Rubrica Analítica para Avaliação de Artefatos de Teste de Desempenho Gerados por LLMs: Um Estudo Exploratório

Lucie Grillo
Universidade Federal do Pampa (UNIPAMPA)
Alegrete, RS, Brasil
lucieaquino.aluno@unipampa.edu.br

Graziela Espindola Bitencourt
Universidade Federal do Pampa (UNIPAMPA)
Alegrete, RS, Brasil
grazielabitencourt.aluno@unipampa.edu.br

Maicon Bernardino
Universidade Federal do Pampa (UNIPAMPA)
Alegrete, RS, Brasil
bernardino@acm.org

Elder Macedo Rodrigues
Universidade Federal do Pampa (UNIPAMPA)
Alegrete, RS, Brasil
elderrodrigues@unipampa.edu.br

Resumo

Modelos de Linguagem de Grande Escala (Large Language Models, LLMs) têm sido investigados como apoio à geração de artefatos de teste de software, sobretudo em tarefas funcionais, como cenários BDD, casos de teste a partir de requisitos e planos de QA. Entretanto, cenários e casos de teste de desempenho exigem propriedades adicionais: escolha adequada do tipo de ensaio, definição de carga, preservação de fluxos e perfis de usuário, métricas observáveis, critérios numéricos de aceitação e viabilidade de execução. Este artigo apresenta uma rubrica analítica para avaliar a validade e a qualidade de artefatos de teste de desempenho gerados por LLMs. A rubrica contempla oito dimensões: correção técnica, rastreabilidade ao requisito, adequação do tipo de teste, coerência da carga, critérios numéricos de aceitação, completude, viabilidade de execução e clareza. O estudo é estruturado como uma avaliação empírica exploratória com duas condições de *prompting*: uma condição Baseline, na qual o modelo recebe o requisito, o contexto técnico e o template de saída; e uma condição Rubric-aware, na qual recebe os mesmos insumos acrescidos dos critérios operacionais da rubrica. Foram gerados 36 artefatos a partir de seis requisitos, três configurações de modelo e duas condições de *prompting*. Após adjudicação, todos os artefatos atenderam aos critérios invalidantes no contexto técnico fornecido aos modelos, mas a viabilidade de execução concentrou as notas mais baixas. Os resultados indicam que a rubrica é útil para revelar diferenças entre validade mínima, completude textual e operacionalização executável.

Palavras-chave

Testes de Desempenho, Modelos de Linguagem de Grande Escala, Rubrica Analítica, Requisitos Não Funcionais, Geração de Casos de Teste

1 Introdução

Testes de desempenho avaliam o comportamento de uma aplicação sob condições específicas de carga, estresse, resistência ou pico. Diferentemente de testes funcionais, que verificam se uma funcionalidade produz a resposta esperada, testes de desempenho exigem a definição de fluxo exercitado, perfil de usuário, volume de carga, crescimento da carga, duração, métricas observadas e critérios mensuráveis de aceitação [13]. Esses elementos tornam a especificação de cenários de desempenho uma atividade particularmente sensível à ambiguidade e à incompletude.

Requisitos de desempenho em linguagem natural frequentemente não indicam, de forma completa, limites, métricas, ambiente ou condições de verificação. Abdeen et al. mostram que a verificação de requisitos de desempenho depende tanto da interpretação do requisito quanto da geração de ambientes e condições de teste adequados [1]. A ausência de critérios numéricos e de rastreabilidade entre requisito e teste pode comprometer a interpretação dos resultados, sobretudo quando o objetivo é avaliar saturação, estabilidade ou vazão.

Ao mesmo tempo, LLMs têm sido explorados para gerar artefatos de teste a partir de requisitos e documentação textual. Estudos recentes investigam geração de cenários BDD, casos de teste de sistema, planos de QA e testes de aceitação em contextos industriais ou semi-industriais [3, 7, 11, 15, 18, 20, 32]. Esses trabalhos indicam que LLMs podem produzir artefatos compreensíveis e úteis como ponto de partida, mas também relatam limitações como redundância, casos fora de escopo, perda de restrições, alucinações e necessidade de revisão humana [3, 7, 18]. No contexto do SAST, trabalhos recentes também investigaram LLMs para geração de testes unitários, geração multimodal de testes GUI e detecção de conflitos semânticos com testes gerados por LLMs, reforçando o interesse da comunidade em avaliar usos de LLMs em diferentes atividades de teste [5, 9]. Entretanto, esses estudos não tratam especificamente da avaliação de artefatos de teste de desempenho.

A lacuna deste trabalho está na interseção entre geração de artefatos de teste por LLMs e avaliação de artefatos de teste de desempenho. Embora estudos recentes investiguem LLMs em testes funcionais, testes GUI, testes unitários, conflitos semânticos e artefatos de alto nível, ainda há pouca evidência sobre critérios específicos para avaliar artefatos de teste de desempenho. Por outro lado, a literatura sobre testes de desempenho destaca que cargas realistas dependem de sessões, perfis, dependências de dados, parâmetros operacionais e contexto de execução [6, 21, 22, 29]. Assim, um artefato de teste de desempenho gerado por LLMs pode parecer linguisticamente adequado e ainda ser tecnicamente inválido, incompleto ou inviável de executar.

Este artigo propõe uma rubrica analítica para avaliar artefatos de teste de desempenho gerados por LLMs. A rubrica separa critérios de validade mínima, como correção técnica, tipo de teste e critérios numéricos de aceitação, de critérios graduais de qualidade,

como rastreabilidade, coerência da carga, completude, viabilidade e clareza. Essa separação evita que um texto fluente receba uma avaliação positiva quando contém erros técnicos ou carece de condições mínimas para um teste de desempenho defensável.

As contribuições deste trabalho são: (1) uma rubrica analítica operacionalizada para avaliar cenários e casos de teste de desempenho gerados por LLMs; (2) um desenho experimental que compara uma condição Baseline e uma condição Rubric-aware sob o mesmo template de saída; e (3) uma aplicação empírica da rubrica em 36 artefatos, com avaliação humana por duas autoras, análise de concordância e identificação de dimensões que exigem adjudicação. O artigo não utiliza LLMs como juízes dos artefatos, pois essa decisão poderia introduzir circularidade, especialmente na condição em que a rubrica é fornecida ao modelo gerador.

2 Fundamentação Teórica e Trabalhos Relacionados

Esta seção reúne os fundamentos sobre testes de desempenho, geração de artefatos por LLMs e avaliação por rubricas que sustentam a proposta e o desenho experimental do estudo.

2.1 Testes de Desempenho e Requisitos Não Funcionais

O syllabus de teste de desempenho do ISTQB define testes de desempenho como atividades voltadas a avaliar eficiência, tempo de resposta, throughput, utilização de recursos e estabilidade sob diferentes cargas [13]. O mesmo corpo de conhecimento diferencia, entre outros, testes de carga, testes de estresse, testes de resistência e testes de pico. Essa distinção é central para a avaliação proposta neste trabalho, pois o tipo de teste condiciona o desenho da carga, a duração, as métricas e a interpretação dos resultados.

A especificação de requisitos de desempenho apresenta desafios próprios. Abdeen et al. propõem uma abordagem para verificação de requisitos de desempenho e geração de ambientes de teste, enfatizando que requisitos desse tipo precisam ser interpretados em termos de métricas, limites e condições observáveis [1]. Wang e Chen, por sua vez, tratam da quantificação automatizada de requisitos de desempenho em linguagem natural, reforçando que expressões qualitativas precisam ser convertidas em valores mensuráveis para apoiar verificação [30]. Essas fontes sustentam a necessidade de uma dimensão específica para critérios numéricos de aceitação.

Além da quantificação, a qualidade de um cenário de desempenho depende da representatividade do *workload*. WESSBAS extrai especificações probabilísticas de carga a partir de logs de sessão, modelando comportamento de usuários para apoiar testes de carga e predição de desempenho [29]. Schulz et al. propõem uma abordagem de *behavior-driven load testing* que utiliza conhecimento contextual para especificar testes de carga em linguagem estruturada [21]. Em trabalho posterior, Schulz et al. discutem geração de modelos de carga ajustados ao contexto para testes contínuos e representativos [22]. Esses trabalhos não tratam de LLMs, mas fundamentam o que significa uma carga coerente, contextualizada e executável.

Battista et al. apresentam o E2E-Loader, um framework para apoiar testes de desempenho de aplicações web a partir de testes fim-a-fim [6]. O trabalho evidencia que a execução de testes de desempenho depende de dados, sessões, protocolos, parâmetros de carga e preservação de comportamentos de usuário. Essa evidência é relevante para a dimensão de viabilidade de execução da rubrica, pois um artefato de teste de desempenho precisa ser mais do que uma descrição conceitual: ele deve conter informação suficiente para que o teste possa ser preparado e executado.

2.2 Geração de Artefatos de Teste por LLMs

LLMs têm sido avaliados em diferentes tarefas de geração de testes. Rathnayake et al. investigam a geração de cenários BDD com LLMs, observando que parâmetros de geração e estratégias de *prompting* afetam a qualidade dos resultados e que métricas lexicais podem não refletir avaliações humanas [20]. Arora et al. estudam a geração de cenários a partir de requisitos em contexto industrial com apoio de recuperação de informação, mostrando que artefatos gerados podem cobrir aspectos relevantes, mas ainda exigem revisão de correção e escopo [3]. Bhatia et al. analisam o uso de ChatGPT para projetar casos de teste de sistema a partir de especificações, relatando benefícios de clareza e cobertura, mas também redundância e desvios de requisito [7].

Outros estudos reforçam a necessidade de protocolos de avaliação específicos para artefatos de engenharia de software. Hasan et al. investigam geração automática de casos de teste de alto nível com LLMs [11]. Masuda et al. analisam geração de casos de teste de alto nível em contexto industrial [15]. Xue et al. propõem um pipeline para geração de testes de aceitação em software financeiro, evidenciando que regras de domínio complexas dificultam o uso direto de modelos generalistas [32]. Pangas et al. relatam o uso de LLMs para preencher lacunas em planos de teste no Firefox, observando que artefatos úteis ainda dependem de julgamento humano [18].

Esses trabalhos são centrais para situar a pesquisa, mas não devem ser lidos como evidência direta sobre geração de testes de desempenho. A contribuição deste artigo parte justamente dessa lacuna: adaptar a avaliação de artefatos gerados por LLMs a critérios específicos de desempenho, como coerência da carga, limiares numéricos e viabilidade de execução.

2.3 LLMs, Requisitos Não Funcionais e *Prompting*

A engenharia de *prompt* fornece estratégias para estruturar instruções, contexto técnico e formato de resposta em tarefas complexas. Catálogos de padrões de *prompt* e revisões sobre *prompt engineering* em Engenharia de Software indicam que a forma da instrução influencia o comportamento do modelo, mas também apontam fragilidade, baixa padronização e necessidade de documentação cuidadosa dos prompts [4, 26, 31]. Neste estudo, os prompts são tratados como prompts estruturados baseados em requisito, contexto técnico e formato obrigatório de resposta, sem exemplos preenchidos de saída.

No caso de requisitos não funcionais, o problema é ainda mais delicado. Estudos sobre geração de código com requisitos não funcionais indicam que atributos como performance, eficiência, robustez e uso de recursos exigem avaliação específica e podem produzir trade-offs com funcionalidade [14, 17, 25]. Embora esses trabalhos

não avaliem casos de teste de desempenho, eles sustentam a necessidade de critérios próprios para avaliar artefatos relacionados a requisitos não funcionais.

Estudos empíricos envolvendo LLMs também exigem documentação cuidadosa. Baltes et al. recomendam relatar modelo, versão, configuração, *prompts*, traços de execução, baselines e validação humana quando LLMs são usados em estudos de Engenharia de Software [4]. Essas recomendações orientam o protocolo experimental deste artigo, especialmente quanto ao registro de modelos, configurações, prompts enviados e artefatos gerados.

2.4 Qualidade de Testes em Linguagem Natural e Avaliação por Rubricas

A qualidade de casos de teste é multidimensional. Tran et al. investigam atributos de qualidade valorizados por profissionais em casos e suítes de teste, incluindo usabilidade, manutenibilidade, confiabilidade e capacidade de detecção de falhas [27]. A qualidade de especificações em linguagem natural também pode ser comprometida por ambiguidades, instruções incompletas e problemas estruturais. Hauptmann et al. introduzem a noção de *test smells* em testes escritos em linguagem natural [12], e Soares et al. catalogam e identificam *smells* em testes manuais, fornecendo base para avaliar clareza e completude de artefatos textuais de teste [24].

Rubricas analíticas ajudam a decompor avaliações complexas em dimensões observáveis. Rao e Callison-Burch discutem avaliação baseada em rubricas para saídas de LLMs, destacando a importância de critérios explícitos e independentes [19]. Graziotin et al. apresentam diretrizes metodológicas para instrumentos em Engenharia de Software comportamental, enfatizando clareza de construto, consistência e cautela ao interpretar medidas [10]. Usman et al. propõem um instrumento de avaliação de qualidade para revisões sistemáticas em Engenharia de Software, reforçando a importância de critérios explícitos e guias de aplicação [28]. Para relatar concordância entre avaliadores, Díaz et al. discutem confiabilidade e acordo entre avaliadores em estudos colaborativos de Engenharia de Software [8]. Essas fontes fundamentam a operacionalização e o protocolo de avaliação humana da rubrica proposta.

3 Rubrica Analítica

A rubrica foi construída para avaliar artefatos compostos por duas partes: um cenário de teste de desempenho e um caso de teste de desempenho. O cenário descreve o objetivo estratégico do teste, o requisito de origem, o tipo de ensaio, o fluxo-alvo, o perfil de usuários, o vetor de carga, as métricas e os critérios de aceitação. O caso de teste descreve a operacionalização desse cenário, incluindo pré-condições, dados necessários, ferramenta ou mecanismo de execução, parâmetros de carga, passos ou comando, critérios de falha e evidências esperadas.

Essa distinção evita confundir um plano conceitual com um artefato executável. Um cenário pode estar alinhado ao requisito e ainda carecer de detalhes operacionais. Um caso pode conter um comando ou uma sequência de passos e ainda não estar rastreado a um requisito válido. Por isso, a unidade de análise deste estudo é o artefato completo, composto por cenário e caso.

3.1 Fundamentação da Construção

A rubrica foi construída a partir da combinação de quatro grupos de evidência da literatura e de critérios técnicos derivados do domínio experimental. O primeiro grupo envolve requisitos e testes de desempenho. Fontes normativas e técnicas indicam que um teste de desempenho não pode ser avaliado apenas por sua descrição textual: ele precisa explicitar objetivo, tipo de ensaio, carga, ambiente, métricas observáveis e critérios de aceitação [1, 13]. Trabalhos sobre quantificação de requisitos de desempenho reforçam que expressões vagas precisam ser transformadas em parâmetros verificáveis antes que um teste possa ser interpretado de modo consistente [30]. Essas fontes sustentam dimensões como rastreabilidade, adequação do tipo de teste e critérios numéricos de aceitação.

O segundo grupo envolve modelagem de *workload* e execução de testes de desempenho. WESSBAS mostra que cargas representativas dependem de sessões, perfis de usuários, tempos de pensamento, distribuições de requisições e dependências entre ações [29]. Schulz et al. aproximam testes de carga de especificações estruturadas em linguagem natural, mas com contexto, *workload*, duração, métricas e *quality gates* explícitos [21]. Em trabalho posterior, os autores destacam que cargas podem variar conforme contexto operacional e que testes contínuos exigem modelos ajustados ao contexto [22]. Battista et al. reforçam a dimensão operacional ao mostrar que testes de desempenho web dependem de sessões, dados, protocolos, correlações e parâmetros executáveis [6]. Essas fontes fundamentam principalmente coerência da carga e viabilidade de execução.

O terceiro grupo envolve qualidade de artefatos de teste em linguagem natural. Casos de teste possuem atributos de qualidade que variam conforme contexto e que precisam ser definidos de modo observável para apoiar revisão e manutenção [27]. Além disso, testes escritos em linguagem natural podem conter defeitos estruturais e semânticos, como ambiguidade, conhecimento tácito, ações sem verificação e pré-condições mal posicionadas [12, 24]. Esse grupo de fontes sustenta as dimensões de completude e clareza, mas também reforça que fluência textual não equivale a qualidade técnica.

O quarto grupo envolve avaliação por instrumentos e estudos empíricos com LLMs. Rubricas analíticas permitem decompor julgamentos complexos em critérios explícitos, reduzindo a dependência de uma impressão global da saída [19]. Ao mesmo tempo, diretrizes sobre instrumentos em Engenharia de Software recomendam explicitar construtos, itens, escalas, procedimentos de aplicação e limites de validade [10, 28]. Como o estudo envolve artefatos gerados por LLMs, também é necessário documentar prompts, versões, parâmetros, artefatos e validação humana [4]. A aplicação inicial da rubrica por duas avaliadoras e o relato de concordância seguem essa lógica de transparência, sem tratar a rubrica como instrumento psicometricamente validado em sentido amplo [8].

3.2 Dimensões e Escalas

A rubrica possui oito dimensões. D1, D3 e D5 são critérios binários e invalidantes. D2, D4, D6, D7 e D8 são critérios ordinais em escala de 1 a 3. A Tabela 1 apresenta uma visão consolidada das dimensões.

Tabela 1: Dimensões da rubrica para artefatos de teste de desempenho

Dimensão	Natureza	Escala	Âncora Operacional	Fontes Principais
D1 – Correção Técnica	Geral	Binária	Atende se usa apenas elementos compatíveis com o sistema, o benchmark e o contexto fornecido; não atende se inventa endpoints, componentes, métricas, flags ou recursos não fornecidos.	[2, 4, 23]
D2 – Rastreabilidade ao Requisito	Requisito	Ordinal (1–3)	Avalia se objetivo, fluxo, carga, métricas e critérios do artefato preservam o requisito original e suas restrições.	[1, 3, 7, 18]
D3 – Adequação do Tipo de Teste	Desempenho	Binária	Atende quando o tipo de teste escolhido, como carga, estresse, resistência ou pico, corresponde ao objetivo do requisito.	[1, 13]
D4 – Coerência da Carga	Desempenho	Ordinal (1–3)	Avalia se usuários, taxa, *ramp-up*, duração, perfil, mix de operações e demais parâmetros formam um vetor de carga coerente.	[6, 21, 22, 29]
D5 – Critérios Numéricos de Aceitação	Desempenho	Binária	Atende quando há limites mensuráveis associados a métricas observáveis, como latência, vazão, WIPS, taxa de erro ou uso de recursos.	[1, 13, 30]
D6 – Completude	Geral	Ordinal (1–3)	Avalia se campos essenciais do cenário e do caso estão preenchidos de modo suficiente para compreensão, revisão e reprodução.	[21, 24, 27]
D7 – Viabilidade de execução	Execução	Ordinal (1–3)	Avalia se o artefato pode ser executado com dados, ferramenta, parâmetros, pré-condições, evidências e forma de análise especificados, sem contradições operacionais.	[1, 6, 22, 29]
D8 – Clareza	Geral	Ordinal (1–3)	Avalia se o artefato é compreensível, organizado, não ambíguo e sem contradições que prejudiquem revisão ou execução.	[12, 16, 24, 27]

3.3 Critérios Invalidantes

Um artefato será considerado válido apenas se atender simultaneamente D1, D3 e D5. A regra é:

$$Valido(a) = D1(a) \wedge D3(a) \wedge D5(a)$$

Essa regra impede que artefatos linguisticamente claros, mas tecnicamente incorretos, recebam avaliação global positiva. Por exemplo, um caso que inventa endpoints inexistentes falha em D1; um cenário que declara teste de carga, mas busca ponto de ruptura, falha em D3; e um artefato que usa apenas expressões como "rápido", "aceitável" ou "estável", sem limiares mensuráveis, falha em D5.

Os três critérios foram definidos como invalidantes porque representam condições mínimas para que o artefato possa ser tratado como teste de desempenho. D1 impede que a avaliação premie saídas plausíveis, mas incompatíveis com o sistema ou com o benchmark. Esse risco é particularmente relevante em artefatos gerados por LLMs, pois estudos sobre geração de testes e explicações de teste apontam que saídas compreensíveis ainda podem conter omissões, extrapolações ou incorreções técnicas [2, 3, 7, 18]. No domínio TPC-W, por exemplo, inventar uma rota, uma métrica ou uma flag do RBE compromete a relação entre o artefato e o sistema sob teste [23].

D3 é invalidante porque o tipo de teste determina a estrutura do experimento. Testes de carga, estresse, resistência e pico exercitam objetivos diferentes e exigem decisões distintas sobre crescimento da carga, duração, ponto de observação e interpretação do resultado [13]. Um artefato que escolhe o tipo errado pode até conter números e comandos, mas passa a responder a uma pergunta diferente da formulada pelo requisito. D5, por sua vez, é invalidante porque requisitos de desempenho só se tornam verificáveis quando associados a métricas e limiares observáveis [1, 30]. Sem critérios numéricos, não há base para distinguir sucesso, falha, degradação aceitável ou saturação.

As dimensões ordinais permanecem úteis mesmo quando um artefato é invalidado. Elas permitem diagnosticar se a falha está concentrada em rastreabilidade, carga, completude, viabilidade ou clareza. Contudo, a soma ordinal não deve compensar falhas invalidantes.

3.4 Dimensões Ordinais

Nas dimensões ordinais, a nota 1 indica ausência ou falha substantiva, a nota 2 indica atendimento parcial e a nota 3 indica atendimento adequado. Em D2, a nota 1 representa perda ou inversão do requisito; a nota 2 indica preservação parcial do objetivo com lacunas; e a nota 3 indica conexão explícita entre requisito, objetivo, fluxo, carga, métricas e critérios. Em D4, a nota 1 representa carga ausente, contraditória ou incompatível; a nota 2 representa vetor parcialmente especificado; e a nota 3 representa carga coerente e reproduzível.

Em D6, a nota 1 indica ausência de campos essenciais; a nota 2 indica campos presentes, mas incompletos; e a nota 3 indica preenchimento suficiente para revisão e reprodução. Em D7, a nota 1 indica caso não executável; a nota 2 indica execução possível apenas com ajustes relevantes; e a nota 3 indica execução viável com ferramenta, dados, parâmetros, evidências e forma de análise definidos. Em D8, a nota 1 indica ambiguidade ou contradição relevante; a nota 2 indica clareza parcial; e a nota 3 indica texto organizado, preciso e sem contradições relevantes.

D7 foi calibrada para distinguir executabilidade sintática de viabilidade operacional. A presença de um comando RBE sintaticamente plausível não basta, por si só, para nota 3. A nota 3 exige que os valores necessários à execução estejam concretos e que o artefato

indique como coletar ou verificar as evidências exigidas pelo requisito, como logs, taxa de erro, percentis, janelas temporais ou interações específicas. A nota 2 é atribuída quando a execução é plausível, mas ainda depende de substituir *placeholders*, confirmar disponibilidade de logs ou definir pós-processamento relevante. A nota 1 é reservada para artefatos sem procedimento executável ou com contradições operacionais que impeçam a execução. A rubrica não exige que o artefato entregue scripts auxiliares completos; exige que as dependências operacionais estejam explícitas o suficiente para orientar a execução e a análise por uma pessoa avaliadora.

As dimensões ordinais diagnosticam problemas distintos; assim, um artefato pode ser claro e completo, mas usar carga inadequada, ser pouco executável ou não preservar o requisito original.

No caso de testes de desempenho, essa distinção é especialmente importante. A literatura de *workload* mostra que representatividade e execução dependem de elementos que não aparecem em casos de teste funcionais simples, como mix de operações, sessões, tempos de pensamento, duração, ramp-up, ramp-down, dados e dependências entre requisições [6, 29]. Portanto, D4 e D7 funcionam como dimensões específicas de desempenho dentro da rubrica: elas avaliam se a saída gerada se aproxima de um experimento de desempenho configurável, e não apenas de uma descrição textual plausível.

3.5 Pontuação Diagnóstica

Para artefatos válidos, será calculada uma pontuação diagnóstica ordinal:

$$ScoreOrdinal(a) = D2(a) + D4(a) + D6(a) + D7(a) + D8(a)$$

Essa pontuação varia de 5 a 15 e será usada apenas como síntese complementar. A análise principal será por dimensão, pois médias globais podem ocultar falhas tecnicamente importantes. A decisão de validade será sempre determinada pelos critérios D1, D3 e D5.

A pontuação ordinal não recebe pesos diferenciados nesta versão da rubrica. Essa decisão evita introduzir uma hierarquia artificial entre dimensões sem evidência empírica suficiente para justificar pesos. Em vez disso, o estudo reporta distribuições por dimensão, exemplos qualitativos de falhas e a taxa de validade mínima. Assim, a pontuação funciona como resumo descritivo para artefatos que já passaram pelos critérios invalidantes, enquanto a interpretação substantiva permanece ancorada nas dimensões individuais.

3.6 Escopo e Limites

A rubrica avalia a qualidade do artefato produzido, não o desempenho real do sistema. Um artefato com nota alta deve estar tecnicamente alinhado ao contexto fornecido, rastreado ao requisito, completo, claro e suficientemente operacional para orientar execução. Isso não significa que o sistema passará no teste, nem que o *workload* representa produção real em sentido estatístico. No domínio TPC-W, por exemplo, a rubrica verifica se o artefato usa corretamente RBE, WIPS, mixes, dataset e parâmetros fornecidos; ela não substitui a execução do benchmark nem a análise dos resultados medidos [23].

Também não se afirma que a rubrica esteja validada como instrumento universal. O objetivo é oferecer uma rubrica analítica operacionalizada para este estudo, fundamentada em literatura de desempenho, qualidade de testes, *workload* e avaliação de saídas de LLMs. A validade de uso depende do procedimento empírico: calibração das avaliadoras, aplicação inicial independente entre avaliadoras, justificativas por dimensão, cálculo de concordância e adjudicação de divergências. Por isso, a rubrica deve ser interpretada como instrumento de avaliação estruturada e diagnóstico, não como métrica absoluta de qualidade.

Por fim, a rubrica foi desenhada para avaliar artefatos textuais compostos por cenário e caso de teste de desempenho. Ela pode ser adaptada para outros domínios, ferramentas ou benchmarks, mas essa adaptação exigiria revisar D1, D4 e D7, pois esses critérios dependem fortemente do sistema sob teste, das métricas observáveis e do mecanismo de geração de carga. Essa limitação é deliberada: uma rubrica genérica demais reduziria o risco de erro contextual, mas perderia poder para detectar falhas técnicas específicas de desempenho.

4 Metodologia Experimental

Esta seção descreve o desenho empírico para avaliar artefatos de teste de desempenho gerados por LLMs. O experimento compara duas condições de *prompting* sob o mesmo template de saída e aplica a rubrica por meio de avaliação humana realizada inicialmente de forma independente entre duas avaliadoras.

4.1 Questões de Pesquisa

O estudo é guiado pelas seguintes questões:

- **QP1:** Em que medida artefatos de teste de desempenho gerados por LLMs atendem aos critérios invalidantes da rubrica?
- **QP2:** Quais dimensões da rubrica concentram mais falhas nos artefatos gerados?
- **QP3:** A condição Rubric-aware produz artefatos com maior pontuação diagnóstica que a condição Baseline?
- **QP4:** Qual é o nível de concordância entre avaliadoras humanas ao aplicar a rubrica operacionalizada?

4.2 Sistema Sob Teste e Contexto Técnico

O estudo utiliza o benchmark TPC-W como domínio experimental controlado, não como representação direta de aplicações web modernas. O TPC-W simula uma aplicação de comércio eletrônico, com navegação, busca, carrinho e compra, e define um Remote Browser Emulator (RBE) para gerar tráfego HTTP semelhante ao de navegadores reais [23]. A métrica Web Interactions Per Second (WIPS) foi utilizada quando o requisito demandava vazão, pois é própria do benchmark [23].

O contexto técnico fornecido aos modelos foi reconstruído a partir de fontes primárias do benchmark e da implementação efetivamente usada no experimento. Esse contexto incluiu URL base, fluxos disponíveis, parâmetros do RBE, massa de dados, métricas observáveis e exemplos de comando quando aplicável. O mesmo contexto técnico foi fornecido às duas condições de *prompting*, para que a comparação não fosse contaminada por acesso desigual a informação de domínio.

4.3 Corpus de Requisitos

O corpus foi composto por requisitos de desempenho documentados como artefatos independentes. Esses requisitos foram formulados a

partir do domínio experimental TPC-W, de seus fluxos de navegação e compra, dos mixes exercitados pelo RBE, da escala fixa adotada no experimento e da métrica WIPS [23]. Portanto, eles não são tratados como requisitos oficiais de certificação do benchmark, mas como requisitos experimentais derivados de um domínio controlado. Cada requisito contém identificador, descrição textual, objetivo esperado, restrições conhecidas e, quando aplicável, tipo de teste esperado. O corpus cobre testes de carga, estresse, resistência e pico.

A separação entre requisito, contexto técnico e prompt é necessária para avaliar D2. Se o requisito estiver embutido de modo implícito no prompt, torna-se difícil verificar se o artefato preservou o objetivo original ou apenas seguiu instruções de geração.

4.4 Condições de *Prompting*

Foram comparadas duas condições:

(1) **Baseline:** o modelo recebeu o requisito, o contexto técnico fixo e o template de saída. Não recebeu a rubrica nem critérios explícitos de qualidade além da instrução de preencher o template.
(2) **Rubric-aware:** o modelo recebeu exatamente o mesmo requisito, o mesmo contexto técnico e o mesmo template, acrescidos dos critérios operacionais da rubrica.

Não foi incluída uma condição separada de guia de domínio. Informações de domínio indispensáveis para gerar artefatos avaliáveis foram fornecidas igualmente nas duas condições. Assim, a diferença experimental foi a exposição explícita à rubrica, e não a quantidade de contexto técnico.

4.5 Template de Saída

Todas as condições usaram o mesmo template de saída. A parte de cenário continha: identificador, requisito de origem, objetivo, tipo de teste, fluxo-alvo, perfil de usuário ou mix de operações, vetor de carga, métricas observadas, ambiente ou premissas de ambiente e critérios de aceitação. A parte de caso de teste continha: cenário associado, pré-condições, massa de dados, ferramenta ou mecanismo de execução, parâmetros de carga, passos ou comando, métricas coletadas, critérios de aceitação e falha, e evidências esperadas.

O uso de um formato fixo reduz o risco de confundir qualidade técnica com diferenças superficiais de estrutura textual. Esse formato não contém exemplos preenchidos de resposta; ele apenas define os campos que o artefato deve conter. Desse modo, a avaliação pode se concentrar na validade, coerência, completude e viabilidade do conteúdo produzido.

4.6 Modelos e Execução

Foram utilizadas três configurações de modelo, registradas nos artefatos experimentais conforme a interface de execução: Gemini 3.1 Pro Preview em modo High, Claude Opus 4.6 com *thinking*/*adaptive thinking*, e GPT-5.5 via Codex com esforço de raciocínio *xhigh*. A preservação dos prompts efetivamente enviados, das respostas completas e da organização por modelo, requisito e condição de *prompting* segue recomendações para documentação de estudos empíricos com LLMs [4].

Cada combinação de requisito, modelo e condição produziu um artefato. As saídas foram armazenadas integralmente, sem correção manual, para preservar a rastreabilidade do experimento. Para cada combinação, foi realizada uma única geração; o desenho experimental não incluiu repetições destinadas a estimar a variabilidade estocástica das respostas dos modelos.

Na execução atual, o corpus contém seis requisitos de desempenho derivados do domínio TPC-W, selecionados para cobrir diferentes tipos de ensaio: carga, estresse, resistência e pico. Três configurações de modelo foram usadas em duas condições de *prompting*, Baseline e Rubric-aware, totalizando 36 artefatos gerados. Os artefatos foram armazenados sem edição manual antes da avaliação.

4.7 Avaliação Humana

A avaliação foi conduzida por duas autoras do estudo, atuando como avaliadoras humanas. Cada avaliadora recebeu o requisito original, o contexto técnico fixo, o artefato gerado e a rubrica operacionalizada, e a avaliação inicial foi realizada de forma independente entre elas. A avaliação não foi cega quanto ao modelo ou à condição de *prompting*, pois os artefatos e planilhas de avaliação preservaram esses identificadores para rastreabilidade. Essa decisão facilita auditoria dos resultados, mas é tratada como ameaça à validade interna.

As avaliadoras atribuíram notas para D1–D8 e registraram justificativas curtas por dimensão. As divergências foram analisadas após o cálculo de concordância inicial e resolvidas por adjudicação entre as avaliadoras. Esse procedimento separa a confiabilidade inicial da rubrica da decisão final adjudicada.

4.8 Análise dos Dados

A análise reporta a taxa de validade por condição, a frequência de falhas em D1, D3 e D5, a distribuição das notas ordinais em D2, D4, D6, D7 e D8, e a pontuação ordinal diagnóstica dos artefatos válidos. A comparação entre Baseline e Rubric-aware é descritiva e tem finalidade diagnóstica, não confirmatória. Não foram utilizados testes inferenciais, pois o tamanho da amostra e o objetivo diagnóstico da rubrica favorecem uma análise por dimensão e por padrão qualitativo de divergência.

A concordância entre avaliadoras foi calculada antes da adjudicação. Para critérios binários, foram reportados percentual de acordo e, quando aplicável, Cohen's Kappa. Para dimensões ordinais, foi considerada concordância ponderada. A interpretação segue cautela metodológica, pois baixa concordância pode indicar ambiguidade da rubrica, casos difíceis ou diferença substantiva de julgamento, e não apenas erro de avaliadoras [8].

5 Resultados

Foram avaliados 36 artefatos, resultantes da combinação de seis requisitos, três configurações de modelo e duas condições de *prompting*. Esta seção relata a concordância inicial entre avaliadoras e os resultados consolidados após adjudicação. A concordância inicial indica a estabilidade de aplicação da rubrica; a pontuação adjudicada representa a leitura final usada para responder às questões de pesquisa.

5.1 Validade Mínima

As duas avaliadoras classificaram todos os 36 artefatos como válidos segundo os critérios invalidantes D1, D3 e D5, e essa decisão foi

mantida após adjudicação. Em outras palavras, não houve discordância quanto à presença mínima de correção técnica, adequação do tipo de teste e critérios numéricos de aceitação. Esse resultado responde à QP1 no contexto do corpus e do contexto técnico fornecido: todos os artefatos atenderam à validade mínima exigida pela rubrica. Esse achado deve ser interpretado junto ao desenho dos prompts. Tanto a condição Baseline quanto a condição Rubric-aware receberam o mesmo contexto técnico do TPC-W, incluindo rotas válidas, factories RBE, métricas observáveis, dataset e formato obrigatório de resposta. Assim, a ausência de falhas invalidantes não significa que os artefatos sejam diretamente executáveis ou tecnicamente perfeitos; significa que, para as avaliadoras, eles não violaram as condições mínimas definidas pela rubrica.

5.2 Concordância Entre Avaliadoras

A Tabela 2 apresenta o acordo entre avaliadoras por dimensão antes da adjudicação. Houve acordo total nas dimensões invalidantes D1, D3 e D5. Entre as dimensões ordinais, D2 e D4 tiveram 83,3% de acordo, D6 e D8 tiveram 77,8%, e D7 apresentou o menor acordo, com 50,0%. A interpretação de Kappa deve ser cautelosa, pois em várias dimensões houve baixa variância ou uso de uma única categoria por uma avaliadora, situação em que o coeficiente pode ser pouco informativo apesar de o percentual de acordo ser alto [8].

Tabela 2: Concordância entre avaliadoras antes da adjudicação

Dimensão	Acordo	Div.	Kappa*
D1	100,0%	0	n.a.
D2	83,3%	6	0,00
D3	100,0%	0	n.a.
D4	83,3%	6	0,00
D5	100,0%	0	n.a.
D6	77,8%	8	0,00
D7	50,0%	18	0,09
D8	77,8%	8	0,00

Na Tabela 2, Kappa* indica Cohen's Kappa para dimensões binárias e Kappa ponderado linear para dimensões ordinais; n.a. indica ausência de variância suficiente para interpretar o coeficiente.

As divergências em D7 indicam uma diferença substantiva de interpretação da rubrica. Uma avaliadora adotou uma leitura mais operacional, penalizando artefatos que exigiam substituição de *placeholders*, confirmação de logs, pós-processamento para métricas específicas, cálculo de percentis ou análise por janelas temporais. A outra avaliadora atribuiu nota máxima com maior frequência quando o comando Docker/RBE e os parâmetros principais estavam presentes e sintaticamente executáveis. Essa diferença é metodologicamente relevante: ela mostra que a viabilidade de execução precisa ser calibrada com exemplos, pois não se reduz à presença de um comando.

5.3 Pontuações Diagnósticas

A Tabela 3 mostra a pontuação ordinal diagnóstica por avaliadora e a pontuação final adjudicada. As pontuações foram recalculadas a partir das dimensões D2, D4, D6, D7 e D8, seguindo a fórmula da rubrica. As avaliações iniciais já indicavam médias maiores para a condição Rubric-aware. Após adjudicação, a média final foi 13,28 na condição Baseline e 13,72 na condição Rubric-aware, com média geral de 13,50. Em comparação pareada, a condição Rubric-aware melhorou a pontuação em 6 dos 18 pares modelo-requisito e permaneceu igual nos demais, com diferença média de 0,44 ponto.

Tabela 3: Pontuação ordinal por avaliação e condição

Avaliação	Baseline	Rubric-aware	Geral
Avaliadora 1 inicial	13,28	13,72	13,50
Avaliadora 2 inicial	14,28	14,39	14,33
Final adjudicada	13,28	13,72	13,50

A diferença entre avaliadoras também é informativa. A Avaliadora 2 foi mais permissiva nas dimensões D2, D4 e D6, atribuindo nota 3 a todos os artefatos nessas dimensões. A Avaliadora 1 foi mais restritiva em casos nos quais havia parâmetros assumidos, *placeholders* ou divergência em relação às notas de avaliação dos requisitos. A adjudicação consolidou uma leitura mais operacional para D2, D4, D6 e D7, e preservou nota 3 em D8 quando a divergência se referia apenas a verbosidade ou redundância sem ambiguidade técnica.

A Tabela 4 apresenta a distribuição final das notas ordinais. Nenhum artefato recebeu nota 1 nas dimensões ordinais. A dimensão D7 concentrou as notas parciais: 34 artefatos receberam nota 2 e apenas 2 receberam nota 3. D2 e D4 tiveram 6 notas parciais cada, principalmente associadas ao REQ-TPCW-06. D6 teve 8 notas parciais, associadas a *placeholders* ou campos operacionais ainda abertos. D8 recebeu nota 3 em todos os artefatos após adjudicação.

Tabela 4: Distribuição das notas ordinais após adjudicação

Dimensão	Nota 2	Nota 3	Principal interpretação
D2	6	30	Perdas de rastreabilidade concentradas em REQ-TPCW-06.
D4	6	30	Divergências de factory, mix ou ramp-down em REQ-TPCW-06.
D6	8	28	*Placeholders* e campos operacionais ainda abertos.
D7	34	2	Execução dependente de logs, pós-processamento ou extração de métricas.
D8	0	36	Clareza preservada; verbosidade isolada não foi penalizada.

5.4 Padrões de Divergência

O principal padrão de divergência ocorreu em D7, com 18 divergências em 36 artefatos na avaliação inicial. Após adjudicação, D7 permaneceu como a dimensão mais crítica. Isso responde à QP2: mesmo quando os artefatos parecem completos e claros, a viabilidade de execução concentra incertezas. Em vários casos, o artefato

contém comando RBE, mas a avaliação dos critérios exige informação adicional sobre formato do arquivo de saída, disponibilidade de logs, cálculo de percentis, extração de interações Buy Confirm ou comparação de WIPS por janelas temporais. Esses elementos são centrais em testes de desempenho e sustentam a decisão de manter D7 separada de completude textual.

O requisito REQ-TPCW-06 concentrou divergências em D2 e D4. Esse requisito trata de pico de compras e exige subida rápida para 70 browsers emulados, três minutos de medição, verificação de taxa de erro, ausência de falha em massa em Buy Confirm e recuperação sem HTTP 5xx após desaceleração. As notas de avaliação esperavam factory shopping/rbe.EBTPCW2Factory e ramp-down de 60 segundos. Alguns artefatos usaram factory ordering/rbe.EBTPCW3Factory, ramp-down de 15 ou 30 segundos, ou deixaram o ramp-down indefinido. Esses casos explicam por que uma mesma saída pode preservar o objetivo geral de pico, mas ainda receber nota parcial em rastreabilidade ou coerência da carga.

As divergências em D6 ocorreram principalmente quando comandos continham *placeholders*, como `<arquivo_saida>` ou `<max_erros>`. Pela rubrica, esses casos foram adjudicados como parcialmente completos: o campo existe e a intenção é clara, mas ainda há informação operacional a ser definida antes da execução. Em D8, as divergências foram diferentes: a discussão não foi sobre ambiguidade técnica forte, mas sobre verbosidade e redundância. A adjudicação decidiu que verbosidade sem prejuízo interpretativo não reduz a nota de clareza.

6 Discussão

Os resultados sugerem que fornecer requisito, contexto técnico e formato obrigatório de resposta foi suficiente para evitar falhas invalidantes nos artefatos gerados. As duas avaliadoras concordaram que todos os artefatos atenderam D1, D3 e D5, e a adjudicação manteve essa decisão. Contudo, esse achado não implica que os artefatos estejam prontos para execução sem revisão. A distribuição final de D7 mostra que há diferença entre produzir um comando plausível e especificar evidências, logs, pós-processamento e critérios de análise de forma suficientemente operacional.

A ausência de falhas invalidantes não reduz a utilidade da rubrica, pois mostra que validade mínima e qualidade operacional são níveis distintos de avaliação. No desenho adotado, o contexto técnico fornecido foi suficiente para evitar erros graves de domínio, mas não para garantir viabilidade de execução. Assim, o principal valor diagnóstico da rubrica aparece nas dimensões ordinais, especialmente D7.

A condição Rubric-aware apresentou aumento descritivo modesto na pontuação ordinal adjudicada, mas a magnitude da diferença foi pequena. Essa leitura é compatível com a função esperada da rubrica no prompt: tornar explícitos os campos e critérios que o artefato deveria contemplar. Ainda assim, a melhora não deve ser interpretada como evidência de superioridade geral dos modelos nessa condição. Como a rubrica usada para avaliar também informa a condição Rubric-aware, o resultado pode indicar maior alinhamento ao instrumento, não necessariamente maior capacidade geral de geração.

O padrão observado em REQ-TPCW-06 merece atenção especial. Testes de pico exigem não apenas carga máxima e tempo de subida, mas também uma interpretação operacional da desaceleração, da recuperação e das evidências necessárias para caracterizar falha ou retorno à estabilidade. Os artefatos que assumiram ramp-down diferente do esperado, deixaram o valor indefinido ou selecionaram factory incompatível com o mix de compras mostram uma limitação importante: o modelo pode preservar a intenção geral do requisito e ainda perder detalhes experimentais que afetam a avaliação da carga.

Por fim, os resultados reforçam a utilidade diagnóstica da rubrica. Se fosse usada apenas uma nota global, seria fácil concluir que os artefatos são majoritariamente bons, especialmente porque todos passaram pelos critérios invalidantes. A análise por dimensão revela um quadro mais específico: validade mínima, completude textual e clareza não garantem viabilidade de execução. Essa distinção é central para o uso de LLMs como apoio a testes de desempenho, pois a etapa humana de revisão precisa examinar justamente os pontos em que a saída parece completa, mas ainda depende de decisões operacionais externas.

A própria divergência entre avaliadoras é um resultado metodológico relevante. D1, D3 e D5 parecem ter âncoras suficientemente estáveis no contexto do experimento, embora esse resultado também possa ter sido influenciado pelo contexto técnico detalhado fornecido aos modelos. D7, por outro lado, exigiu calibração adicional: uma interpretação centrada na sintaxe do comando leva a notas mais altas, enquanto uma interpretação centrada na execução completa e na análise das evidências leva a notas mais restritivas. A adjudicação adotou a segunda leitura, por ser mais compatível com a definição de viabilidade de execução. Futuras aplicações da rubrica devem incluir exemplos calibrados para D7, especialmente envolvendo logs, percentis, janelas temporais e artefatos com *placeholders*.

7 Ameaças à Validade

7.1 Validade Interna

A principal ameaça à validade interna é a possibilidade de que diferenças entre condições sejam causadas por variações no contexto fornecido, e não pela presença da rubrica. Para mitigar esse risco, as duas condições receberam o mesmo requisito, o mesmo contexto técnico e o mesmo template de saída. A única diferença foi a presença dos critérios operacionais da rubrica na condição Rubric-aware.

Outra ameaça é a circularidade da condição Rubric-aware. Como os critérios usados na avaliação também são fornecidos ao modelo gerador, artefatos dessa condição podem se alinhar melhor à forma da rubrica sem necessariamente representar uma melhoria geral da capacidade do modelo. Por isso, os resultados dessa condição são interpretados como evidência de alinhamento a critérios explícitos, e não como prova de superioridade universal.

A avaliação não foi cega quanto ao modelo e à condição de *prompting*. Além disso, as avaliadoras eram autoras do estudo, o que pode influenciar o julgamento por familiaridade com a rubrica, os requisitos e os objetivos da pesquisa. Como mitigação parcial, as justificativas por dimensão foram preservadas, a avaliação inicial foi feita separadamente e as divergências foram adjudicadas explicitamente.

7.2 Validade de Construto

A rubrica busca operacionalizar uma avaliação estruturada de validade e qualidade de artefatos de teste de desempenho, mas há risco de sobreposição entre dimensões. Completude, viabilidade e clareza, por exemplo, podem se influenciar mutuamente. A operacionalização com âncoras busca reduzir esse risco, tornando explícito o que cada dimensão mede.

Também há risco de que D1, D3 e D5 sejam rígidas demais como critérios invalidantes. A decisão é metodologicamente conservadora: um artefato tecnicamente incorreto, associado ao tipo errado de teste ou sem critério numérico de aceitação não oferece base mínima para avaliação de desempenho. Ainda assim, as dimensões ordinais são mantidas para análise diagnóstica, mesmo em artefatos inválidos. Como nenhum dos 36 artefatos falhou nos critérios de validade mínima considerados, o estudo também não fornece evidência suficiente sobre a capacidade de D1, D3 e D5 distinguirem artefatos claramente inválidos. Essa propriedade deverá ser avaliada em corpora que incluam falhas técnicas, tipos de teste inadequados e ausência de critérios mensuráveis.

7.3 Validade Externa

O uso do TPC-W favorece controle e rastreabilidade, mas limita a generalização para sistemas modernos, como arquiteturas de microserviços, aplicações móveis ou sistemas orientados a eventos. O TPC-W deve ser interpretado como domínio experimental controlado, não como representação completa de aplicações contemporâneas.

Os resultados também dependem dos modelos e das configurações escolhidas. Como LLMs podem mudar ao longo do tempo, o estudo preserva os prompts enviados, as respostas completas e os nomes/configurações registrados nos artefatos, seguindo recomendações para estudos empíricos com LLMs [4].

7.4 Validade de Conclusão

Como a amostra é pequena, testes inferenciais não foram utilizados. Por isso, a análise prioriza estatística descritiva, resultados por dimensão, exemplos qualitativos de falhas e medidas de concordância. Conclusões sobre superioridade de uma condição de prompt são formuladas com cautela e vinculadas ao corpus, aos modelos e ao domínio experimental utilizados. Além disso, a utilização de uma única geração por combinação impede estimar a variabilidade intramodelo e verificar se as diferenças observadas permaneceriam estáveis em execuções repetidas.

7.5 Confiabilidade da Avaliação

A avaliação humana pode variar conforme experiência, interpretação da rubrica e familiaridade com testes de desempenho. As justificativas por dimensão, a avaliação inicial independente entre as autoras e a adjudicação de divergências buscam mitigar essa ameaça. A concordância antes da adjudicação é reportada para indicar a confiabilidade inicial de aplicação da rubrica [8, 10].

A concordância inicial mostrou essa ameaça principalmente em D7. A diferença entre considerar um comando sintaticamente executável e considerar todo o procedimento operacionalmente completo afetou a concordância entre avaliadoras. Essa divergência não invalida a rubrica, mas indica que a dimensão de viabilidade de execução precisa de exemplos calibrados e decisões adjudicadas transparentes. A aplicação também não envolveu especialistas externos independentes; portanto, a concordância observada representa a aplicação inicial da rubrica por avaliadoras familiarizadas com sua construção e não estabelece sua confiabilidade entre avaliadores externos.

8 Conclusão

Este artigo apresentou uma rubrica analítica para avaliar cenários e casos de teste de desempenho gerados por LLMs. A rubrica combina dimensões gerais de qualidade de artefatos de teste, como correção, completude e clareza, com dimensões específicas de desempenho, como adequação do tipo de teste, coerência da carga, critérios numéricos de aceitação e viabilidade de execução.

O estudo foi estruturado como uma avaliação empírica exploratória com duas condições de *prompting*: Baseline e Rubric-aware. A avaliação foi conduzida por duas autoras, com aplicação inicial independente da rubrica e sem uso de LLMs como juiz, reduzindo o risco de circularidade entre geração e avaliação.

Após adjudicação, os 36 artefatos foram classificados como válidos segundo os critérios invalidantes D1, D3 e D5. A condição Rubric-aware apresentou aumento descritivo modesto na pontuação ordinal adjudicada, mas esse aumento deve ser interpretado como evidência de maior alinhamento à rubrica, não como prova de superioridade geral dos modelos. A principal limitação observada concentrou-se em D7, viabilidade de execução, especialmente quando os artefatos dependiam de logs, pós-processamento, cálculo de métricas específicas ou substituição de parâmetros.

Esses resultados reforçam que a avaliação de artefatos de teste de desempenho gerados por LLMs não deve se limitar à fluência textual ou à presença de comandos plausíveis. A decomposição por dimensões permitiu distinguir validade mínima, clareza textual e executabilidade efetiva, oferecendo uma base mais precisa para revisão humana de cenários e casos de teste de desempenho.

Como trabalhos futuros, pretende-se ampliar a avaliação para um conjunto maior e mais diverso de requisitos, sistemas e modelos de LLMs, incluindo aplicações web contemporâneas, arquiteturas baseadas em microserviços e diferentes ferramentas de geração de carga. Também se pretende avaliar múltiplas gerações por combinação e incluir artefatos inválidos ou casos de contraste, de modo a examinar a variabilidade das respostas e a capacidade discriminativa dos critérios invalidantes. Novas rodadas de aplicação com especialistas externos independentes são necessárias, especialmente para a dimensão de viabilidade de execução, que concentrou as maiores divergências neste estudo. Outra direção relevante consiste em complementar a avaliação dos artefatos com sua execução prática, permitindo comparar a qualidade prevista pela rubrica com resultados observados em ambiente de teste. Dessa forma, espera-se evoluir a rubrica de um instrumento diagnóstico inicial para um apoio mais robusto à revisão humana de artefatos de teste de desempenho gerados por LLMs.

Disponibilidade de Artefatos

O pacote de replicação deste estudo está disponível publicamente no Zenodo sob a licença CC BY 4.0: https://doi.org/10.5281/zenodo.21121958. O pacote inclui requisitos, rubrica, prompts enviados, contexto técnico, saídas completas, avaliações individuais, decisões

adjudicadas e relatórios de apoio. Ele não inclui saídas medidas de execução do RBE, pois o estudo avalia a qualidade dos artefatos gerados, e não o desempenho em tempo de execução da implementação TPC-W.

Agradecimentos

Os autores agradecem à UNIPAMPA pelo apoio durante o desenvolvimento deste trabalho. Maicon Bernardino é financiado pelo CNPq (Bolsa nº 307969/2026-6).

Referências

[1] Waleed Abdeen, Xingru Chen, and Michael Unterkalmsteiner. 2023. An approach for performance requirements verification and test environments generation. *Requirements Engineering* 28, 1 (2023), 117–144.

[2] Sabinakhon Akbarova, Felix Dobslaw, and Robert Feldt. 2026. Understanding on the Edge: LLM-generated Boundary Test Explanations. arXiv:2601.22791 [cs.SE] https://arxiv.org/abs/2601.22791

[3] Chetan Arora, Tomas Herda, and Verena Homm. 2024. Generating Test Scenarios from NL Requirements using Retrieval-Augmented LLMs: An Industrial Study. In *Proceedings of the 2024 IEEE/ACM 32nd International Requirements Engineering Conference (RE)*. IEEE, 240–251.

[4] Sebastian Baltes, Florian Angermeir, Chetan Arora, Marvin Muñoz Barón, Chunyang Chen, Lukas Böhme, Fabio Calefato, Neil Ernst, Davide Falessi, Brian Fitzgerald, Davide Fucci, Junda He, Christoph Treude, Marcos Kalinowski, Stefano Lambiase, Daniel Russo, Mircea Lungu, Cristina Martinez Montes, Lutz Prechelt, Paul Ralph, Rijnard van Tonder, and Stefan Wagner. 2025. Guidelines for Empirical Studies in Software Engineering involving Large Language Models. arXiv:2508.15503 [cs.SE] doi:10.48550/arXiv.2508.15503

[5] Nathalia Barbosa, Paulo Borba, and Leuson Da Silva. 2025. Detecção de Conflitos Semânticos com Testes Gerados por LLM. In *Anais do X SAST (SAST 2025)*. Sociedade Brasileira de Computaç ao, Porto Alegre, RS, Brasil, 18–27.

[6] Ermanno Battista, Sergio Di Martino, Sergio Di Meglio, Fabio Scippacercola, and Luigi Libero Lucio Starace. 2023. E2E-Loader: A Framework to Support Performance Testing of Web Applications. In *Proceedings of the 2023 IEEE Conference on Software Testing, Verification and Validation (ICST)*. IEEE, 351–361. doi:10.1109/ICST57152.2023.00040

[7] Shreya Bhatia, Tarushi Gandhi, Dhruv Kumar, and Pankaj Jalote. 2024. System Test Case Design from Requirements Specifications: Insights and Challenges of Using ChatGPT. arXiv:2412.03693 [cs.SE] https://arxiv.org/abs/2412.03693

[8] Jessica Díaz, Jorge Pérez, Carolina Gallardo, and Ángel González-Prieto. 2023. Applying Inter-Rater Reliability and Agreement in collaborative Grounded Theory studies in software engineering. *Journal of Systems and Software* 195 (2023), 111520.

[9] Nayse Fagundes and Leopoldo Teixeira. 2025. Evaluating LLMs for Multimodal GUI Test Generation in Android Applications. In *Anais do X Simpósio Brasileiro de Testes de Software Sistemático e Automatizado (SAST 2025)*. Sociedade Brasileira de Computaç ao, Porto Alegre, RS, Brasil, 28–36.

[10] Daniel Graziotin, Per Lenberg, Robert Feldt, and Stefan Wagner. 2021. Psychometrics in Behavioral Software Engineering: A Methodological Introduction with Guidelines. *ACM Transactions on Software Engineering and Methodology* 31, 1 (2021), 1–36.

[11] Navid Bin Hasan, Md. Ashraful Islam, Junaed Younus Khan, Sanjida Senjik, and Anindya Iqbal. 2025. Automatic High-Level Test Case Generation using Large Language Models. In *Proceedings of the 2025 IEEE/ACM 22nd International Conference on Mining Software Repositories (MSR)*. IEEE Computer Society, Los Alamitos, CA, USA, 674–685.

[12] Benedikt Hauptmann, Maximilian Junker, Sebastian Eder, Lars Heinemann, Rudolf Vaas, and Peter Braun. 2013. Hunting for Smells in Natural Language Tests. In *2013 35th International Conference on Software Engineering (ICSE)*. IEEE, 1217–1220.

[13] International Software Testing Qualifications Board. 2018. *Certified Tester Foundation Level Specialist: Performance Testing Syllabus.* https://www.istqb.org/certifications/certified-tester-performance-testing-ct-pt/ Acessado em: 20 maio 2026.

[14] Feng Lin, Dong Jae Kim, Zhenhao Li, Jinqiu Yang, and Tse-Hsun Chen. 2025. RobuNFR: Evaluating the Robustness of Large Language Models on Non-Functional Requirements Aware Code Generation. arXiv:2503.22851 [cs.SE] https://arxiv.org/abs/2503.22851

[15] Satoshi Masuda, Satoshi Kouzawa, Kyousuke Sezai, Hidetoshi Suhara, Yasuaki Hiruta, and Kunihiro Kudou. 2025. Generating High-Level Test Cases from Requirements using LLM: An Industry Study. arXiv:2510.03641 [cs.SE] https://arxiv.org/abs/2510.03641

[16] Evin Aslan Oğuz and Jochen Malte Kuester. 2024. A Comparative Analysis of ChatGPT-Generated and Human-Written Use Case Descriptions. In *Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems (MODELS 2024) (MODELS Companion '24)*. ACM, New York, NY, USA, 533–540.

[17] Ruwei Pan, Yakun Zhang, Qingyuan Liang, Yueheng Zhu, Chao Liu, Lu Zhang, and Hongyu Zhang. 2026. Toward Functional and Non-Functional Evaluation of Application-Level Code Generation. arXiv:2602.03462 [cs.SE] https://arxiv.org/abs/2602.03462

[18] John Pangas, Suhaib Mujahid, Ahmad Abdellatif, and Marco Castelluccio. 2026. Using LLMs to Bridge the Gaps in QA Test Plans at Firefox. *IEEE Software* 43, 1 (2026), 41–47.

[19] Delip Rao and Chris Callison-Burch. 2026. Autorubric: Unifying Rubric-based LLM Evaluation. arXiv:2603.00077 [cs.CL] https://arxiv.org/abs/2603.00077

[20] Amila Rathnayake, Mojtaba Shahin, and Golnoush Abaei. 2026. Behaviour Driven Development Scenario Generation with Large Language Models. arXiv:2603.04729 [cs.SE] https://arxiv.org/abs/2603.04729

[21] Henning Schulz, Dušan Okanović, André van Hoorn, Vincenzo Ferme, and Cesare Pautasso. 2019. Behavior-driven Load Testing Using Contextual Knowledge—Approach and Experiences. In *Proceedings of the 2019 ACM/SPEC International Conference on Performance Engineering*. ACM, 265–272.

[22] Henning Schulz, Dušan Okanović, André van Hoorn, and Petr Tůma. 2021. Context-tailored Workload Model Generation for Continuous Representative Load Testing. In *Proceedings of the 2021 ACM/SPEC International Conference on Performance Engineering*. ACM, 21–32.

[23] Wayne D. Smith. 2000. *TPC–W: Benchmarking an Ecommerce Solution.* Technical Report Revision 1.2. Transaction Processing Performance Council. https://www.tpc.org/tpcw/tpc-w_wh.pdf

[24] Elvys Soares, Manoel Aranda, Naelson Oliveira, Márcio Ribeiro, Rohit Gheyi, Emerson Souza, Ivan Machado, André Santos, Baldoino Fonseca, and Rodrigo Bonifácio. 2023. Manual Tests Do Smell! Cataloging and Identifying Natural Language Test Smells. In *2023 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*. IEEE, 1–11.

[25] Xin Sun, Daniel Ståhl, Kristian Sandahl, and Christoph Kessler. 2026. Quality Assurance of LLM-generated Code: Addressing Non-functional Quality Characteristics. *Journal of Systems and Software* 238 (2026), 112885.

[26] Irdina Wanda Syahputri, Eko K. Budiardjo, and Panca O. Hadi Putra. 2025. Unlocking the Potential of the Prompt Engineering Paradigm in Software Engineering: A Systematic Literature Review. *AI (Basel)* 6, 9 (2025), 206.

[27] Huynh Khanh Vi Tran, Nauman bin Ali, Michael Unterkalmsteiner, Jürgen Börstler, and Panagiota Chatzipetrou. 2025. Quality Attributes of Test Cases and Test Suites – Importance & Challenges from Practitioners' Perspectives. *Software Quality Journal* 33 (2025), 123–160.

[28] Muhammad Usman, Nauman Bin Ali, and Claes Wohlin. 2023. A Quality Assessment Instrument for Systematic Literature Reviews in Software Engineering. *e-Informatica Software Engineering Journal* 17, 1 (2023), 230105.

[29] Christian Vögele, André van Hoorn, Eike Schulz, Wilhelm Hasselbring, and Helmut Krcmar. 2018. WESSBAS: extraction of probabilistic workload specifications for load testing and performance prediction—a model-driven approach for session-based application systems. *Software and Systems Modeling* 17, 2 (2018), 443–477.

[30] Shihai Wang and Tao Chen. 2025. Light over Heavy: Automated Performance Requirements Quantification with Linguistic Inducement. arXiv:2511.03421 [cs.SE] https://arxiv.org/abs/2511.03421

[31] Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C. Schmidt. 2023. A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT. arXiv:2302.11382 [cs.SE] https://arxiv.org/abs/2302.11382

[32] Zhiyi Xue, Liangguo Li, Senyue Tian, Xiaohong Chen, Pingping Li, Liangyu Chen, Tingting Jiang, and Min Zhang. 2024. LLM4Fin: Fully Automating LLM-Powered Test Case Generation for FinTech Software Acceptance Testing. In *Proceedings of the 33rd ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA 2024) (ISSTA '24)*. ACM, New York, NY, USA, 1643–1655.