Can LLMs Write Your Tests? Evaluating Automated ISO/IEC 25010-Based Test Generation for VR Metaverse Environments

João Azevêdo
Federal Institute of Bahia
Vitória da Conquista
joaosigliani@gmail.com

Alana Rocha
Federal Institute of Bahia
Vitória da Conquista
alanaagne0@gmail.com

Erica Ferreira
Federal Institute of Bahia
Vitória da Conquista
202321190003@ifba.edu.br

Guilherme Prado Silva
Federal Institute of Bahia
Vitória da Conquista
202211130001@ifba.edu.br

José Diaz Amado
Federal Institute of Bahia
Vitória da Conquista
jose_diaz@ifba.edu.br

Carina Santos Silveira
Federal University of Bahia
Camaçari
csssilveira@ufba.br

France Arnaut
Federal Institute of Bahia
Lauro de Freitas
francearnaut@ifba.edu.br

Rosângela Correia
State University of Bahia
Salvador
ranaaccioly@yahoo.com.br

Andrea Machado
SENAI CIMATEC
Salvador
andreamachado3d@gmail.com

Crescencio Lima
Federal Institute of Bahia
Vitória da Conquista
crescencio@gmail.com

Abstract

Virtual Reality (VR) metaverse environments impose demanding non-functional requirements (NFRs) on quality attributes. Manual design of NFR test cases is labour-intensive, expert-dependent, and difficult to keep aligned with evolving quality standards. We investigate whether four state-of-the-art Large Language Models (LLMs) can generate automated, ISO/IEC 25010-aligned NFR test cases for a VR metaverse, and whether human assessor ratings agree with an LLM-as-a-Judge evaluation of the same tests. We applied a five-stage Automated Test Case Generation (ATCG) workflow to five ISO/IEC 25010 quality attributes. Each generated test case was evaluated under two independent perspectives: (i) a two-assessor human panel rating; and (ii) an LLM-as-a-Judge protocol implemented as a custom skill applying the same rubric. Both perspectives were aggregated into weighted composite scorecards using domain-prioritised attribute weights. The two evaluation perspectives yield different rankings. Under human assessment, DeepSeek-V2 ranked first, followed by Copilot, ChatGPT, and Gemini. Under the LLM-as-a-Judge evaluation, Copilot ranked first, followed by ChatGPT, Gemini, and DeepSeek-V2 last. The rank inversion of DeepSeek-V2 traces to the judge penalising missing preconditions and measurement tool specifications that human assessors filled in with domain knowledge. LLMs are viable ATCG tools for VR metaverse NFRs when prompts are anchored to ISO/IEC 25010 sub-characteristics. Evaluation perspective is a first-class variable: human assessors favour conceptual breadth, while rubric-anchored LLM judges enforce specification completeness.

Keywords

Metaverse, VR Testing, LLMs, ISO/IEC 25010, Quality Assurance

1 Introduction

The proliferation of Virtual Reality (VR) and metaverse platforms has created a new class of interactive software whose quality requirements transcend traditional functional correctness. Users of immersive environments demand responsive frame rendering, resilient session management, robust security of personal data, and seamless cross-platform compatibility [6, 8]. Meeting these expectations requires systematic verification of non-functional requirements (NFRs), yet NFR testing remains one of the most labour-intensive activities in software quality assurance (SQA): tests must be designed against abstract quality attributes, executed in heterogeneous runtime conditions, and re-validated whenever the system evolves.

The ISO/IEC 25010 standard provides a structured decomposition of software product quality into eight characteristics and twenty-three sub-characteristics, offering a concrete vocabulary for specifying and evaluating NFRs [13]. However, translating ISO quality sub-attributes into executable test cases still requires substantial domain expertise, and automated approaches lag behind functional test generation tools. Large Language Models (LLMs) have recently emerged as powerful generative assistants for software engineering tasks, including code completion, requirements elicitation, and test case drafting [1, 16]. Studies show that LLMs can generate requirements aligned with ISO/IEC 25010 with up to 80% accuracy [1], and that they reduce the manual effort required to produce unit tests [17]. Whether this capability extends to the domain of NFR testing for VR systems—where quality attributes have nuanced, environment-specific semantics [11]—remains an open question.

This paper presents an empirical study that closes this gap. We evaluate four freely accessible, state-of-the-art LLMs in their ability to generate test cases for five ISO/IEC 25010 attributes applied to a VR metaverse system [8]. Our research questions are:

RQ$_1$. *To what extent can LLMs generate valid, comprehensive, and actionable NFR test cases for VR metaverse systems when guided by ISO/IEC 25010 quality attributes?*

RQ$_2$. *To what extent do human evaluator ratings agree with LLM-as-a-Judge assessments of the generated test cases, and what are the implications of any discrepancies for automated quality evaluation?*

The contributions of this work are:

(1) A reusable five-stage Automated Test Case Generation (ATCG) workflow that operationalises ISO/IEC 25010 attributes as LLM prompt scaffolding;
(2) A comparative empirical evaluation of DeepSeek-V2, Copilot, ChatGPT, and Gemini across Usability, Performance Efficiency, Reliability, Security, and Compatibility;
(3) A dual-perspective quality scorecard combining human assessor ratings with LLM-as-a-Judge evaluation, yielding a composite 0–10 score per LLM under both perspectives; and,
(4) Guidelines for practitioners applying LLM-based ATCG to immersive software systems, including prompt design principles and security-compliance considerations.

2 Related Work

This section reviews the literature relevant to automated test case generation using LLMs, software quality assessment based on the ISO/IEC 25010 standard, and testing approaches for VR applications. First, studies that employ LLMs to support software testing activities are discussed. Next, the ISO/IEC 25010 quality model is presented as a framework for defining and evaluating software quality attributes. The section then examines the particular challenges associated with VR software quality and testing, followed by a review of traditional test automation techniques. Finally, the identified research gap is highlighted, motivating the proposed approach that combines LLM-based test generation with ISO/IEC 25010 quality attributes in the context of VR applications.

2.1 LLMs for ATCG

The use of LLMs for test generation has gained considerable research momentum. Celik and Mahmoud [5] performed a review on LLMs for Automated Test Case Generation. Wang et al. [16] surveyed LLM-based software testing approaches and identified test case generation as one of the highest-impact application areas, noting that prompt engineering quality is the dominant factor influencing output correctness. Yuan et al. [17] evaluated ChatGPT for unit test generation across multiple programming languages and reported improvements over traditional automated tools in test readability and fault detection, while identifying coverage completeness as a remaining limitation.

Tasarsu et al. [15] conducted a systematic literature review (SLR) to analyze existing methods for LLM-based test case generation, highlighting trends, strengths, limitations, and research gaps in this area. Zhang et al. [18] surveyed LLM-based automated program repair and test generation, concluding that structured prompts anchored to formal specifications outperform open-ended test requests. Almonte et al. [1] focused on NFR generation, demonstrating that LLMs can produce ISO-aligned requirements with competitive accuracy when given explicit standard definitions in the prompt.

Three recent studies from the Brazilian testing community further contextualise this work. Barbosa et al. [3] investigated the use of LLM-generated tests for detecting semantic merge conflicts in Java code, demonstrating that automatically generated test cases can expose behavioural regressions that escape static analysis—a finding that motivates broader adoption of LLM-based ATCG beyond functional unit testing. Fagundes and Teixeira [7] evaluated multiple LLMs for multimodal GUI test generation in Android applications, comparing models on their ability to interpret UI screenshots and produce actionable test steps; their work highlights the sensitivity of output quality to prompt structure and multimodal grounding, echoing our observations about sub-characteristic specification. André and Alves [2] conducted an empirical study on generating exploratory test suites with ChatGPT, reporting that ChatGPT can produce diverse, readable exploratory tests but tends to miss complex interaction paths that require domain knowledge—a limitation directly relevant to NFR testing for immersive systems where domain context is critical.

Taken together, this body of work establishes that LLMs can generate useful test artifacts, but prior studies focus predominantly on functional unit tests or requirements text, leaving NFR test case generation, particularly for immersive systems, largely unexplored.

2.2 ISO/IEC 25010 Quality Models

ISO/IEC 25010 defines eight product quality characteristics: Functional Suitability, Performance Efficiency, Compatibility, Usability, Reliability, Security, Maintainability, and Portability. Peters and Aggrey [13] demonstrated how the standard's structure can be instantiated for enterprise systems by mapping each sub-characteristic to measurable indicators. Their work illustrates the standard's flexibility across domains and motivates its adoption as a prompt scaffold in our approach. For VR environments, three characteristics (Usability, Performance Efficiency, and Reliability) directly govern the perceptual quality of immersion, while Security and Compatibility address cross-platform data integrity—all five being relevant for real-time interactive systems.

2.3 VR Software Quality and Testing

VR application quality carries domain-specific properties absent from traditional quality models. Li et al. [11] analysed over 1.1 million user reviews of VR applications and constructed a taxonomy of twelve quality attributes specific to VR, including comfort, interaction responsiveness, and immersiveness, attributes not fully addressed by ISO/IEC 25010. Chang and Suh [6] confirmed that immersion and interaction quality are primary drivers of user satisfaction in VR exhibition contexts. Rzig et al. [14] examined the challenges of testing augmented and virtual reality applications, reporting that VR testers face unique difficulties in automating spatial, sensory, and hardware-dependent test oracles. These findings underscore the importance of adapting standard quality models to the VR context and justify our selection of five ISO attributes most relevant to VR runtime quality.

2.4 Traditional Test Automation Approaches

Prior to the LLM era, reinforcement learning (RL) was the dominant AI-based paradigm for automated test generation. Lima et al. [12] demonstrated that RL agents can learn action sequences that maximise code coverage and improve fault detection without manual script authoring. Cardoso et al. [4] evaluated DRL-MOBTEST on nine Android applications, achieving 74.43% average code coverage and a 63.52% improvement over random exploration. Islam et al. [10] mapped the landscape of AI techniques in software testing

and identified RL and neural approaches as the most productive directions for automation.

Despite these achievements, RL-based approaches require long training cycles, access to internal application states, and reconfiguration whenever the target system changes. LLMs bypass these constraints: being pre-trained on vast corpora. They generate test cases through prompt engineering without additional training cost. This agility makes them attractive for VR projects with rapidly evolving feature sets.

2.5 Research Gap

No prior work has combined LLMs with the ISO/IEC 25010 framework to automatically generate NFR test cases for VR metaverse environments. Table 1 positions this study against the most closely related work. Key observations are threefold. First, all prior empirical studies target either functional or GUI test generation; none focus on ISO-anchored NFR testing. Second, no study evaluates LLMs specifically on VR or immersive platforms, where quality attributes carry environment-specific semantics. Third, no prior work combines a human assessor panel with LLM-as-a-Judge evaluation to cross-validate quality ratings—a methodological contribution that enables RQ_2. This study directly addresses all three gaps.

3 Research Method

This study follows a *descriptive empirical* design. The target system is an educational VR metaverse [8], built on Unity3D and deployed via a social VR platform, featuring avatar locomotion, historically themed mini-games, and cross-platform support (browser, mobile, and standalone headset). The five-stage workflow is illustrated in Figure 1. Stage 4 (Quality Evaluation) was performed under two complementary perspectives: a human assessor panel and an LLM-as-a-Judge protocol.

3.1 Stage 1 — LLM Selection

Four widely adopted, freely accessible LLMs were selected to ensure comparable conditions and to reflect tools available to practitioners without institutional licences:

- DeepSeek-V2 — open-weight model, accessed through its official web interface;
- Microsoft Copilot (GPT-4o core) — available on the web and within Windows;
- OpenAI ChatGPT (GPT-4o) — free tier with usage limits;
- Google Gemini 2.5 Flash — free tier access.

In addition to these four generator LLMs, Claude Anthropic Sonnet 4.6 was used exclusively as the LLM-as-a-Judge (Stage 4), not as a test generator. This separation ensures that the judge model is independent of all evaluated systems.

Each model was queried through a fresh session (no prior conversation history, incognito mode, or newly created account) to prevent context leakage across queries.

3.2 Stage 2 — ISO/IEC 25010 Attribute Selection

Five of the eight quality characteristics defined by ISO/IEC 25010 were selected as most critical for VR runtime quality. The excluded characteristics (Maintainability, Portability, Functional Suitability) relate primarily to development-time concerns rather than user-facing execution quality. Table 2 lists the selected attributes and their most relevant sub-characteristics.

3.3 Stage 3 — Prompt Engineering and Test Case Generation

For each attribute, an identical prompt template was completed and submitted to all four LLMs. The template contained three components: (1) a synthesised description of the VR metaverse system (avatar locomotion, game objectives, historical characters, and cross-platform deployment); (2) the ISO/IEC 25010 attribute definition and its key sub-characteristics; and (3) an instruction to return one test cases in the structured format: *objective / steps / expected result*. Holding the template constant across models ensures that differences in output quality are attributable to model capability rather than prompt variation. In total, 20 test generation prompts were issued (5 attributes × 4 LLMs).

Table 3 presents the prompt template used for all LLMs to generate the tests. One template per attribute was instantiated. All templates are publicly available in the artifact repository.

3.4 Stage 4 — Quality Evaluation

Each generated test case was evaluated under two independent perspectives.

Human evaluation. Two independent assessors rated each test case on four criteria, with consensus reached for discrepancies:

(1) Coverage — Does the test address the declared ISO attribute and its sub-characteristics?
(2) Executability — Can the test be run on the current prototype without missing dependencies?
(3) Clarity — Are the objective, steps, and expected result unambiguous?
(4) Novelty — Does the test add value beyond standard regression scenarios?

Each criterion was rated on a five-point ordinal scale: Very Low (**VL**), Low (**L**), Medium (**M**), High (**H**), Very High (**VH**). Ratings were subsequently converted to a 0–10 numeric scale (**VH** =10, **H** =8, **M** =6, **L** =4, **VL** =2) and averaged across criteria to produce per-attribute scores. These scores were then combined into a weighted composite using the weights shown in Table 2. Security carries the highest weight (25%) to reflect the regulatory priority imposed by Brazil's Lei Geral de Proteção de Dados (LGPD) on systems handling user identity data.

LLM-as-a-Judge evaluation. To cross-validate human assessments and address RQ_2, we additionally applied an LLM-as-a-Judge [9, 19] protocol. Zheng et al. [19] demonstrated that strong LLMs can serve as reliable proxies for human evaluators in open-ended quality assessments. Gu et al. [9] systematised this paradigm, cataloguing bias types (position, verbosity, self-enhancement) and mitigation strategies; we adopt their recommendation of single-turn, criteria-anchored rubric prompting to minimise these biases.

We implemented a custom skill (`vr-testing-judge`) backed by Claude Anthropic Sonnet 4.6 as the judge model. The skill instructs the model to act as an impartial ISO/IEC 25010 software quality expert and to score each test case independently on the same four criteria used by human assessors, using the same VH/H/M/L/VL

Table 1: Comparison of related work on LLM-based test generation.

Study	LLM(s)	Domain	Test Type	Standard	Eval. Method	Judge
Wang et al. 2024	Multiple	General SE	Functional	—	Survey/metrics	Human
Yuan et al. 2024	ChatGPT	Multi-language	Unit tests	—	Automated	Human
Almonte et al. 2025	Multiple	General RE	NFR req.	ISO 25010	Accuracy	Human
André & Alves 2025	ChatGPT	Web apps	Exploratory	—	Empirical	Human
Barbosa et al. 2025	Multiple	Java code	Conflict detect.	—	Empirical	Human
Celik & Mahmoud 2025	Multiple	General SE	Functional	—	Review	Human
Fagundes & Teixeira 2025	Multiple	Android GUI	GUI/func.	—	Empirical	Human
Tasarsu et al. 2026	Multiple	General SE	Functional	—	SLR	Human
This work	4 LLMs	VR Metaverse	NFR tests	ISO 25010	Scorecard	Human + LLM-j

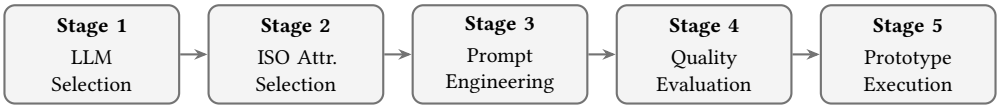


Figure 1: Five-stage workflow for LLM-based ISO/IEC 25010-guided automated test case generation.

Table 2: ISO/IEC 25010 attributes selected, key sub-characteristics, and attribute weight distribution for the composite scorecard.

Attribute	Key sub-characteristics	Weight
Security	Confidentiality, integrity, non-repudiation	25%
Performance Effic.	Time behaviour, resource utilisation	20%
Reliability	Maturity, fault tolerance, recoverability	20%
Usability	Appropriateness recognisability, learnability, operability	20%
Compatibility	Co-existence, interoperability	15%

ordinal scale converted to the 0–10 numeric equivalents. The judge is explicitly instructed not to infer information absent from the test text, mitigating verbosity bias. Table 4 summarises the judge skill structure.

3.5 Stage 5 — Prototype Execution

Each test case generated by an LLM was manually executed on the VR metaverse prototype [8]. Execution outcomes (pass/fail) were recorded and analysed qualitatively to assess the practical utility of the generated tests and to identify patterns in defect discovery relative to ISO attribute type.

3.6 Adherence to Reporting Guidelines for LLM-Based Studies

LLMs introduce specific threats to the reproducibility of empirical software engineering research, including non-determinism, opaque configurations, and the rapid evolution of underlying models. To mitigate these threats and to strengthen the transparency of our study, we follow the *Guidelines for Empirical Studies in Software Engineering involving Large Language Models* [**?**], which consolidate eight recommendations for designing and reporting LLM-based studies. Our study spans two of the study types identified by the guidelines: *LLMs used as tools*—the four generator models in Stages 1–3 and 5—and *LLMs used as judges*—the LLM-as-a-Judge introduced in Stage 4.

```
Template – <atribute>

You are a QA analyst. Generate one test case focused on
<atribute> (ISO/IEC 25010) for the metaverse described below:

You will test an educational, immersive, and inclusive meta-
verse that recreates the Battle of Itaparica from a decolonial
perspective. The 3D environment features the island modeled
with buildings, objects, and historical characters (e.g., Maria Fe-
lipa as a guide). Students explore scenarios, interact with NPCs,
and play minigames and quizzes that reinforce cultural, social,
and military content from the period. The metaverse runs in
a browser, on mobile devices, and on VR headsets, allowing
asynchronous multiplayer and cloud-based progress saving. It
was built using Spatial.io, a development platform that uses the
Unity game engine and the C# programming language.

Attribute: <atribute>
Sub-attributes: <sub-atributes>
Format the test case as:
Objective: . . .
Steps:
Expected Result: . . .
```

Table 3: Prompt template

`VR-Testing Judge Skill`

Role: Impartial judge, ISO/IEC 25010 software quality expert.
Context provided: ISO attribute + sub-attributes; full test case (Objective / Steps / Expected Result); VR-Testing metaverse description (Spatial.io, Unity, C#, browser/mobile/VR, async multiplayer).
Criteria evaluated (per test case):

- Coverage: does the test address the declared attribute and sub-characteristics?
- Executability: can the test run on the prototype without missing dependencies?
- Clarity: are objective, steps, and expected result unambiguous?
- Novelty: does the test add value beyond generic regression scenarios?

Scale: VH=10, H=8, M=6, L=4, VL=2. Per-test score = mean of four criteria.
Output: Per-criterion rating + 1–2 sentence justification + per-test score + aggregate summary across the test batch.
Bias mitigations: Single-turn rubric prompting; no inferred information; criteria defined before test is presented.

Table 4: LLM-as-a-Judge skill structure (`vr-testing-judge`).

4 Results

The following subsections present the quality evaluation results for each ISO/IEC 25010 attribute (Tables 5–9), followed by the weighted composite scorecard (Table 10) and prototype execution observations.

4.1 Usability

Table 5 summarizes the Usability results. Human assessors rated DeepSeek-V2 highest, assigning **VH** scores for Coverage, Executability, and Clarity. In contrast, the LLM-as-a-Judge assigned only **M** or **L** scores, citing the absence of a defined participant count and other missing experimental details. ChatGPT and Copilot showed greater agreement between evaluators, with both receiving high scores for Coverage and Clarity. Gemini 2.5 exhibited the opposite pattern, as the judge consistently rated its outputs higher than human assessors.

Table 5: Usability evaluation results.

	Coverage		Exec.		Clarity		Novelty	
LLM	**Hu**	**J**	**Hu**	**J**	**Hu**	**J**	**Hu**	**J**
ChatGPT	**VH**	**VH**	**H**	**M**	**VH**	**H**	**H**	**VH**
Copilot	**H**	**VH**	**H**	**H**	**H**	**VH**	**M**	**VH**
DeepSeek-V2	**VH**	**M**	**VH**	**L**	**VH**	**M**	**H**	**M**
Gemini 2.5	**L**	**VH**	**M**	**M**	**M**	**H**	**L**	**VH**

Figure 2 compares the usability profiles produced by human assessors and the LLM-as-a-Judge. ChatGPT (Figure 2a) achieved strong agreement on Coverage, although humans assigned higher scores for Execution and Clarity, while the judge rated Novelty more favorably. Gemini 2.5 (Figure 2d) showed substantial disagreement, with the judge assigning markedly higher scores for Coverage, Clarity, and Novelty. DeepSeek-V2 (Figure 2c) presented the greatest divergence: human assessors awarded top ratings for Coverage, Execution, and Clarity, whereas the judge penalized missing experimental details and assigned substantially lower scores. For Copilot (Figure 2b), the judge generally rated the generated tests more positively than humans, particularly for Coverage, Clarity, and Novelty, while both evaluators agreed on Executability.

4.2 Performance Efficiency

Table 6 summarizes the Performance Efficiency results. Human assessors rated DeepSeek-V2 and Copilot most favorably, assigning **VH** scores across most criteria. The LLM-as-a-Judge largely agreed with Copilot's strong performance but assigned lower scores to DeepSeek-V2, citing missing details regarding measurement-tool configuration and platform-specific evaluation procedures. ChatGPT received consistently high ratings from both evaluators, whereas Gemini 2.5 exhibited the largest positive bias from the judge, particularly for Coverage, Clarity, and Novelty.

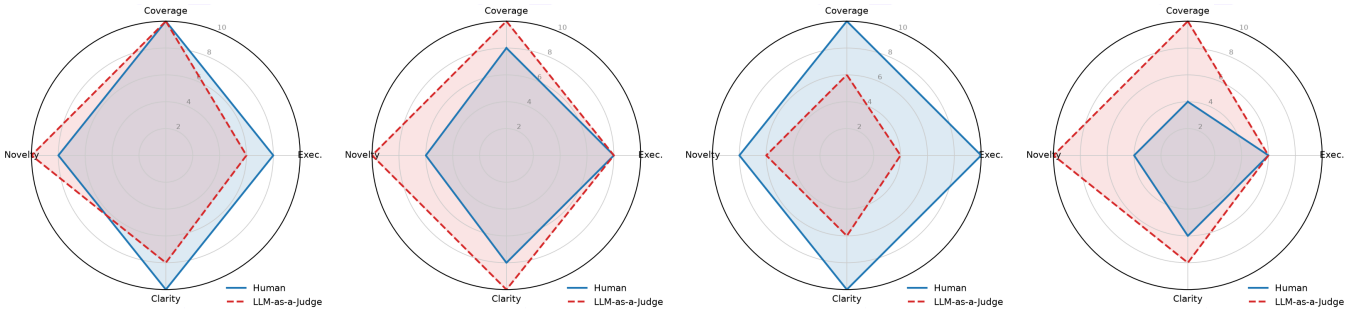
Table 6: Performance Efficiency evaluation results.

	Coverage		Exec.		Clarity		Novelty	
LLM	**Hu**	**J**	**Hu**	**J**	**Hu**	**J**	**Hu**	**J**
ChatGPT	**H**	**H**	**H**	**M**	**VH**	**H**	**M**	**H**
Copilot	**VH**	**VH**	**VH**	**VH**	**H**	**VH**	**H**	**H**
DeepSeek-V2	**VH**	**H**	**VH**	**M**	**VH**	**M**	**VH**	**M**
Gemini 2.5	**M**	**H**	**M**	**M**	**M**	**H**	**L**	**H**

Figure 3 illustrates the agreement and divergence patterns between human and LLM-as-a-Judge. Copilot (Figure 3b) achieved the strongest alignment, with both evaluators assigning high scores across all dimensions. ChatGPT (Figure 3a) also showed relatively consistent assessments, although human evaluators rated Executability and Clarity more favorably. In contrast, DeepSeek-V2 (Figure 3c) presented the greatest disagreement, with human assessors assigning uniformly high scores while the judge penalized missing implementation details. Gemini 2.5 (Figure 3d) exhibited the opposite pattern, as the judge consistently provided higher ratings than human evaluators, particularly for Novelty. Overall, the radar profiles reveal that evaluation differences stem primarily from the level of detail expected for practical execution rather than from disagreements regarding test objectives.

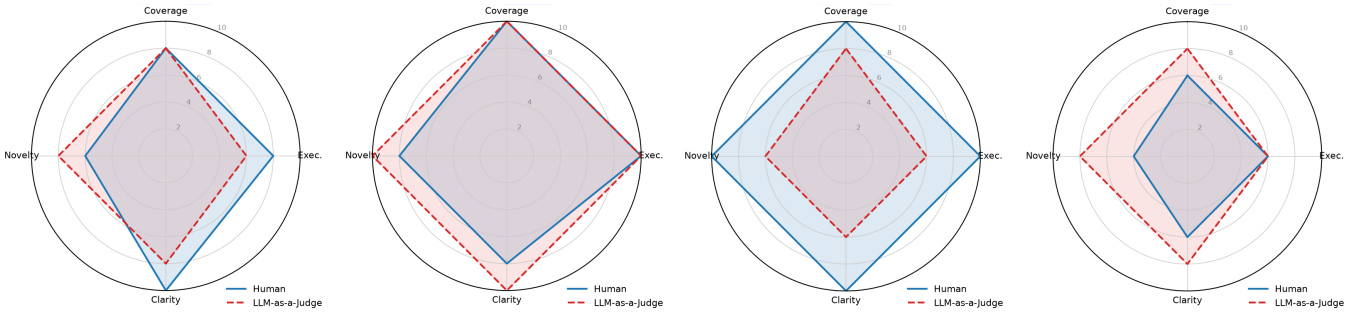
4.3 Reliability

Table 7 summarizes the Reliability results, which exhibit the highest overall agreement between human assessors and the LLM-as-a-Judge. Copilot achieved the most consistent performance, receiving **VH** ratings from both evaluators on Coverage and Executability. DeepSeek-V2 also performed strongly, although the judge assigned



(a) ChatGPT (b) Copilot (c) DeepSeek-V2 (d) Gemini 2.5

Figure 2: Usability attribute radar profiles across the four evaluation criteria for each LLM.



(a) ChatGPT (b) Copilot (c) DeepSeek-V2 (d) Gemini 2.5

Figure 3: Performance Efficiency attribute radar profiles across the four evaluation criteria for each LLM.

lower scores for Executability, Clarity, and Novelty due to missing implementation details. ChatGPT received favorable evaluations overall, with the judge assigning high scores for Clarity and Novelty. Gemini 2.5 showed the largest discrepancy, as the judge consistently rated its outputs more positively than human assessors.

Table 7: Reliability evaluation results.

	Coverage		Exec.		Clarity		Novelty	
LLM	**Hu**	**J**	**Hu**	**J**	**Hu**	**J**	**Hu**	**J**
ChatGPT	**H**	**M**	**H**	**H**	**H**	**VH**	**M**	**VH**
Copilot	**VH**	**VH**	**VH**	**VH**	**H**	**VH**	**H**	**VH**
DeepSeek-V2	**VH**	**VH**	**VH**	**M**	**VH**	**H**	**VH**	**H**
Gemini 2.5	**L**	**VH**	**M**	**M**	**M**	**H**	**L**	**H**

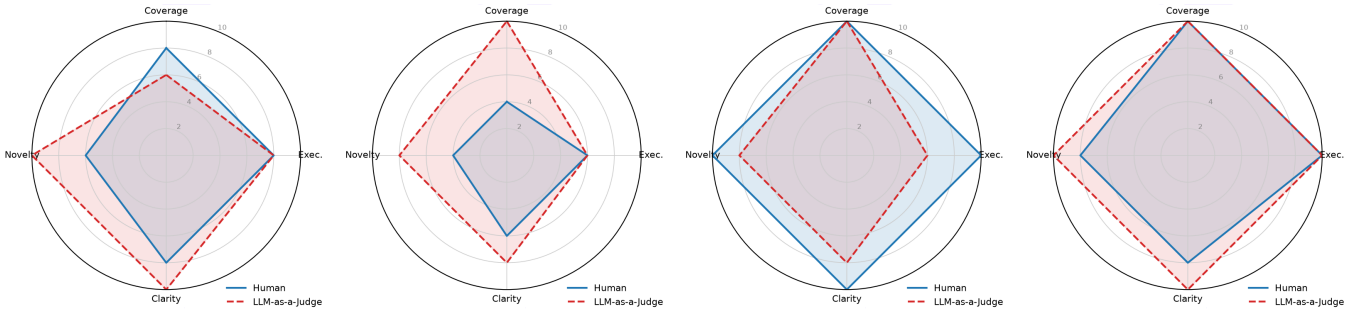
Figure 4 presents the agreement patterns across the four models. Copilot (Figure 4d) demonstrates the strongest alignment, with nearly overlapping radar profiles. ChatGPT (Figure 4a) exhibits moderate differences, primarily because the judge rated Clarity and Novelty more favorably. DeepSeek-V2 (Figure 4c) presents the opposite trend, with human evaluators assigning higher scores across most dimensions. Gemini 2.5 (Figure 4b) shows the greatest divergence, as the judge consistently provided higher ratings than human assessors. Overall, the radar profiles indicate that Reliability produced the closest human–judge agreement among all evaluated quality attributes.

4.4 Security

Table 8 summarizes the Security results, which show a high degree of agreement between human assessors and the LLM-as-a-Judge. All models received strong Coverage evaluations, with the judge assigning **VH** ratings across the board. Copilot and ChatGPT obtained consistently high scores for Executability and Clarity, reflecting the structured nature of their security test scenarios. DeepSeek-V2 also performed well, although the judge assigned lower scores for Executability and Clarity due to missing implementation details. Gemini 2.5 exhibited the largest positive difference between evaluators, particularly for Coverage and Novelty, as the judge placed greater value on its LGPD-related and attack-oriented scenarios.

Table 8: Security evaluation results.

	Coverage		Exec.		Clarity		Novelty	
LLM	**H**	**J**	**H**	**J**	**H**	**J**	**H**	**J**
ChatGPT	**H**	**VH**	**M**	**H**	**H**	**VH**	**M**	**H**
Copilot	**VH**	**VH**	**VH**	**H**	**H**	**VH**	**VH**	**H**
DeepSeek-V2	**VH**	**VH**	**VH**	**M**	**VH**	**M**	**VH**	**H**
Gemini 2.5	**M**	**VH**	**M**	**M**	**M**	**M**	**L**	**H**



(a) ChatGPT (b) Gemini 2.5 (c) DeepSeek-V2 (d) Copilot

Figure 4: Reliability attribute radar profiles across the four evaluation criteria for each LLM.

Figure 5 presents the evaluation profiles for each model. Copilot (Figure 5b) achieved the strongest alignment, with nearly overlapping human and judge assessments across all criteria. ChatGPT (Figure 5a) also showed close agreement, although the judge consistently assigned slightly higher scores. DeepSeek-V2 (Figure 5c) displayed the opposite pattern, with human assessors rating the generated tests more favorably than the judge. Gemini 2.5 (Figure 5d) presented the largest divergence, as the judge assigned substantially higher ratings for Coverage and Novelty. Overall, the radar profiles indicate that differences between evaluators were primarily associated with the level of procedural detail provided for test execution rather than with the relevance of the proposed security scenarios.

4.5 Compatibility

Table 9 summarizes the Compatibility results. Copilot achieved the highest consistency, receiving **VH** scores from both evaluators across all criteria. DeepSeek-V2 illustrates the largest disagreement: while human assessors assigned **VH** throughout, the judge downgraded Executability (**L**) and Clarity (**M**) due to the absence of explicit preconditions, device specifications, and tool references. ChatGPT and Gemini received higher judge ratings for Novelty, reflecting the value attributed to their multi-platform synchronization scenarios.

Table 9: Compatibility evaluation results.

	Coverage		Exec.		Clarity		Novelty	
LLM	**Hu**	**J**	**Hu**	**J**	**Hu**	**J**	**Hu**	**J**
ChatGPT	**H**	**H**	**H**	**M**	**H**	**H**	**M**	**VH**
Copilot	**VH**	**VH**	**VH**	**VH**	**H**	**VH**	**H**	**VH**
DeepSeek-V2	**VH**	**VH**	**VH**	**L**	**VH**	**M**	**VH**	**H**
Gemini 2.5	**M**	**H**	**M**	**M**	**M**	**H**	**L**	**VH**

Figure 6 visualizes the differences between human and judge assessments. ChatGPT (Figure 6a) exhibits moderate divergence, with higher human ratings for Executability and higher judge ratings for Novelty. Gemini 2.5 (Figure 6d) follows a similar pattern, as the judge consistently assigns higher scores except for Executability. DeepSeek-V2 (Figure 6c) presents the most pronounced discrepancy, with human ratings substantially exceeding judge ratings, particularly for Executability and Clarity. In contrast, Copilot (Figure 6b) shows near-complete agreement between evaluators, reinforcing its position as the most balanced and consistently rated model for Compatibility.

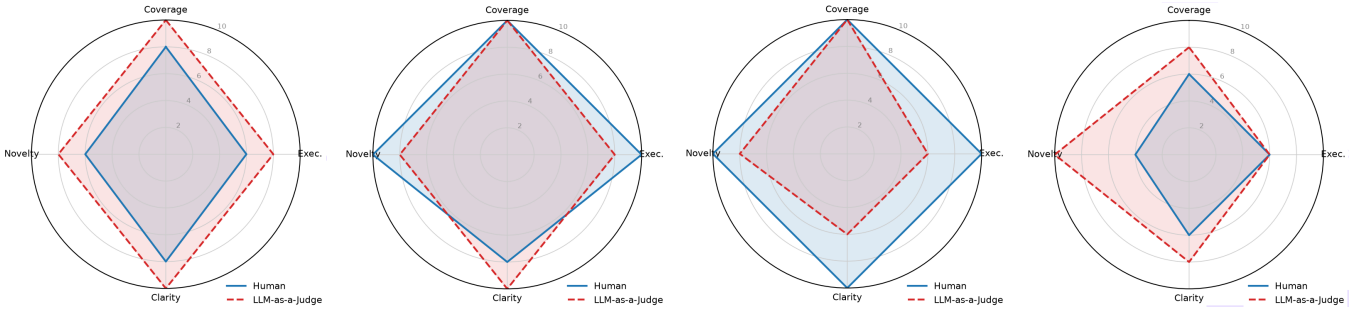
4.6 Weighted Composite Scorecard

Table 10 presents the final weighted quality scores derived from per-attribute numeric scores combined using the weights in Table 2. Two composite scores are reported: the human assessor score (Hu) and the LLM-as-a-Judge score (J), both computed with the same weight distribution.

The human ranking places DeepSeek-V2 first (9.0), Copilot second (7.8), ChatGPT third (7.4), and Gemini last (7.0). The LLM-as-a-Judge ranking inverts this order at the top: Copilot leads (9.6), followed by ChatGPT (8.4), Gemini (7.9), and DeepSeek-V2 last (6.9). The rank reversal of DeepSeek-V2 is the most striking finding: despite being rated best by human assessors, it received the lowest composite score from the judge, primarily because its tests consistently lacked explicit preconditions, measurement tool specifications, and participant count definitions—gaps penalised heavily by Executability and Clarity criteria when evaluated without tacit domain knowledge. Copilot's rise to first under the LLM-judge reflects its consistent provision of explicit tool references (Burp Suite, MSI Afterburner), quantified thresholds, and structured expected result tables that left no pass/fail ambiguity.

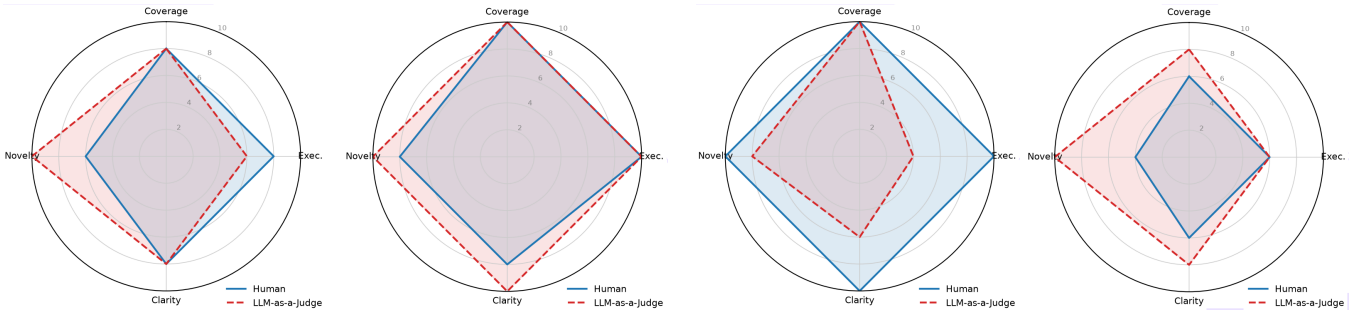
4.7 Prototype Execution

Test cases generated by DeepSeek-V2 and Copilot showed the highest execution success rate on the VR prototype, with most tests providing concrete, measurable steps that could be followed without ambiguity. Several DeepSeek-V2 security tests uncovered latent issues in session token management. ChatGPT tests were generally executable but required manual refinement to address missing pre-conditions. Gemini tests exhibited the highest ambiguity rate, with several steps lacking specific interaction targets or measurable expected values, reducing their practical utility in the prototype execution phase.



(a) ChatGPT (b) Copilot (c) DeepSeek-V2 (d) Gemini 2.5

Figure 5: Security attribute radar profiles across the evaluation criteria for each LLM.



(a) ChatGPT (b) Copilot (c) DeepSeek-V2 (d) Gemini 2.5

Figure 6: Compatibility attribute radar profiles across the four evaluation criteria for each LLM.

Table 10: Weighted quality scorecard: human assessor (Hu) and LLM-as-a-Judge (J) composite scores, strengths, and drawbacks.

LLM	Hu	J	Strengths	Drawbacks
DeepSeek-V2	9.0	6.9	Quantitative metrics across all attributes; separation by platform; LGPD-aware adverse scenarios.	Missing preconditions and tool configurations identified by LLM-judge; no explicit measurement procedures.
Copilot	7.8	9.6	Explicit tool references (Unity Profiler, Burp Suite, mitmproxy); well-structured expected results; LGPD compliance scenarios.	Subjectivity in "thinking aloud"; absence of intermittent network scenarios.
ChatGPT	7.4	8.4	Clear target audience; objective time metrics; precisely situated failure injection.	Missing participant count in usability; limited accessibility coverage.
Gemini 2.5	7.0	7.9	Rich thematic contextualisation; multi-headset compatibility; graceful degradation scenarios.	Subjective pass/fail criteria; lack of specific preconditions; co-existence sub-attribute undertested.

5 Discussion

This section interprets the results obtained. First, we analyze the performance differences among the evaluated models. Next, we compare human assessments with LLM-as-a-Judge evaluations to examine their level of agreement and divergence. We then discuss the influence of prompt engineering on test case quality, followed by considerations related to security, data protection regulations, and the LGPD.

5.1 DeepSeek-V2's Consistent Superiority

DeepSeek-V2 consistently outperformed its peers across all five ISO attributes. Qualitative analysis of the generated tests suggests that its advantage stems from three factors: (1) systematic coverage of every sub-characteristic listed in the prompt, rather than selecting a subset; (2) inclusion of quantitative thresholds (e.g., maximum acceptable latency in milliseconds, minimum frame rate) that convert abstract quality goals into executable oracles; and (3) scenario diversity that goes beyond happy-path verification to include adversarial and boundary conditions. These characteristics are precisely those that distinguish a professional test suite from a superficial one, suggesting that DeepSeek-V2's pre-training includes a richer exposure to formal software quality artefacts.

5.2 Human vs. LLM-as-a-Judge: Agreement and Divergence (RQ_2)

The most consequential finding to emerge from our dual-perspective evaluation is the rank inversion of DeepSeek-V2. Human assessors placed it first (9.0/10), while the LLM-as-a-Judge ranked it last (6.9/10). This divergence is not random noise: it traces systematically to how each evaluator weights *tacit executability* versus *explicit specification completeness*. Human assessors, familiar with the VR-Testing prototype, applied implicit domain knowledge to fill in missing preconditions (e.g., inferring that a profiler would be needed even if unnamed). The LLM judge, instructed not to infer information absent from the test text, penalised these same gaps heavily in Executability and Clarity. This finding replicates the verbosity-vs-specificity tension documented by Gu et al. [9]: human evaluators readily accept plausible tests, while rubric-anchored LLM judges enforce specification completeness.

The two perspectives agree most closely on Security (all four LLMs received **VH** Coverage from both) and diverge most on Usability and Compatibility, where DeepSeek's human-versus-judge gap is largest. Agreement is also high for Copilot, which ranked second (human) and first (judge)—suggesting that Copilot's explicit, tool-referenced style satisfies both evaluation modes.

These findings have practical implications for automated quality assurance pipelines. If LLM-as-a-Judge is used as a filter for test cases before human review, it will favour tests that are explicitly specified over tests that are comprehensive but assume contextual knowledge. Practitioners should therefore calibrate the judge rubric to the expected reader expertise: teams with deep domain knowledge may find human assessments more predictive of actual test utility, while teams adopting externally generated tests benefit from the judge's stricter executability enforcement.

5.3 Prompt Engineering as the Primary Quality Lever

A cross-model pattern emerged: when the prompt explicitly enumerated ISO sub-characteristics (e.g., "confidentiality, integrity, and non-repudiation" for Security), all LLMs produced higher-coverage outputs than when left to interpret the parent attribute alone. This aligns with findings from Zhang et al. [18] and Almonte et al. [1], who reported that formal specification anchoring significantly improves LLM output quality. Practitioners should therefore include the full sub-characteristic list from ISO/IEC 25010 in the prompt rather than relying on the model's implicit understanding of the parent attribute.

5.4 Security, LGPD, and Regulatory Context

Security received the highest weight (25%) in our scorecard because VR metaverse systems collect and process user identity, biometric movement data, and social interaction logs—all regulated under the LGPD. The gap between Copilot (**VH** Novelty in Security) and ChatGPT/Gemini highlights a practical concern: two out of four evaluated LLMs produced security tests that did not address authentication or integrity validation, which are minimum LGPD compliance requirements. Teams adopting LLM-generated security test cases should therefore complement them with explicit prompts referencing the applicable regulatory framework, as Copilot's LGPD-specific scenarios demonstrate this approach is feasible.

5.5 Comparison with RL-Based Approaches

RL-based test automation (Lima et al. [12]; Cardoso et al. [4]) offers complementary strengths: RL agents can explore emergent state-space failures that are not predictable from a static system description, and can optimise for coverage metrics through interaction. LLMs, by contrast, generate tests purely from textual descriptions without system interaction, which limits their ability to discover integration-level defects. The key practical advantage of LLMs is speed: generating a full NFR test suite via the five-stage workflow requires hours, whereas RL approaches require training cycles spanning days to weeks. A hybrid strategy—using LLMs for initial NFR test scaffolding and RL for exploration-based fault discovery.

5.6 Implications for Practice

Based on our findings, we offer four guidelines for practitioners: (1) Anchor prompts to ISO sub-characteristics: include the full sub-characteristic list in the prompt rather than the attribute name alone; (2) Treat Copilot as the primary generator for explicitly specified tests and DeepSeek-V2 for conceptual breadth: Copilot's explicit tool references and structured expected results satisfy both human and LLM-judge evaluation modes, while DeepSeek-V2's comprehensive scenarios may need specification augmentation before deployment; (3) Use LLM-as-a-Judge as a completeness filter: applying a rubric-anchored LLM judge (our custom skill `vr-testing-judge` Claude Sonnet 4.6 in our setting) before human review flags underspecified preconditions and undefined measurement procedures that human assessors with domain familiarity tend to overlook; (4)Calibrate the judge to team context: when domain experts will execute the tests, human scores are more predictive of practical utility; when tests will be shared with external teams or integrated into automated pipelines, the LLM judge's stricter executability criterion is the better quality gate.

6 Threats to Validity

Following, we discuss potential threats.

Internal validity. The qualitative ordinal evaluation introduces assessor subjectivity, particularly for Clarity and Novelty. We mitigated this through a two-assessor independent evaluation with consensus resolution, but a single VR domain may have introduced shared assumptions. Prompt template formulation directly influences LLM outputs; minor phrasing variations may yield different results. LLM temperature settings and session context—while controlled by using fresh sessions—were not explicitly documented, which limits exact replication. The LLM-as-a-Judge is itself a language model and may exhibit position, verbosity, or self-enhancement biases [9]; we mitigated this through single-turn rubric prompting and by prohibiting the judge from inferring information absent from the test text, but residual bias cannot be excluded.

External validity. The study targets a single educational VR metaverse [8]. Results may not generalise to commercial VR platforms, non-metaverse interactive systems, or domains where the five selected ISO attributes have different relative importance. Replication

with additional systems and domains is needed to establish broader validity.

Construct validity. The ordinal-to-numeric mapping assumes equal intervals between levels, which is an approximation. The attribute weights in Table 2 were assigned based on domain judgement (LGPD priority for Security) rather than formal elicitation from multiple stakeholders; different weight configurations would alter the composite ranking, though DeepSeek-V2's margin is large enough to remain first under reasonable weight perturbations.

Conclusion validity. The set of test cases generated per attribute is limited in size. LLMs are continuously updated; model versions evaluated here may produce different results in future releases. The manual execution phase does not constitute a statistically powered defect-detection study and should be treated as a qualitative pilot observation rather than definitive evidence of production readiness.

7 Conclusion

This paper presented an empirical evaluation of four state-of-the-art LLMs—DeepSeek-V2, Copilot, ChatGPT, and Gemini 2.5 Flash—as ATCG for ISO/IEC 25010 NFR in a VR metaverse environment. We proposed a reusable five-stage ATCG workflow and a dual-perspective weighted quality scorecard combining human assessor ratings with LLM-as-a-Judge evaluation.

Addressing RQ_1, all four models demonstrated that LLM-based NFR test generation is viable when prompts are anchored to explicit ISO sub-characteristics. Addressing RQ_2, reveals a systematic divergence: human assessors applied tacit domain knowledge to accept conceptually rich but underspecified tests, while the rubric-anchored judge penalised missing preconditions, tool configurations, and participant counts. This finding confirms that *evaluation perspective is a first-class variable* in LLM-based ATCG studies and should be explicitly reported alongside test quality scores.

Future work should: (i) replicate the workflow across additional domains and VR systems to establish external validity; (ii) formalise assessor protocols with inter-rater reliability measurement (Cohen's κ); (iii) explore hybrid LLM+RL pipelines combining prompt-driven test scaffolding with exploration-based fault discovery; (iv) develop automated execution frameworks that bridge LLM-generated test specifications and runnable test scripts; and (v) investigate prompt augmentation strategies that improve DeepSeek-V2's executability specification without sacrificing its conceptual breadth.

Artifact Availability

The dataset of generated test cases, evaluation rubrics, and scoring spreadsheets supporting this study is available at: https://github.com/crescenciolima/VR-Testing.

Acknowledgments

Generative AI tools were used to assist with literature search and text editing during manuscript preparation including *Gemini*, *ChatGPT*, and *Claude*. All experimental design, evaluation rubrics, data analysis, result interpretation, and scientific claims are the sole responsibility of the authors.

References

[1] Jomar Thomas Almonte, Santhosh Anitha Boominathan, and Nathalia Nascimento. 2025. Automated Non-Functional Requirements Generation in Software Engineering with Large Language Models: A Comparative Study. arXiv:2503.15248 [cs.SE] https://arxiv.org/abs/2503.15248

[2] Natália André and Everton Alves. 2025. Generating Exploratory Test Suites with ChatGPT: Insights from an Empirical Study. In *Anais do X Simpósio Brasileiro de Testes de Software Sistemático e Automatizado* (Recife/PE). SBC, Porto Alegre, RS, Brasil, 65–74. doi:10.5753/sast.2025.14094

[3] Nathalia Barbosa, Paulo Borba, and Léuson Silva. 2025. Detecção de Conflitos Semânticos com Testes Gerados por LLM. In *Anais do X Simpósio Brasileiro de Testes de Software Sistemático e Automatizado* (Recife/PE). SBC, Porto Alegre, RS, Brasil, 18–27. doi:10.5753/sast.2025.13826

[4] Ana Cardoso, Cleicy Santos, Eliane Collins, Kelen Lima, Pablo Quiroga, and Marlon Griego. 2023. Evaluation of Automatic Test Case Generation for the Android Operating System Using Deep Reinforcement Learning. In *Anais do XXII Simpósio Brasileiro de Qualidade de Software (SBQS)*. SBC, Porto Alegre, RS, Brasil, 228–235. https://sol.sbc.org.br/index.php/sbqs/article/view/27109

[5] Arda Celik and Qusay H. Mahmoud. 2025. A Review of Large Language Models for Automated Test Case Generation. *Machine Learning and Knowledge Extraction* 7, 3 (2025). doi:10.3390/make7030097

[6] Sunghyup Chang and Jungwook Suh. 2025. The Impact of VR Exhibition Experiences on Presence, Interaction, Immersion, and Satisfaction: Focusing on the Experience Economy Theory (4Es). *Systems* 13, 1 (2025), 55. doi:10.3390/systems13010055 Article no. 55.

[7] 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* (Recife/PE). SBC, Porto Alegre, RS, Brasil, 28–36. doi:10.5753/sast.2025.13852

[8] Erica Ferreira, Beatriz Oliveira, Carina Silveira, France Arnaut, Crescencio Lima, Andrea Machado, José Amado, and Juliendrios Oliveira. 2025. A gamificação como estratégia de memorabilidade em prol da aprendizagem. In *Anais do XXIV Simpósio Brasileiro de Jogos e Entretenimento Digital* (Salvador/BA). SBC, Porto Alegre, RS, Brasil, 1637–1647. doi:10.5753/sbgames.2025.10179

[9] Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Zhouchi Lin, Bowen Zhang, Lionel Ni, Wen Gao, Yuanzhuo Wang, and Jian Guo. 2026. A survey on LLM-as-a-judge. *The Innovation* 7, 6 (2026), 101253. doi:10.1016/j.xinn.2025.101253

[10] Mahmudul Islam, Farhan Khan, Sabrina Alam, and Mahady Hasan. 2023. Artificial Intelligence in Software Testing: A Systematic Review. doi:10.1109/TENCON58879.2023.10322349

[11] Shanshan Li, Liguo Wei, Yicong Liu, Cuiyun Gao, Shing-Chi Cheung, and Michael R. Lyu. 2023. Towards Modeling Software Quality of Virtual Reality Applications from Users' Perspectives. arXiv:2308.06783 [cs.SE] https://arxiv.org/abs/2308.06783

[12] Diogo Lima, Danyllo Albuquerque, Emanuel Dantas Filho, Mirko Perkusich, and Angelo Perkusich. 2023. Integrating Reinforcement Learning in Software Testing Automation: A Promising Approach. In *Anais do III Workshop Brasileiro de Engenharia de Software Inteligente* (Campo Grande/MS). SBC, Porto Alegre, RS, Brasil, 39–41. doi:10.5753/ise.2023.235976

[13] Emmanuel Peters and George Kwamina Aggrey. 2020. An ISO 25010 Based Quality Model for ERP Systems. *Advances in Science, Technology and Engineering Systems Journal* 5, 2 (2020), 578–583.

[14] Dhia Elhaq Rzig, Nafees Iqbal, Isabella Attisano, Xue Qin, and Foyzul Hassan. 2023. Virtual Reality (VR) Automated Testing in the Wild: A Case Study on Unity-Based VR Applications. In *Proceedings of the 32nd ACM SIGSOFT International Symposium on Software Testing and Analysis* (Seattle, WA, USA) *(ISSTA 2023)*. Association for Computing Machinery, New York, NY, USA, 1269–1281. doi:10.1145/3597926.3598134

[15] Murat Tasarsu, Ahmet Vedat Tokmak, and Cagatay Catal. 2026. Test Case Generation Using Large Language Models: A Systematic Literature Review. *Cluster Computing* 29, 4 (2026), 227. doi:10.1007/s10586-026-06021-z

[16] Junjie Wang, Yuchao Huang, Chunyang Chen, Zhe Liu, Song Wang, and Qing Wang. 2024. Software Testing With Large Language Models: Survey, Landscape, and Vision. *IEEE Transactions on Software Engineering* 50, 4 (2024), 911–936. doi:10.1109/TSE.2024.3368208

[17] Zhiqiang Yuan, Mingwei Liu, Shiji Ding, Kaixin Wang, Yixuan Chen, Xin Peng, and Yiling Lou. 2024. Evaluating and Improving ChatGPT for Unit Test Generation. *Proc. ACM Softw. Eng.* 1, FSE, Article 76 (July 2024), 24 pages. doi:10.1145/3660783

[18] Quanjun Zhang, Chunrong Fang, Yuxiang Ma, Weisong Sun, and Zhenyu Chen. 2023. A Survey of Learning-based Automated Program Repair. *ACM Trans. Softw. Eng. Methodol.* 33, 2, Article 55 (Dec. 2023), 69 pages. doi:10.1145/3631974

[19] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-bench and Chatbot Arena. In *Proceedings of the 37th International Conference on Neural Information Processing Systems* (New Orleans, LA, USA) *(NIPS '23)*. Curran Associates Inc., Red Hook, NY, USA, Article 2020, 29 pages.