Comparing Large Language Models on Scrum Certification-Style Questions: Accuracy, Stability, and Error Patterns

Robson Alves Vilar
VIRTUS Research, Development and Innovation Center, Federal University of Campina Grande, Brazil
robson.vilar@virtus.ufcg.edu.br

Emanuel Dantas Filho
VIRTUS Research, Development and Innovation Center, Federal University of Campina Grande, Brazil
emanuel.dantas@virtus.ufcg.edu.br

Ademar França de Sousa Neto
VIRTUS Research, Development and Innovation Center, Federal University of Campina Grande, Brazil
ademar.sousa@virtus.ufcg.edu.br

Mirko Perkusich
VIRTUS Research, Development and Innovation Center, Federal University of Campina Grande, Brazil
mirko@virtus.ufcg.edu.br

Danyllo Wagner Albuquerque
VIRTUS Research, Development and Innovation Center, Federal University of Campina Grande, Brazil
danyllo.albuquerque@virtus.ufcg.edu.br

João Paiva
VIRTUS Research, Development and Innovation Center, Federal University of Campina Grande, Brazil
joao.paiva@virtus.ufcg.edu.br

Kyller Gorgônio
VIRTUS Research, Development and Innovation Center, Federal University of Campina Grande, Brazil
kyller@virtus.ufcg.edu.br

Angelo Perkusich
VIRTUS Research, Development and Innovation Center, Federal University of Campina Grande, Brazil
perkusich@virtus.ufcg.edu.br

Abstract

Large Language Models (LLMs) are increasingly used in exam- and certification-style question answering tasks, where their ability to retrieve, interpret, and apply domain-specific knowledge can be systematically assessed. In Software Engineering, such settings are particularly relevant when questions depend on strict adherence to normative definitions, roles, artifacts, and rules. This paper evaluates the performance of three contemporary LLMs, *GPT-5 mini*, *Gemini 3 Flash*, and *DeepSeek Chat 3.2*, in answering 993 Scrum certification-style questions aligned with the Professional Scrum Master I (PSM I) assessment format. We evaluated the models under three prompting strategies (*zero-shot*, *chain-of-thought*, and *source-grounded*), with repeated executions to assess intra-model stability. We also analyzed performance across Scrum topics and question formats, complemented by a qualitative analysis of recurring error patterns in incorrect answers. Results revealed clear differences among models, with Gemini 3 Flash achieving the highest accuracy, followed by GPT-5 mini and DeepSeek Chat 3.2, while intra-model variability remained low across all conditions. By question format, the models achieved the highest accuracy on single-answer multiple-choice items, whereas multi-select and True/False questions were more error-prone. By topic, performance was more consistent in normatively explicit areas such as Artifacts, Empiricism, and Product Value, but more fragile in Scrum Values, Self-Managing Teams, and Stakeholders & Customers. The qualitative analysis showed that errors were systematic rather than random, involving overgeneralization, restrictive wording, compound distractors, and conflicts between common market interpretations and strict Scrum definitions.

Keywords

Large Language Models, Scrum, Software Engineering, Empirical Evaluation, Knowledge Assessment, Error Analysis

1 Introduction

Agile Software Development has become a dominant approach to building and evolving software systems across industries. Its principles of collaboration, adaptation, and continuous improvement have shaped how software teams organize work, deliver value, and respond to change [14]. As Agile adoption has grown, so has the demand for education and certification mechanisms that support a shared understanding of Agile values, frameworks, and practices [5]. Scrum is a prominent example of this movement. Certifications such as the *Professional Scrum Master* (PSM) and *Professional Scrum Product Owner* are widely used to assess practitioners' understanding of the Scrum framework and its underlying principles [13].

Scrum is particularly well-suited to certification-style assessment because it is grounded in a concise yet normative body of knowledge. The *Scrum Guide* (2020) specifies accountabilities, events, artifacts, commitments, and rules that practitioners are expected to interpret consistently. As a result, Scrum certification-style questions provide a rigorous and measurable setting for assessing domain-specific knowledge whose correctness depends not only on semantic plausibility but also on strict adherence to normative definitions, role boundaries, and rule-based terminology.

In parallel, Large Language Models (LLMs) have been increasingly investigated as question-answering systems in exam- and certification-style settings, where their ability to retrieve, interpret, and apply domain-specific knowledge can be assessed in a controlled manner [9, 22]. Recent studies have examined LLM performance in real assessment contexts, including Brazilian exams such as ENADE, ENEM, and POSCOMP, reporting promising results in question answering and exam-solving tasks [11, 15, 22]. However, strong performance in broad assessment settings does not necessarily imply reliability in domains governed by strict definitions and prescriptive rules. LLMs may still produce plausible but

incorrect answers, and their behavior can be sensitive to prompt formulation, raising concerns for certification-style tasks that require precise adherence to domain-specific terminology and normative constraints [18, 20].

These findings point to the need for investigating LLMs in Software Engineering assessment settings that combine domain specificity, normative precision, and objective scoring, particularly in rule-based certification-style tasks where correctness depends on distinguishing closely related concepts, accountabilities, and normative constraints. This paper addresses that gap through an empirical study of three LLMs (*GPT-5 mini*, *Gemini 3 Flash*, and *DeepSeek Chat 3.2*) on Scrum certification-style questions aligned with the Professional Scrum Master I (PSM I) assessment format. Using a dataset of 993 English questions, we compare model performance across three prompting strategies (*zero-shot*, *chain-of-thought*, and *source-grounded*), analyze variability across repeated executions, and examine how performance varies by Scrum topic and question format. Beyond aggregate accuracy and single-model comparisons, the study includes a qualitative examination of incorrect answers to identify recurring error patterns and distinguish failures associated with model limitations from those related to ambiguity or underspecification in the questions themselves. The main contributions of this paper are:

(1) *A multi-model empirical comparison of contemporary LLMs* on Scrum certification-style questions, considering accuracy and performance differences across models and prompting strategies.
(2) *An analysis of response stability and task sensitivity*, examining intra-model variability across repeated executions and performance differences by Scrum topic and question format.
(3) *A qualitative characterization of error patterns* that helps explain when failures are associated with model limitations and when ambiguous or weakly specified questions influence incorrect answers.
(4) *Empirical implications for evaluating LLMs in normative Software Engineering domains*, showing why aggregate accuracy alone is insufficient and should be complemented by analyses of stability, task characteristics, and recurring failure modes.

The remainder of this paper is organized as follows. Section 2 presents the background on Scrum as a normative Software Engineering domain, LLMs in Agile and Scrum contexts, and LLMs in exam- and certification-style question answering. Section 3 presents the method and study protocol. Section 4 reports the quantitative and qualitative results. Section 5 discusses the implications of these findings for evaluating LLMs in Software Engineering, Scrum certification-style assessments, and future research. Section 6 outlines threats to validity, and Section 7 concludes with final remarks and future work.

2 Background

This section presents the conceptual and empirical background underlying our study. We first discuss Scrum and its certification ecosystem, then review the use of LLMs in Agile and Scrum contexts, and finally summarize evidence on LLMs in exam- and certification-style assessment settings. Together, these areas establish the empirical and conceptual basis for investigating how reliably LLMs answer Scrum certification-style questions.

Scrum and Agile Certification. Scrum is one of the most widely adopted Agile frameworks, emphasizing iterative delivery, self-managing teams, inspection, adaptation, and continuous improvement [14]. Its values, rules, and certification ecosystem are associated with official reference materials such as the *Scrum Guide*, as well as professional certifications including PSM and the Professional Scrum Product Owner (PSPO) [19]. Because Scrum has a prescriptive and normative structure, it provides a shared vocabulary and explicit rules that support consistent adoption of Agile practices across teams and organizations [14]. Scrum certification exams typically rely on multiple-choice questions derived from official materials, offering a controlled setting for assessing knowledge accuracy. The PSM I assessment, in particular, is an introductory certification composed of approximately 80 text-based questions to be completed in 60 minutes, with a minimum score of 85% required to pass. These characteristics make Scrum certification-style questions an appropriate empirical context for examining how automated systems, including LLMs, interpret and reproduce canonical Agile knowledge [21].

LLMs in Agile and Scrum Contexts. Recent studies have applied LLMs to communication, documentation, automation, and decision-making in Agile development. These applications include improving user-story quality and supporting Agile events [2, 25], coordinating team and scaled-Agile activities through LLM-based agents [3, 4], and assisting coding, project management, and model-driven Scrum activities [7, 12, 17]. However, this literature primarily emphasizes productivity and process support, leaving less explored whether LLMs can accurately reproduce the normative knowledge codified in the *Scrum Guide*.

LLMs in Educational and Certification Assessments. A third relevant body of work examines LLM performance in formal educational, professional, and certification-style assessments beyond software development and Agile practice. Studies in medicine, business, academic admissions, and computing indicate that modern LLMs can achieve strong results in well-structured, text-based exams. However, their performance is less reliable in the presence of ambiguity, procedural reasoning demands, or visual prompts. For instance, Gerard et al. [8] reported medical-exam performance at or above student level for leading models. In contrast, Ashrafimoghari et al. [1] found that LLMs performed above typical business-school candidates on the GMAT. In computing, Viegas et al. [22] showed that several LLMs outperformed human candidates on text-based POSCOMP questions, while still struggling with visual items. Inoshita et al. [9] further demonstrated that prompt techniques can significantly improve accuracy in real estate certification tasks. These findings suggest that LLMs have substantial potential in assessment-like settings, but also reinforce the need to evaluate them in controlled, domain-specific contexts where correctness depends on precise interpretation of rules and terminology.

Despite this progress, to the best of our knowledge, no prior study has systematically evaluated or compared LLMs specifically in Scrum question-answering and Agile certification contexts. Prior

work on LLMs in Agile settings has mainly focused on productivity, collaboration, artifact improvement, and workflow automation, rather than on the models' ability to reproduce canonical Scrum knowledge under exam-like conditions. Similarly, studies on LLMs in educational and professional assessments rarely address Software Engineering domains governed by intentionally prescriptive rules and terminology. This is particularly important because, in Scrum certification-style questions, a plausible answer may still be incorrect if it conflicts with the normative definitions of the *Scrum Guide*, ignores restrictive wording, or overgeneralizes a specific accountability.

Building on this context, our study evaluates three contemporary LLMs on 993 Scrum certification-style questions aligned with the PSM I assessment format. Rather than focusing only on aggregate accuracy or on a single model configuration, we analyze performance across models, prompting strategies, repeated executions, Scrum topics, and question formats, and we complement the quantitative results with a qualitative analysis of recurring error patterns. This allows us to provide a more fine-grained view of LLM reliability in a normative Software Engineering domain.

To further clarify how this study positions itself relative to the existing literature, Table 1 provides a comparative summary. While previous studies have extensively explored LLMs for Agile workflow support or general educational assessments, our study bridges these areas by systematically evaluating LLMs specifically in a normative Software Engineering certification context.

Table 1: Comparative summary of related work and this study's contribution.

Research Context	Primary Focus / Contribution
LLMs in Agile and Scrum (e.g., [2, 3, 25])	Workflow automation, productivity, collaboration, and artifact generation in Agile teams.
LLMs in General Assessments (e.g., [1, 8, 22])	Model accuracy in broad educational, business, and general computing exams (e.g., POSCOMP, GMAT).
This Study	LLM accuracy, stability, and error patterns in *normative* Agile certification assessments (Scrum PSM I).

3 Method

This section describes the methodological design used to evaluate LLM behavior in Scrum-style question-answering. The study evaluates not only whether LLMs select correct answers, but also whether their behavior remains stable across repeated executions, how performance varies across question characteristics, and which recurring error patterns emerge when their answers diverge from the gold standard. To this end, the method combines controlled model execution, multiple prompting strategies, repeated runs, topic and format-based analyses, and a qualitative examination of incorrect answers.

3.1 Research Questions

The study is guided by three research questions that progressively address overall model performance, task sensitivity, and qualitative error patterns.

Research Questions

RQ1: How do different LLMs perform when answering Scrum certification-style questions?
RQ1.1: What performance differences can be observed among models?
RQ1.2: Is there intra-model variability across repeated executions?
RQ2: How do question topic and question format influence LLM performance?
RQ2.1: What performance differences can be observed across Scrum topics?
RQ2.2: What performance differences can be observed across question formats?
RQ3: What error patterns can be observed in LLM answers to Scrum certification-style questions?
RQ3.1: Which errors are more strongly associated with intrinsic model limitations?
RQ3.2: Which errors are associated with ambiguity or underspecification in the questions?

Together, the research questions examine model performance and stability (RQ1), sensitivity to topic and format (RQ2), and recurring sources of error (RQ3).

3.2 Study Design

We conducted a mixed-method empirical study combining controlled LLM execution, quantitative question-response evaluation, and qualitative error analysis, following the guidelines for empirical Software Engineering research proposed by Wohlin et al. [24] and Ralph et al. [16]. The study comprised 993 Scrum certification-style questions, three prompting strategies, three LLMs, five executions per model–prompt combination, quantitative analyses for RQ1 and RQ2, and a qualitative error analysis for RQ3.

3.2.1 Dataset. The dataset consists of 993 English Scrum certification-style questions obtained from a PSM I preparation question bank maintained by an experienced Scrum training provider. It is important to clarify that this dataset contains certification-style preparation questions, not official exam items. The questions have been used and refined across multiple course editions and were designed to reflect the format and content emphasis of the Professional Scrum Master I (PSM I) assessment. They cover the main Scrum knowledge areas defined in the *Scrum Guide* (2020). Each question includes a textual stem, a set of alternatives, a Scrum topic category, and a validated answer key.

The dataset comprises three formats commonly used in certification style assessments: True/False questions (249 items), single-answer multiple choice questions (591 items), and multi-select questions with multiple correct options (153 items). All questions were anonymized and stored using a minimal schema (`question`, `category`, `options`, `gold_set`). The anonymized dataset schema

and the experimental outputs are available as supplementary material. The release of the full question text follows the usage permissions associated with the original question bank.

The full dataset was used for the overall model comparison and repeated-execution analysis associated with RQ1.1 and RQ1.2. For the finer-grained analyses in RQ2, we adopted task-specific balanced subsamples to reduce category imbalance and enable fairer comparisons. Throughout the analyses, the three prompting strategies are abbreviated as ZS (zero-shot), CoT (chain-of-thought), and SG (source-grounded). For topic-based analysis, we used a balanced sample of 104 questions, with eight questions for each of 13 Scrum subject areas: Artifacts, Coaching and Mentoring, Done, Empiricism, Events, Facilitation, Forecasting and Release Planning, Product Backlog Management, Product Value, Scrum Team, Scrum Values, Self-Managing Teams, and Stakeholders & Customers. For question-format analysis, we used a balanced sample of 96 questions, with 32 questions per format category.

3.2.2 Prompting Strategies. We evaluated three prompting strategies that reflect different levels of instruction and source grounding:

- *Zero-shot*: the question and its alternatives are presented without additional guidance, serving as a baseline for direct answer generation.
- *Chain-of-Thought*: the model is instructed to reason briefly before stating the final answer, encouraging explicit intermediate justification [23].
- *Source-grounded*: the model is instructed to justify its answer by explicitly grounding it in the *Scrum Guide* (2020), promoting alignment with canonical material [10]. We emphasize that the full text of the *Scrum Guide* was not provided in the prompt context; instead, the models were required to rely on their pre-trained parametric knowledge of the document.

These strategies were selected to represent common ways of querying LLMs in question-answering scenarios, ranging from direct answering to more structured and source-aware prompts. The same prompting structure was used for all models within each prompting condition, allowing differences in performance to be interpreted primarily in terms of model behavior rather than prompt variation.

3.2.3 LLMs and Generated Responses. Each question was answered independently using three LLMs. To ensure full reproducibility, we specify the exact model identifiers used in our API requests: *GPT-5 mini* (`gpt-5-mini`), *Gemini 3 Flash* (`gemini-3-flash-preview`), and *DeepSeek Chat 3.2* (`deepseek-chat`). All models were accessed programmatically via their respective commercial APIs (OpenAI, Google GenAI, and DeepSeek) during April 2026. To reduce cross-question contamination, each question was executed in isolation, with no conversational memory between runs. For reproducibility purposes, we emphasize that we utilized the default model parameters provided by the APIs at the time of execution (e.g., default temperature and token limits), and no explicit seed configurations were applied.

To investigate response stability, each model–prompt combination was executed five times over the full dataset. This repeated-execution design enabled the analysis of intra-model variability and reduced the risk of drawing conclusions from a single stochastic run. The generated responses were then validated for schema conformity and normalized to a canonical representation before scoring.

3.2.4 Quantitative Analyses. For RQ1, we compared model performance using exact-match accuracy, computed by comparing each predicted answer against the gold-standard answer. Exact-match scoring was also applied to multi-select questions, meaning that a response was considered correct only when the predicted set matched the complete gold-standard set. Average accuracy was computed across the five repeated executions for each model–prompt combination.

To assess the statistical significance of the observed differences, we applied non-parametric tests suitable for paired nominal data. We used Cochran's Q test to evaluate variance across the five repeated runs and across the three prompting strategies. For pairwise model comparisons, we applied McNemar's test with continuity correction, adjusted via the Bonferroni method to account for multiple comparisons (significance threshold set at $p < 0.0167$).

To assess RQ1.2, we analyzed intra-model variability across repeated executions using descriptive measures, including standard deviation, minimum and maximum observed accuracy, and run-to-run amplitude. This analysis allowed us to examine whether model performance remained stable across repeated runs under the same experimental conditions.

For RQ2, we analyzed performance by Scrum topic and question format using the balanced subsamples described above. The topic-level analysis examined how accuracy varied across the 13 Scrum subject areas, while the format-level analysis compared True/False, single-answer multiple-choice, and multi-select questions. These analyses were descriptive and comparative, aiming to identify systematic strengths and weaknesses associated with task characteristics.

3.2.5 Qualitative Analyses. For RQ3, we complemented the quantitative results with a qualitative analysis of selected incorrect answers, following qualitative synthesis practices commonly used in empirical Software Engineering [6]. We first inspected disagreement patterns across the full dataset, specifically filtering for the most severe failure scenario: questions in which all three models failed across all prompting conditions. This strict filtering criteria yielded exactly four questions in the entire dataset. Consequently, rather than selecting an arbitrary sample, we conducted an exhaustive qualitative analysis of these four universal failure cases.

The qualitative analysis considered two broad categories of error: (*i*) errors more strongly associated with intrinsic model limitations, such as overgeneralization, conceptual confusion, difficulty handling restrictive wording, or difficulty managing multiple correct answers; and (*ii*) errors associated with ambiguity, underspecification, or multiple plausible interpretations in the question itself.

The analysis was conducted collaboratively by the two main authors, who reviewed and discussed the selected cases to interpret the likely sources of error. When discrepancies suggested possible issues with the gold standard, the interpretations were further checked with the dataset provider, who served as an additional source to clarify item intent and answer-key consistency. This procedure allowed us to examine not only whether models answered

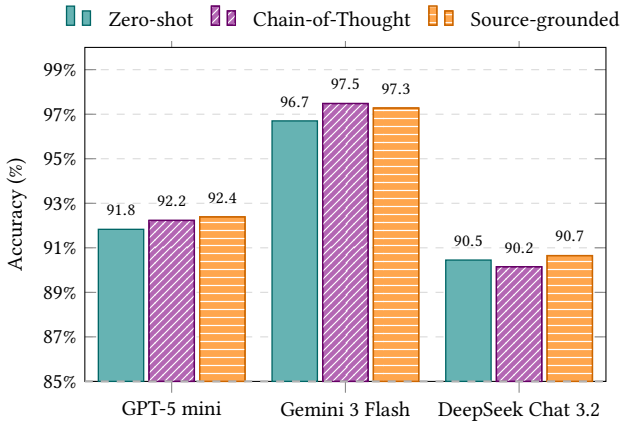


Figure 1: Average accuracy (%) by model and prompting strategy across 993 questions. The dashed line indicates the PSM I passing threshold (85%).

incorrectly, but also what kinds of failure patterns emerged in a normative Software Engineering question-answering task.

4 Results and Discussion

This section presents the results of our empirical study of LLM behavior in Scrum-style question-answering. We organize the discussion around three dimensions aligned with the research questions: overall model performance and response stability (RQ1), the influence of Scrum topic and question format (RQ2), and qualitative error patterns in incorrect answers (RQ3). We also discuss what these findings indicate about the evaluation of LLMs in normative Software Engineering knowledge tasks.

4.1 Accuracy Across Models and Prompts (RQ1.1)

Figure 1 presents the average accuracy of the three evaluated LLMs under the three prompting strategies. Overall performance was high across the 993 questions in the dataset, with all models exceeding the PSM I passing threshold. At the same time, differences among models were clear and consistent across prompting conditions.

Gemini 3 Flash achieved the highest accuracy across all prompting strategies. A pairwise McNemar's test with Bonferroni correction confirmed that the superiority of Gemini 3 Flash over both GPT-5 mini and DeepSeek Chat 3.2 is statistically significant ($p < 0.001$). However, the difference between GPT-5 mini and DeepSeek Chat 3.2 was not statistically significant ($p > 0.05$). Furthermore, Cochran's Q test revealed no statistically significant differences among the three prompting strategies within any of the models ($p > 0.05$). In practical terms, this robustly indicates that model choice had a definitive effect on answer accuracy, whereas prompt choice did not yield statistically significant variations in this Scrum certification-style question-answering task.

Prompting produced smaller effects than model selection. GPT-5 mini improved slightly from zero-shot to the more structured prompts, reaching its best result under the source-grounded condition (92.39%). Gemini 3 Flash also benefited from structured prompting, peaking under Chain-of-Thought (97.48%), while remaining very close under source-grounded prompting (97.28%). DeepSeek Chat 3.2 showed only modest variation, with its best result under source-grounded prompting (90.65%). Overall, these results suggest that prompt engineering can generate incremental improvements, but does not eliminate underlying differences in model capability.

These differences are relevant because high aggregate performance can obscure substantial variation among models. Although all three models exceeded the 85% PSM I passing threshold, the gap between approximately 97% and 90% accuracy remains practically meaningful. Across a dataset of nearly one thousand questions, this difference corresponds to dozens of additional incorrect answers, reinforcing the need to analyze model-specific behavior rather than treating contemporary LLMs as interchangeable systems.

Structured prompting produced modest gains in some conditions, especially when prompts encouraged reasoning or grounding in the *Scrum Guide*. However, these gains did not substantially change the overall ranking of models, suggesting that prompt design improved performance incrementally but did not compensate for differences in model capability.

4.2 Intra-Model Variability Across Repeated Executions (RQ1.2)

To assess response stability, each model–prompt combination was executed five times over the full dataset. Overall, intra-model variability was low across all evaluated conditions, with standard deviations ranging from 0.100 to 0.583 percentage points and run-to-run ranges from 0.200 to 1.600 percentage points. This stability was statistically confirmed by Cochran's Q test, which detected no significant differences across the five runs for any model–prompt combination ($p > 0.05$).

Gemini 3 Flash showed the lowest variability overall, particularly in the zero-shot condition (standard deviation = 0.100; range = 0.200), indicating highly stable behavior across repeated runs. DeepSeek Chat 3.2 also remained stable across conditions, with a standard deviation ranging from 0.167 to 0.342 and a range from 0.400 to 0.800. GPT-5 mini showed slightly higher variation, especially under source-grounded prompting (standard deviation = 0.583; range = 1.600), although even this fluctuation remained small in absolute terms.

These results indicate that repeated executions did not substantially alter the study's overall conclusions. The ranking of models remained stable across runs, and no condition exhibited large oscillations that would call the main findings into question. This strengthens the interpretation that the observed differences among models and prompting strategies are not artifacts of isolated stochastic executions.

At the same time, low variability should not be confused with correctness. A model may behave consistently while still being systematically wrong in specific cases, such as misinterpreting restrictive wording, failing on particular question formats, or converging on plausible but incorrect answers in ambiguous items. Thus, response stability should be understood as one dimension of model behavior, not as evidence of uniform accuracy. In this study,

the low intra-model variability strengthens the credibility of the subsequent analyses: the performance differences observed across Scrum topics (RQ2.1), question formats (RQ2.2), and recurring error patterns (RQ3) are unlikely to result from isolated run-level fluctuations. This means that the weaknesses identified in later analyses (such as fragility in *Stakeholders & Customers* or sensitivity to restrictive wording) reflect stable behavioral tendencies rather than stochastic artifacts.

4.3 Performance by Scrum Topic (RQ2.1)

Table 2 presents the topic-level accuracy results across the three evaluated models and prompting strategies. This analysis used a balanced topic sample of 104 questions (eight per Scrum subject area), allowing fair comparisons across topics.

The results show that LLM performance varies substantially across Scrum topics. Six topics, *Artifacts*, *Empiricism*, *Product Value*, *Coaching and Mentoring*, *Facilitation*, and *Forecasting and Release Planning*, reached perfect or near-perfect accuracy across all models and prompting strategies, indicating that the evaluated models handled questions consistently when answers depended on direct recognition of canonical Scrum concepts. The complete accuracy results for all thirteen topics are available in the supplementary material file `Model_accuracy_by_Scrum_topic.xlsx`. Table 2 focuses on the remaining topics, where performance was lower or more variable.

In contrast, lower and more variable performance was observed in topics that require finer distinctions among responsibilities, values, or stakeholder relationships. The clearest case is *Stakeholders & Customers*, which showed low accuracy across all models and prompting strategies. *Self-Managing Teams* also showed high sensitivity to prompting in Gemini 3 Flash, increasing from 33% under zero-shot prompting to 100% under both Chain-of-Thought and source-grounded prompting. *Scrum Values* exhibited another form of topic-level fragility: while Gemini 3 Flash reached 100% across all prompts, GPT-5 mini and DeepSeek Chat 3.2 showed considerably lower results in some configurations.

Prompting effects were uneven across topics. In some cases, structured prompts improved performance, as observed for *Self-Managing Teams* with Gemini 3 Flash and for *Product Backlog Management* with source-grounded prompting. However, these improvements were not consistent across all models or topics. For example, GPT-5 mini performed worse on *Scrum Values* under Chain-of-Thought prompting than under zero-shot prompting, and DeepSeek Chat 3.2 also showed decreases in some topic–prompt combinations. Thus, prompting should not be interpreted as uniformly beneficial; its effect appears to depend on the interaction between model, topic, and question wording.

Overall, the topic-level analysis confirms that high aggregate accuracy can mask localized weaknesses. Even when models perform well on average, they may still be fragile in Scrum areas that require distinguishing between similar accountabilities, interpreting value-oriented concepts, or resolving context-dependent stakeholder relationships. These findings motivate the subsequent analysis by question format and the qualitative examination of recurring error patterns.

4.4 Performance by Question Format (RQ2.2)

To examine the effect of question format, we used the balanced sample of 96 questions described in Section 3.2, with 32 questions per format category. Figure 2 presents the accuracy results for each model across question format and prompting strategy.

Across models and prompts, single-answer multiple-choice questions achieved the highest accuracy, with an aggregated mean of approximately 94.45%. Multi-select questions averaged 88.2%, while True/False questions showed the lowest overall performance, at approximately 87.5%. This pattern indicates that question format influences model performance even when the topic domain and scoring procedure remain fixed.

Model-specific differences remained visible within this general pattern. GPT-5 mini showed its lowest results on multi-select questions, suggesting sensitivity to formats that require identifying a complete set of correct alternatives. Gemini 3 Flash achieved the strongest results across all formats, reaching 100% accuracy on single-answer multiple-choice questions under both Chain-of-Thought and source-grounded prompting. DeepSeek Chat 3.2 performed worst on True/False questions, especially under source-grounded prompting.

These findings show that aggregate accuracy can mask format-specific weaknesses. Single-answer multiple-choice questions appear to be the most favorable format for the evaluated models, whereas multi-select and True/False questions introduce additional difficulty. In multi-select items, the model must avoid both missing correct alternatives and selecting plausible distractors; in True/False items, a single restrictive term or subtle normative distinction can invert the expected answer. Question format is, therefore, a meaningful variable in certification-style evaluations of LLMs, one that aggregate scoring alone cannot capture.

4.5 Errors Stemming from Model Limitations (RQ3.1)

The previous analyses showed that the evaluated LLMs achieved high aggregate accuracy, but also revealed variation across models, topics, and question formats. To better understand these results, this subsection examines recurring error patterns in incorrect answers that are more closely associated with model behavior, specifically, overgeneralization, difficulty handling restrictive wording, and incorrect treatment of compound alternatives. The analysis was conducted collaboratively by the two main authors using traceable question identifiers and explicit analytic categories, with cross-checking by the dataset provider for cases involving possible gold-standard inconsistencies.

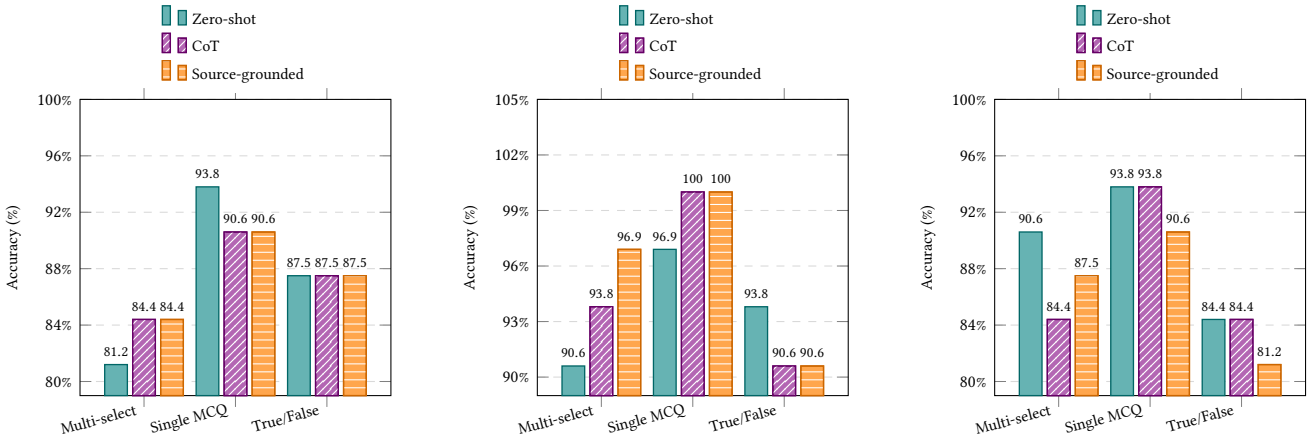
The first group of errors does not resemble hallucination in the narrow sense of inventing facts. Instead, this exhaustive set of four universal failure cases suggests difficulties in handling restrictive wording, strong semantic associations, and compound alternatives. Three questions illustrate these recurring limitations.

The first error pattern involves semantic association overriding modal qualification, as illustrated by Question 280:

> **Question 280 – True/False**
> True or False: The Product Backlog might commit to a Product Goal.
> 1) TRUE

Table 2: Mean accuracy (%) for topics with variable or low performance. Topics reaching perfect or near-perfect accuracy across all models and conditions are omitted. Shaded cells indicate values below 80% (darker = lower).

Topic	Gemini 3 Flash			GPT-5 mini			DeepSeek Chat 3.2		
	ZS	CoT	SG	ZS	CoT	SG	ZS	CoT	SG
Done	100	100	100	88	88	88	88	88	75
Events	95	100	100	100	100	100	100	100	100
Product Backlog Mgmt.	94	94	100	88	100	88	88	88	88
Scrum Team	96	88	88	100	100	100	100	100	100
Scrum Values	100	100	100	75	38	62	100	62	75
Self-Managing Teams	33	100	100	88	75	88	75	88	88
Stakeholders & Customers	50	50	50	62	62	62	62	50	50



(a) GPT-5 mini **(b) Gemini 3 Flash** **(c) DeepSeek Chat 3.2**

Figure 2: Accuracy (%) by question format and prompting strategy across the three evaluated models.

2) FALSE
Correct answer: 2) FALSE

In this scenario, all three models, across all prompt approaches, incorrectly classified the statement as "TRUE". The justification generated by the models, revealed especially in the Chain-of-Thought (CoT) prompts, demonstrates that they successfully retrieved the correlation between the artifacts ("Product Backlog") and their respective commitments ("Product Goal"), a novelty introduced in the 2020 Scrum Guide. However, the error appears to result from insufficient weighting of the modal verb “might” when compared with the strong semantic association among the main terms.

In this case, the models appear to prioritize the strong association between *Product Backlog*, *commitment*, and *Product Goal*, while underweighting the restrictive effect of the modal verb “might”. As a result, they treat a mandatory relationship in Scrum as if it were compatible with mere possibility. The error, therefore, reflects difficulty in handling fine-grained logical qualification, rather than a failure to retrieve the relevant Scrum concepts.

The second pattern involves the influence of common market interpretations over strict normative definitions, as illustrated by Question 675:

Question 675 – True/False
True or False: The Product Owner is the main person responsible for engaging the stakeholders
1) TRUE
2) FALSE
Correct answer: 2) FALSE

Again, all three models selected the incorrect answer. The official answer key indicates that the statement is false, based on the 2020 Scrum Guide, which assigns the responsibility for engaging with stakeholders to the entire Scrum Team, removing the Product Owner from the position of an exclusive or "main" intermediary. However, LLMs are trained on vast corpora of data collected from the internet, which encompass over a decade of agile forums, blogs, old preparatory courses, and practical market discussions. In the "common sense" of the technology market, the Product Owner is, in fact, expected and taught to be the main point of contact with the client and stakeholders.

This case suggests difficulty in prioritizing the strict normative definition of Scrum over widespread informal interpretations of the Product Owner role. Even when the Source-grounded technique was applied to direct the models’ attention to the Scrum Guide roles, the pre-training data’s probabilistic bias persisted. The models used

their text-generation capabilities to create fallacious rationalizations (for example, arguing that, since the PO manages the Backlog, they are, by logical consequence, the primary person responsible for the stakeholders). This illustrates a significant limitation of LLMs in certification exams that require textual rigor: their tendency to favor majority knowledge from the public domain over recent, stringent normative updates.

The third pattern concerns difficulty with compound distractors in multiple-choice questions, as observed in Question 556:

> **Question 556 – Multi-select**
> When should Product Backlog items be refined? (Choose the best two answers.)
> 1. Only in the first Sprint
> 2. During Sprint Planning events
> 3. Throughout the Sprints
> 4. During the Sprint, if they have not been refined in the previous Sprints
> *Correct answer: 2, 3*

In this question, the models consistently opted for options 3 and 4, missing the correct answer. Option 4 is a classic complex distractor, built on a true premise ("During the Sprint") and a false limiting condition ("if they have not been refined in the previous Sprints"). The Scrum Guide is clear in stating that refinement is an ongoing activity that occurs "as needed". This error illustrates the difficulty of evaluating compound propositions in multiple-choice questions. Models tend to approve an alternative if the first clause has high semantic similarity to the reference documents, often failing to invalidate it based on a subsequent restrictive conditional clause. Furthermore, the models rejected option 2 ("During Sprint Planning events") due to a negative association bias: in the Scrum data ecosystem, there is a massive emphasis on teaching that refinement"is an ongoing activity, not a formal event". The term "event" in option 2 may have discouraged the models from recognizing that refinement can also occur during Sprint Planning when needed.

4.6 Errors Associated with Ambiguity and Question Formulation (RQ3.2)

The errors examined in RQ3.1 were primarily associated with how models process and weight information, overgeneralizing semantic associations, underweighting modal qualifiers, or failing to evaluate compound alternatives. RQ3.2 shifts the focus to the questions themselves, examining how ambiguity, underspecification, or competing interpretations in the item formulation can also lead to incorrect answers, even when the model's reasoning is locally coherent. In this group of cases, ambiguity, underspecification, or competing interpretations of the reference material make the expected answer less straightforward. The most prominent example involves a tension between a general Scrum rule and a more specific statement associated with the Sprint Retrospective, as illustrated by Question 61:

> **Question 61 - Multiple choice**
> During the Sprint Retrospective, the Scrum Team members identify the most helpful changes to improve their effectiveness. Where are the most impactful improvements registered? (choose the best answer)
> 1) Next Sprint Backlog
> 2) Issue Tracker
> 3) Product Backlog
> 4) Process Improvement Backlog
> *Correct answer: 3) Product Backlog*

In this case, all models selected option 1 (Next Sprint Backlog). Rather than indicating a straightforward model limitation, this error appears to reflect ambiguity in how the question relates two Scrum Guide statements. The author of the question based the answer key (Product Backlog) on the framework's macro rule: the Product Backlog section unequivocally states that it is the "sole source of work undertaken by the Scrum Team". The logic follows that any improvement requiring work must reside there. However, in the section dedicated to the Sprint Retrospective, the guide establishes an exception or specific rule: "The most impactful improvements are addressed as soon as possible. They may even be added to the Sprint Backlog for the next Sprint."

The formulation of Question 61 appears to have guided the models toward the more context-specific rule. By including the opening phrase "During the Sprint Retrospective...", the prompt activated the attention weights directed at the scope of that specific ceremony in the model's knowledge vector space. As a result, the LLM searched for the rule most semantically proximal to the Retrospective event, retrieving the explicit instruction that authorizes the registry in the "Next Sprint Backlog". In this sense, the models selected a semantically defensible answer, even though it diverged from the adopted gold standard. The question is inherently ambiguous, as it does not specify which hierarchical level of rule (the general law of the Product Backlog or the Retrospective's exception) should be applied. This case shows that some incorrect answers may be linked to ambiguity or underspecification in the question, rather than to a simple failure to retrieve Scrum knowledge.

A second formulation-related pattern involves restrictive linguistic markers. This connects Questions 280 and 675 in terms of question wording. Terms such as "might" and "main" can change the truth value of an otherwise plausible statement, requiring precise attention to the wording of the item and to the terminology of the Scrum Guide.

In Question 675, the inclusion of the restrictive word "main" transforms a premise that is almost true in daily practice into a false statement in the exam environment. Creators of certification tests frequently introduce adverbs of intensity or exclusivity (main, solely, only, strictly) to invalidate statements that are instinctively correct for experienced professionals. This tricky formulation style exploits human cognitive heuristics of speed reading. The results suggest that LLMs can be similarly vulnerable to such wording effects. An ambiguous formulation, or one that uses hyper-specific lexical nuances, acts as a vector of confusion, forcing the model to weigh the semantic probability of the entire sentence against a single pivoting word. Most of the time, as tests consistently demonstrated, the model fails to assign sufficient weight to the restrictive word, leading to incorrect answers.

By exhaustively analyzing the only four questions in the dataset where all models failed across all prompting conditions, a consistent picture emerges: these incorrect answers were not random. They clustered around specific types of difficulty, modal qualification, normative recency, compound distractors, and ambiguous item

formulation that cut across models and prompting strategies. This suggests that the errors reflect structural properties of both the models and the questions, rather than isolated stochastic failures. The analysis of these four cases helps answer RQ3.1 and RQ3.2 by showing that incorrect responses were not random hallucination events, but recurring failure patterns that can be categorized.

Overall, the cases show that incorrect answers resulted from both recurring model limitations and weaknesses in question formulation. This explains how high aggregate accuracy can coexist with systematic failures in specific Scrum certification-style questions.

5 Implications

This study has implications for evaluating LLMs in Software Engineering, for Scrum-style certification assessments, and for future research on domain-specific question-answering tasks.

Implications for LLM Evaluation in Software Engineering The results show that aggregate accuracy alone is insufficient to characterize LLM behavior in normative Software Engineering domains. Evaluations should also consider stability, task sensitivity, and recurring errors, particularly when strict terminology conflicts with widespread informal interpretations or when restrictive markers invalidate otherwise plausible statements. Scrum certification questions therefore assess not only knowledge retrieval but also whether models can subordinate broad semantic associations to rule-based definitions. Model choice had a stronger effect than prompt choice, indicating that structured prompting may provide incremental gains but cannot fully compensate for differences in model capability.

Implications for Scrum Certification-Style Assessment. Scrum certification-style questions provide a useful setting for evaluating LLMs because they combine objective scoring with normative domain knowledge. In this study, the models performed better on topics grounded in explicit terminology and well-delimited concepts, such as Artifacts, Empiricism, and Product Value. In contrast, weaker or more variable performance was observed in topics such as Scrum Values, Self-Managing Teams, and Stakeholders & Customers, where questions often require finer distinctions among responsibilities, values, and stakeholder relationships.

Question format also shaped model performance. Single-answer multiple-choice questions were the most favorable format, whereas multi-select and True/False questions introduced additional difficulty. Multi-select items require the model to identify a complete set of correct alternatives while avoiding plausible distractors. True/False items can be affected by restrictive terms or subtle normative distinctions that invert the expected answer. These findings suggest that certification-style benchmarks should report performance by question format rather than treating all items as equivalent.

Furthermore, the fact that all evaluated models comfortably exceeded the PSM I passing threshold raises practical questions for the certification industry itself. If LLMs can easily pass these text-based assessments, traditional online exams without strict proctoring become highly vulnerable to automated cheating. To maintain their discriminative value, future Agile certifications may need to evolve beyond static multiple-choice questions, incorporating scenario-based reasoning, oral assessments, or interactive simulations where human judgment and practical experience are harder for current LLMs to emulate.

Implications for Question and Benchmark Design. The qualitative analysis showed that incorrect answers were not random. Some errors were associated with model-side limitations, such as overgeneralization, difficulty handling modal verbs, and difficulty evaluating compound alternatives. Other errors were associated with ambiguity, underspecification, or competing interpretations in the question formulation. This distinction is important because an incorrect model answer may reveal not only a model limitation, but also a weakness in the assessment item itself.

For researchers designing benchmarks, these results suggest that question banks should be inspected not only for coverage and answer-key correctness, but also for ambiguity, restrictive wording, and dependence on narrow interpretations of reference material. In normative domains such as Scrum, small linguistic markers can substantially change the expected answer. As a result, qualitative error analysis can complement quantitative scoring by identifying whether failures arise from model behavior, question design, or their interaction.

Implications for Future Research. Future work should evaluate additional models, languages, Agile frameworks, and certification levels; investigate retrieval-based grounding; and apply metamorphic reformulations to distinguish conceptual understanding from sensitivity to wording. Because the qualitative analysis covered only four universal failure cases, broader samples of incorrect answers are also needed to validate the distinction between model limitations and question-design flaws.

Scrum certification-style questions constitute a controlled empirical setting for studying LLM behavior in normative Software Engineering tasks. Such settings enable analysis not only of whether models answer correctly, but also of how stable their answers are, where performance degrades, and which types of errors persist despite high aggregate accuracy.

6 Threats to Validity

Threats to validity are discussed following the classification proposed by Wohlin et al. [24].

Conclusion validity. The quantitative analyses rely on exact-match accuracy computed over balanced subsamples, which limits the precision of topic and format-level comparisons. The balanced samples of 104 and 96 questions were designed to reduce category imbalance. Still, the small number of items per category (eight per topic and 32 per format) means that individual question characteristics can disproportionately influence observed results. The qualitative error analysis necessarily involves interpretive judgment. Furthermore, while focusing exclusively on the four universal failure cases provided a rigorous baseline, this small sample limits the broader generalizability of our qualitative findings. Consequently, the distinction between model errors and question design flaws should be viewed as a strong exploratory finding that requires validation through a larger set of incorrect answers. Although the analysis was conducted collaboratively by the authors using traceable question identifiers and explicit analytic categories, the selection of representative cases and the assignment of errors to categories (model-side vs. question-formulation) involves subjectivity that

cannot be fully eliminated. The interpretations were cross-checked with the dataset provider for cases involving potential gold-standard inconsistencies, thereby reducing but not eliminating this threat.

Construct validity. Model performance was operationalized primarily as answer correctness relative to predefined gold-standard responses. This operationalization is appropriate for certification-style question answering. Still, it does not capture all aspects of model behavior, such as explanation quality, partial correctness, or the plausibility of alternative interpretations. This limitation is especially relevant for multi-select items and for questions involving subtle distinctions among Scrum accountabilities, values, and stakeholder relationships. To mitigate this threat, we complemented the quantitative analysis with a qualitative examination of error patterns, which helped distinguish more objective failures from cases involving ambiguity, underspecification, or plausible alternative interpretations. In addition, the dataset was derived from a PSM I preparation question bank and reviewed by Scrum-certified experts, which increases its alignment with the intended assessment context.

Internal validity. All questions were executed independently, with no conversational memory across runs, to reduce cross-question contamination. Because LLM outputs may vary stochastically, each model–prompt combination was executed five times over the full dataset. The low variability observed across repeated runs reduces the likelihood that the main findings are artifacts of isolated executions. Nevertheless, some threats remain. First, the observed results may have been influenced by hidden model-side changes, service-side variability, or undocumented differences in inference behavior across providers. Furthermore, as stated in Section 3, we utilized the default hyperparameters (e.g., temperature and top-p) provided by each API (OpenAI, Google GenAI, and DeepSeek) during the executions in April 2026. Because these default configurations differ across providers, this introduces a confounding variable regarding whether performance differences stem solely from intrinsic model capabilities or are partially influenced by more optimized default settings, potentially affecting direct comparability. Second, the correctness of the gold-standard answers remains a potential source of error, particularly in questions that admit multiple plausible interpretations or depend on narrow certification-oriented readings. Third, the qualitative error analysis necessarily involves interpretive judgment, even though it was conducted using traceable question identifiers, explicit analytic categories, and discussion among the authors.

External validity. The dataset consists of 993 English Scrum certification-style questions aligned with the *Scrum Guide* (2020), which limits generalization to other Agile frameworks, certification ecosystems, languages, or future revisions of Scrum. Likewise, the evaluated models (*GPT-5 mini*, *Gemini 3 Flash*, and *DeepSeek Chat 3.2*) represent a relevant but limited sample of contemporary LLMs. The dataset is also exclusively in English, which limits generalization to certification-style assessments conducted in other languages, including Portuguese, which is directly relevant to the Brazilian Software Engineering community. LLM performance in normative domains may vary across languages due to differences in training data distribution and the availability of Scrum-related reference material in non-English corpora. Results may differ for other models, providers, prompting configurations, inference settings, or interaction modes. Nevertheless, the study design is replicable and may be adapted to other Software Engineering domains that combine canonical knowledge, professional standards, and certification-style evaluation.

7 Final Remarks

This paper evaluated *GPT-5 mini*, *Gemini 3 Flash*, and *DeepSeek Chat 3.2* on 993 Scrum certification-style questions under three prompting strategies and repeated executions. Gemini 3 Flash achieved the highest accuracy, while all models exceeded the PSM I passing threshold and showed low intra-model variability. Model choice had a stronger effect on performance than prompting strategy.

Despite high aggregate accuracy, performance varied across Scrum topics and question formats. The models were more reliable on normatively explicit topics and single-answer multiple-choice questions, but more fragile on *Scrum Values*, *Self-Managing Teams*, *Stakeholders & Customers*, multi-select items, and True/False questions. The qualitative analysis further identified systematic errors involving restrictive wording, modal qualification, compound alternatives, normative recency, and ambiguous question formulation.

These findings show that aggregate accuracy alone is insufficient to characterize LLM reliability in normative Software Engineering tasks. Future work should evaluate additional models, grounding mechanisms, languages, Agile frameworks, and certification levels, as well as apply metamorphic testing to assess robustness under controlled question reformulations.

Artifact Availability

The primary database, the prompts used in this study, and the resulting datasets containing the Large Language Model (LLM) responses are publicly available at https://github.com/robsonvilar/comparing-llms-on-scrum-certification-sbes-2026.

Acknowledgements

The authors used ChatGPT to support language editing, clarity, readability, and minor textual refinement during manuscript preparation. No AI tool was used to generate scientific content, define the study design, select or screen studies, extract data, perform data analysis, interpret results, or draw conclusions. All AI-assisted text was reviewed, revised, and validated by the authors, who take full responsibility for the content of this manuscript.

This work has been partially funded by the project "*iSOP Base: Investigação e desenvolvimento de base arquitetural e tecnológica da Intelligent Sensing Operating Platform (iSOP)*" supported by CENTRO DE COMPETÊNCIA EMBRAPII VIRTUS EM HARDWARE INTELIGENTE PARA INDÚSTRIA - VIRTUS-CC, with financial resources from the PPI HardwareBR of the MCTI grant number 055/2023, signed with EMBRAPII.

References

[1] Vahid Ashrafimoghari et al. 2024. Evaluating large language models on the GMAT: Implications for the future of business education. *arXiv preprint arXiv:2401.02985* (2024).

[2] Beatriz Cabrero-Daniel, Tomas Herda, Victoria Pichler, and Martin Eder. 2024. Exploring human-ai collaboration in agile: Customised llm meeting assistants. In *International Conference on Agile Software Development*. Springer Nature Switzerland Cham, 163–178.

[3] Jarosław A Chudziak and Konrad Cinkusz. 2024. Towards LLM-augmented multi-agent systems for agile software engineering. In *2024 39th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE, 2476–2477.

[4] Konrad Cinkusz et al. 2024. Cognitive agents powered by large language models for agile software project management. *Electronics* 14, 1 (2024), 87.

[5] Rodrigo A Crisostomo, Nicolas A Nunez, and Giuliano Lopez-Burga. 2024. Enhancing Professional Employability: The Impact of Agile Methodology Training. *International Journal of Engineering Pedagogy* 14, 8 (2024).

[6] Daniela S. Cruzes and Tore Dybå. 2011. Recommended Steps for Thematic Synthesis in Software Engineering. In *2011 International Symposium on Empirical Software Engineering and Measurement*. IEEE, 275–284. doi:10.1109/ESEM.2011.36

[7] G Dhruva, Ishaan Shettigar, Srikrshna Parthasarthy, and VM Sapna. 2024. Agile Project Management Using Large Language Models. In *2024 5th International Conference on Innovative Trends in Information Technology (ICITIIT)*. IEEE, 1–6.

[8] Alexandre O Gérard et al. 2025. Evaluating and leveraging large language models in clinical pharmacology and therapeutics assessment: From exam takers to exam shapers. *British Journal of Clinical Pharmacology* (2025).

[9] Keito Inoshita. 2024. Assessing GPT's Legal Knowledge in Japanese Real Estate Transactions Exam. In *2024 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT)*. IEEE, 149–155.

[10] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Advances in Neural Information Processing Systems*, Vol. 33. 9459–9474.

[11] N. C. Mendonça. 2024. Evaluating ChatGPT-4 Vision on Brazil's National Undergraduate Computer Science Exam. *ACM Transactions on Computing Education* 24, 3 (2024), 1–56.

[12] Rahul Modak et al. 2023. Integrating LLMs into Agile Software Development: A 2023 Perspective on Productivity and Code Quality. *Well Testing Journal* 32, 2 (2023), 130–146.

[13] Alessandra Montenegro. 2019. Competences for the Future: A Comparative Analysis of Agile Certifications. *European Project Management Journal* 9, 2 (2019), 46–54.

[14] Andrea S Patrucco, Filomena Canterino, and Inga Minelgaite. 2022. How do scrum methodologies influence the team's cultural values? A multiple case study on agile teams in Nonsoftware industries. *IEEE Transactions on Engineering Management* 69, 6 (2022), 3503–3513.

[15] R. Pires, T. S. Almeida, H. Q. Abonizio, and R. F. Nogueira. 2023. Evaluating GPT-4's Vision Capabilities on Brazilian University Admission Exams. *arXiv preprint arXiv:2311.14169* (2023).

[16] Paul Ralph, Nauman Bin Ali, Sebastian Baltes, Domenico Bianculli, Jessica Diaz, Yvonne Dittrich, Neil Ernst, Michael Felderer, Robert Feldt, Antonio Filieri, et al. 2020. Empirical Standards for Software Engineering Research. *arXiv preprint arXiv:2010.03525* (2020).

[17] Leila Samimi and Shekoufeh Kolahdouz Rahimi. 2025. Bridging Agility and Automation: Enhancing Model-Driven Engineering with LLMs in Scrum. In *Agile Model-driven Engineering Workshop*. CEUR Workshop Proceedings.

[18] Edson G. Santana Jr. et al. 2025. Which Prompting Technique Should I Use? An Empirical Investigation of Prompting Techniques for Software Engineering Tasks. *arXiv preprint arXiv:2506.05614* (2025). Preprint; under review.

[19] Scrum.org. 2020. Scrum Guide and Professional Scrum Certifications. https://www.scrum.org/. Accessed: 2026-04-13.

[20] Jiho Shin, , et al. 2023. Prompt Engineering or Fine-Tuning: An Empirical Assessment of LLMs for Code. *arXiv preprint arXiv:2310.10508* (2023).

[21] Hanna Soroka-Potrzebna. 2021. The importance of certification in project management in the labor market. *Procedia Computer Science* 192 (2021), 1934–1943.

[22] Cayo Viegas, Rohit Gheyi, and Márcio Ribeiro. 2025. Assessing the Capability of LLMs in Solving POSCOMP Questions. *Journal of the Brazilian Computer Society* 31, 1 (Oct. 2025), 991–1004.

[23] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems*, Vol. 35. 24824–24837.

[24] Claes Wohlin, Per Runeson, Martin Höst, Magnus C. Ohlsson, Björn Regnell, and Anders Wesslén. 2012. *Experimentation in Software Engineering* (2 ed.). Springer. doi:10.1007/978-3-642-29044-2

[25] Zheying Zhang et al. 2024. Llm-based agents for automating the enhancement of user story quality: An early report. In *International Conference on Agile Software Development*. Springer Nature Switzerland Cham, 117–126.