Domínio	# Média Ator	# Média Ação	# Média Objeto	# Média condição	# Média Clareza	# Média completude	# Média Consistência	# Média Relevância
Financinhas - 1	100%	100%	100%	70%	4.55	4.35	4.4	4.85
Financinhas - 2	100%	100%	100%	100%	5	4.6	5	4.9
Abrace - 1	100%	100%	100%	80%	4.8	4.9	5	4.85
Abrace - 2	100%	100%	100%	50%	4	4.5	4.5	4.5
FakeNet -1	100%	100%	100%	100%	4.2	4.2	4.5	5
CSI - 1	100%	100%	100%	60%	4.85	4.7	4.85	4.6
CSI - 2	100%	100%	100%	60%	4.9	4.7	4.6	5
Rapidu - 1	100%	100%	100%	90%	4.9	4.9	4.9	4.95
Rapidu - 2	70%	100%	100%	70%	4.7	4.5	5	4.8
+ New page								
	AVERAGE 96.67%	AVERAGE 100.00%	AVERAGE 100.00%	AVERAGE 75.56%	AVERAGE 4.66	AVERAGE 4.59	AVERAGE 4.75	AVERAGE 4.83

Fonte: Criado pelo autor.

Figura 14 – Médias da combinação Gemini + CoT

COT + Gemini

Domínio	# Média Ator	# Média Ação	# Média Objeto	# Média condição	# Média Clareza	# Média completude	# Média Consistência	# Média Relevância
Financinhas - 1	100%	100%	100%	40%	4.1	3.95	4.8	4.5
Financinhas - 2	100%	100%	100%	100%	5	5	5	5
Abrace - 1	100%	100%	100%	80%	4.9	4.85	4.9	4.6
Abrace - 2	100%	100%	100%	60%	4.9	4.4	4.8	4.9
FakeNet - 1	100%	100%	100%	100%	4.43	4.36	5	4.86
CSI - 1	100%	100%	100%	57%	4.86	4.57	4.71	4.93
CSI - 2	100%	100%	100%	28.5%	4.71	4.57	4.43	5
Rapidu - 1	100%	100%	100%	70%	5	4.65	4.75	4.55
Rapidu - 2	40%	100%	100%	100%	4.9	4.4	5	4.9
+ New page								
	AVERAGE 93.33%	AVERAGE 100.00%	AVERAGE 100.00%	AVERAGE 70.61%	AVERAGE 4.76	AVERAGE 4.53	AVERAGE 4.82	AVERAGE 4.80

Fonte: Criado pelo autor.

Figura 15 – Médias da combinação Gemini + ToT

Médias: TOT + Gemini

Domínio	# Média Ator	# Média Ação	# Média Objeto	# Média condição	# Média Clareza	# Média completude	# Média Consistência	# Média Relevância
Financinhas - 1	100%	100%	100%	70%	4.5	4.45	4.8	4.9
Financinhas - 2	100%	100%	100%	100%	5	4.6	5	4.9
Abrace - 1	100%	100%	100%	80%	4.8	4.9	5	4.85
Abrace - 2	100%	100%	100%	70%	4.9	4.4	4.7	5
FakeNet - 1	100%	100%	100%	100%	4.65	4.7	4.8	4.9
CSI - 1	100%	100%	100%	50%	4.9	4.6	4.95	4.65
CSI - 2	100%	100%	100%	50%	4.9	4.7	4.7	4.7
Rapidu - 1	100%	100%	100%	62.5%	4.81	4.5	4.81	4.63
Rapidu - 2	25%	100%	100%	100%	5	4.25	5	4.88
+ New page								
	AVERAGE 91.67%	AVERAGE 100.00%	AVERAGE 100.00%	AVERAGE 75.83%	AVERAGE 4.83	AVERAGE 4.57	AVERAGE 4.86	AVERAGE 4.82

Fonte: Criado pelo autor.

Figura 16 – Médias da combinação DeepSeek + IO

Médias: IO + DeepSeek

Domínio	# Média Ator	# Média Ação	# Média Objeto	# Média condição	# Média Clareza	# Média completude	# Média Consistência	# Média Relevância
Financinhas - 1	100%	100%	100%	50%	4.1	3.9	4.7	4.7
Financinhas - 2	100%	100%	100%	90%	5	4.5	5	4.8
Abrace - 1	100%	100%	100%	40%	4.5	4.9	4.45	4.71
Abrace - 2	100%	100%	100%	40%	4.5	4.4	4.3	4.8
FakeNet - 1	100%	100%	100%	70%	4.25	4.3	4.6	4.8
CSI - 1	100%	100%	100%	60%	4.75	4.65	4.9	4.75
CSI - 2	100%	100%	100%	50%	5	4.7	4.6	4.9
Rapidu - 1	100%	100%	100%	80%	4.75	4.5	4.65	4.45
Rapidu - 2	80%	100%	100%	60%	4.9	4.4	5	4.9
+ New page								
	AVERAGE 97.78%	AVERAGE 100.00%	AVERAGE 100.00%	AVERAGE 60.00%	AVERAGE 4.64	AVERAGE 4.47	AVERAGE 4.69	AVERAGE 4.76

Fonte: Criado pelo autor.

Figura 17 – Médias da combinação DeepSeek + CoT

COT + DeepSeek

Domínio	# Média Ator	# Média Ação	# Média Objeto	# Média condição	# Média Clareza	# Média completude	# Média Consistência	# Média Relevância
Financinhas - 1	100%	100%	100%	50%	3.85	3.65	4.1	4.6
Financinhas - 2	100%	100%	100%	70%	5	4.7	5	5
Abrace - 1	100%	100%	100%	70%	4.95	4.8	4.9	4.75
Abrace - 2	100%	100%	100%	80%	5	4.7	4.8	5
FakeNet - 1	100%	100%	100%	100%	4.8	4.55	4.9	5
CSI - 1	100%	100%	100%	40%	4.45	3.8	4.55	4.6
CSI - 2	100%	100%	100%	40%	4.9	4.5	4.6	5
Rapidu - 1	100%	100%	100%	80%	4.9	4.75	4.65	4.55
Rapidu - 2	70%	100%	100%	90%	5	4.6	5	5
+ New page								
	AVERAGE 96.67%	AVERAGE 100.00%	AVERAGE 100.00%	AVERAGE 68.89%	AVERAGE 4.76	AVERAGE 4.45	AVERAGE 4.72	AVERAGE 4.83

Fonte: Criado pelo autor.

Figura 18 – Médias da combinação DeepSeek + ToT

Médias: TOT + DeepSeek

Domínio	# Média Ator	# Média Ação	# Média Objeto	# Média condição	# Média Clareza	# Média completude	# Média Consistência	# Média Relevância
Financinhas - 1	100%	100%	100%	60%	4.05	3.9	4.1	4.5
Financinhas - 2	100%	100%	100%	100%	5	4.9	5	4.9
Abrace - 1	100%	100%	100%	80%	5	4.85	5	4.9
Abrace - 2	100%	100%	100%	70%	5	4.6	4.6	5
FakeNet - 1	100%	100%	100%	70%	4.39	4.44	4.67	4.89
CSI - 1	100%	100%	100%	60%	4.7	4.5	4.85	4.7
CSI - 2	100%	100%	100%	70%	5	4.8	4.6	5
Rapidu - 1	100%	100%	100%	80%	4.9	4.6	4.7	4.8
Rapidu - 2	100%	100%	100%	80%	4.9	4.6	4.7	4.8
+ New page								
	VERAGE 100.00%	AVERAGE 100.00%	AVERAGE 100.00%	AVERAGE 74.44%	AVERAGE 4.77	AVERAGE 4.58	AVERAGE 4.69	AVERAGE 4.83

Fonte: Criado pelo autor.