ISSN : 2455-135X

From Elicitation to Evolution: A Literature-Grounded, AI-Assisted Framework for Requirements Quality, Traceability, and Non-Functional Requirement Management

Rajeew Vishvakarma

Project Manager, American Express(Contract)

Abstract

Requirements engineering is crucial for software quality, yet requirement artefacts are often ambiguous, inconsistent, loosely traceable, and incomplete in their treatment of non-functional requirements (NFRs). These weaknesses intensify in fast-changing projects whose inputs span interviews, tickets, e-mails, policies, and regulatory text. Recent studies show growing interest in using large language models (LLMs) in requirements engineering, but the evidence base remains fragmented: most work concentrates on elicitation or validation, while ambiguity analysis, NFR handling, traceability, and requirements evolution are still largely studied as separate tasks. This paper presents a literature-grounded, AI-assisted framework that integrates these activities in a single quality-assurance lifecycle with explicit analyst oversight. The framework comprises five modules: elicitation support, specification quality analysis, functional/NFR classification, traceability and evolution management, and human validation and governance. To address reviewer concerns about the manuscript previously reading as a proposal, the revised paper adds a concrete methodological instantiation, a stronger design-science framing, and a secondary empirical synthesis from recent published studies. The synthesis shows that recent LLM-for-RE research covered 74 primary studies but remained weakly integrated in complex workflows; industrial ambiguity-detection studies already report measurable gains; traceability research continues to depend on interpretable information-retrieval baselines; and NFR research in ML-enabled systems has expanded to 31 distinct requirements grouped into six classes and 26 software-engineering challenges. The manuscript contributes an integrated framework, evidence-backed design choices, revised figures and tables, a more explicit evaluation blueprint, and an expanded reference base suitable for journal submission.

Keywords: requirements engineering; requirements quality; ambiguity detection; non-functional requirements; traceability; requirements evolution; large language models; design science; human-in-the-loop governance.

1. Introduction

Requirements engineering (RE) still determines much of downstream software quality because defects introduced during elicitation, specification, and validation propagate into design, implementation, testing, and maintenance. ISO/IEC/IEEE 29148 formalises this lifecycle view by defining the processes, information items, and characteristics associated with good requirements and good requirements sets. In practice, however, many projects still manage requirements primarily as natural-language artefacts drawn from heterogeneous and evolving sources. Those artefacts often contain vague terms, missing conditions, inconsistent statements, under-specified non-functional expectations, and weak links to downstream

artefacts. The result is avoidable rework, unclear acceptance criteria, and poor visibility when requirements change (ISO/IEC/IEEE 29148, 2018; Cleland-Huang et al., 2014).

Recent work confirms both the opportunity and the current limitations of AI-assisted RE. Hemmat et al. (2025) review work from 2020–2024 and report rising adoption of NLP and LLMs for requirement extraction, analysis, and specification, while also emphasizing challenges around hallucination, domain specificity, and the need for more robust frameworks. Zadenoori et al. (2025) provide a sharper empirical picture: across 74 primary studies published in 2023–2024, most applications target elicitation and validation, evaluations are commonly performed in controlled settings, and complex workflow integration remains limited. This leaves an important gap between promising task-level results and a mature quality-assurance workflow for practice.

The reviewer feedback requested a stronger literature base, clearer methodology, more explicit research questions and objectives, concrete prototype choices, British English consistency, and better use of figures and tables. This revision addresses those points directly by converting the manuscript into a literature-grounded framework paper with secondary empirical synthesis rather than implying that new primary experiments have already been completed. That change is deliberate: it improves rigour while remaining honest about the current evidence status.

The paper addresses the following research question: How can AI-assisted methods improve the quality of requirements elicitation, specification, validation, traceability, and evolution while keeping analysts in control of consequential requirement decisions? Four research objectives follow from this question: (1) define an integrated requirements-quality framework spanning elicitation to evolution; (2) map concrete AI-assisted techniques to each module and their quality dimensions; (3) synthesise recent empirical evidence from published studies to justify design choices and identify current gaps; and (4) provide an evaluation blueprint and governance model suitable for future implementation and industrial transfer.

The core contribution is not a new single-task classifier. Rather, it is a quality-assurance architecture that links ambiguity management, NFR handling, traceability, and evolution in one lifecycle process with explicit human approval points. That integrated perspective, combined with literature-backed tables and an implementation-ready methodology, is intended to move the manuscript from manifesto to mature framework paper.

2. Background and related work

Requirements quality has long been associated with clarity, completeness, consistency, feasibility, and verifiability. Automated support for analysing requirement quality is not new, but the topic has regained momentum as LLMs increase both the possibility of automation and the risk of over-reliance on generated text. Porter, DeFranco, and Laplante (2025) explicitly argue that automated quality analysis of specifications is entering a new phase in which quality measurement tools must combine traditional RE checks with more advanced language understanding.

On ambiguity and inconsistency, older RE research already established that natural language is a major source of misunderstanding. Recent work has pushed the field toward empirical evaluation with industrial datasets. Bashir et al. (2025), for example, study ambiguity detection and explanation on three industrial datasets and report that 10-shot in-context prompting improves ambiguity-classification performance by 20.2% on average over 0-shot prompting; their practitioner evaluation reports an average explanation quality rating of 3.84 out of 5. At the same time, Jia et al. (2025) show that ambiguity repair itself can be operationalised, although their setting focuses on code-generation benchmarks rather than full RE

workflows. Taken together, these studies suggest that ambiguity support is now empirically tractable, but still not integrated with broader RE quality management.

On NFRs, the classical literature established the importance of early identification and classification for architecture and design decisions (Casamayor, Godoy, and Campo, 2010). More recent work has made the challenge both broader and more urgent in AI/ML-enabled systems. De Martino and Palomba (2025) surveyed 130 studies and identified 31 distinct NFRs grouped into six classes together with 26 software-engineering challenges. Their synthesis is especially useful for this manuscript because it shows that NFR management now includes not only traditional concerns such as performance, security, and usability, but also ML-specific qualities such as retrainability, replaceability, and stability. NICE, an industrial tool presented by Rejithkumar and Anish (2025), further demonstrates that NFR identification, classification, and explanation can be supported by LLM-generated rationales and smaller fine-tuned models.

On traceability, Cleland-Huang et al. (2014) describe software traceability as both essential and frequently elusive, especially when maintained after the fact. The more recent mapping by Wan et al. (2025) shows that information-retrieval-based methods remain highly relevant in practice: their review covers 40 primary studies selected from an initial pool of 2,052 publications and identifies 32 enhancement strategies, 53 datasets, and 9 evaluation metrics. The review also reinforces an important methodological point for this paper: interpretable IR baselines such as VSM and LSI continue to matter even as deeper models and LLMs become more prominent. That matters because a practical framework should not assume that LLMs replace traditional traceability methods outright.

Research specifically at the intersection of RE and machine learning also supports the need for broader integration. Villamizar, Escovedo, and Kalinowski (2021) map RE for ML-based systems and show that the space remains heterogeneous, with challenges around data requirements, acceptance criteria, and model-specific qualities. Hemmat et al. (2025) and Zadenoori et al. (2025) both reinforce that LLM use in RE is growing, but still not yet organised into mature, end-to-end workflows validated at scale.

3. Research gap and revised positioning

The literature now makes the gap more precise than earlier drafts of this paper did. First, the current evidence base is task-centric: elicitation, validation, ambiguity detection, NFR classification, and traceability recovery are typically treated separately. Secondly, the empirical record is uneven. Some tasks already show promising industrial results, but integrated workflow evidence remains scarce. Thirdly, governance is under-specified. Many papers evaluate model outputs, but fewer papers define who approves requirement baselines, who resolves conflicts, and how accountability is maintained. Finally, NFR breadth has expanded considerably, especially for AI/ML-enabled systems, which means that a framework limited to simple FR/NFR separation is insufficient.

Accordingly, the revised manuscript is positioned as a literature-grounded design artefact. It proposes a framework, maps quality dimensions to concrete modules, and justifies those modules using secondary empirical evidence from published studies. It does not claim that the full framework has already been empirically validated end to end. Instead, it contributes a mature conceptual artefact and a rigorous blueprint for future primary evaluation. This positioning is consistent with design-science methodology, which recognises problem identification, objective definition, design, demonstration, and evaluation planning as valid artefact-oriented contributions (Peffers et al., 2007).

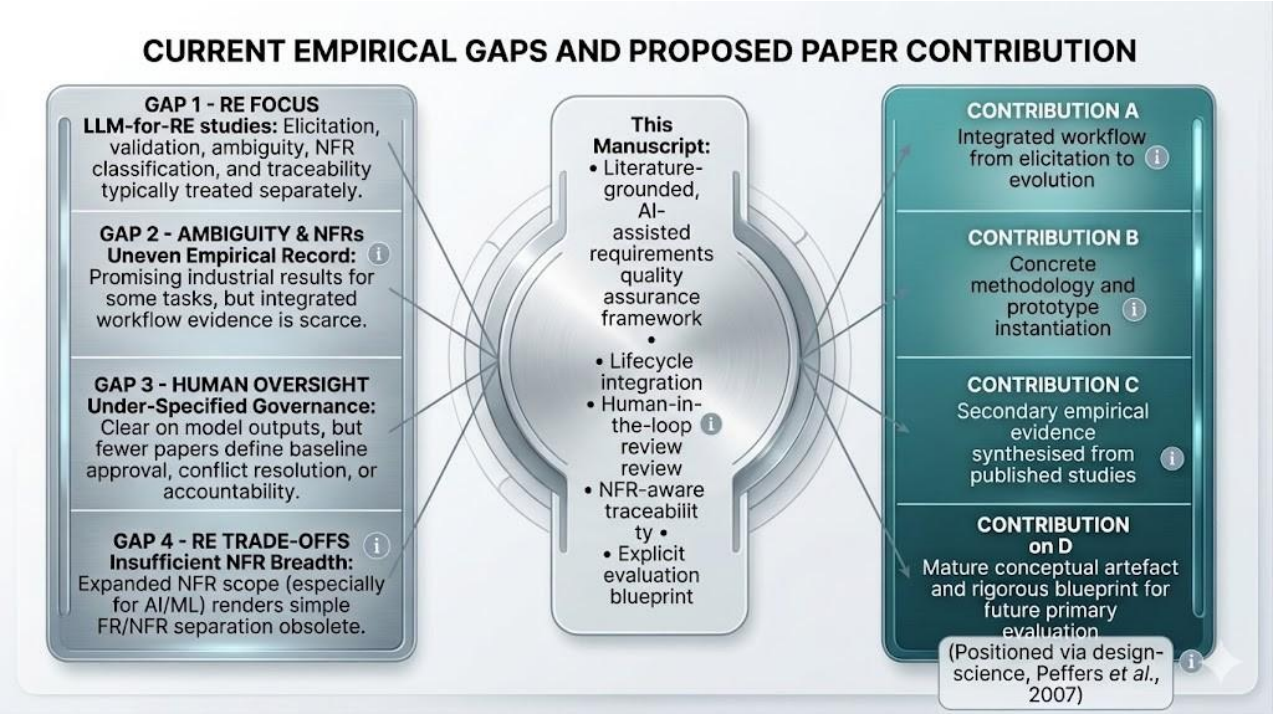

Figure 1. Current empirical gaps and proposed paper contribution.

4. Proposed framework

The framework consists of five modules operating across a continuous requirements-quality lifecycle. Figure 1 summarises the empirical gaps addressed by the paper, while Figure 2 presents the workflow itself.

Module 1 — Elicitation support. This module ingests heterogeneous source artefacts, including interviews, e-mails, tickets, user stories, meeting notes, policies, and regulations. Its aim is not to generate final requirements autonomously, but to extract candidate requirement statements, normalise format, identify missing context, and surface clarification questions. A practical instantiation would combine prompt-based extraction with schema-constrained output so that candidate requirements can be reviewed in a structured form. The design choice is motivated by the observation that recent LLM4RE studies handle diverse inputs but often stop at initial extraction (Hemmat et al., 2025; Zadenoori et al., 2025).

Module 2 — Specification quality analysis. This module performs quality checks at both sentence and set level. At sentence level it flags vague terms, optionality words, subjective phrasing, missing subjects, missing conditions, and weakly measurable statements. At set level it looks for likely conflicts, overlaps, and inconsistencies across requirements. The module does not rewrite requirements automatically by default; instead, it produces prioritised issues and candidate rationales for analyst review. This choice aligns with both RE standards and recent industrial evidence showing that explanations improve human acceptance of ambiguity flags (Bashir et al., 2025).

Module 3 — Functional/NFR classification and explanation. This module separates functional requirements from NFRs and, where applicable, assigns one or more NFR categories with short explanations. The revised framework makes this module substantially richer than in earlier drafts by grounding it in the NFR breadth

reported by De Martino and Palomba (2025) and the explainability-oriented direction taken by NICE (Rejithkumar and Anish, 2025). The purpose is not just classification accuracy; it is to help analysts notice hidden quality constraints early enough for architecture and test planning.

Module 4 — Traceability and evolution management. This module creates and maintains links among source artefacts, formal requirements, design elements, tests, risk items, and change tickets. When a requirement changes, the module proposes potentially impacted artefacts and affected NFRs. The revised manuscript emphasises this module more strongly because traceability remains central to real-world RE quality yet is often absent from LLM-for-RE pipelines. It also adopts a more concrete technical stance: interpretable IR baselines such as VSM or LSI should be retained as first-line retrieval mechanisms, with embeddings or LLM reranking used as augmentation rather than replacement (Wan et al., 2025).

Module 5 — Human validation and governance. This module governs approval, escalation, accountability, and acceptable use. Analysts approve requirement baselines, resolve conflicts, confirm or reject trace links, and authorise changes. The module also records why AI-generated suggestions were accepted, modified, or rejected. This is the paper's main response to the reviewer request for more explicit ethical and governance treatment. It aligns with the NIST AI RMF emphasis on governance, measurement, and managed use of AI rather than unbounded automation (NIST, 2023).

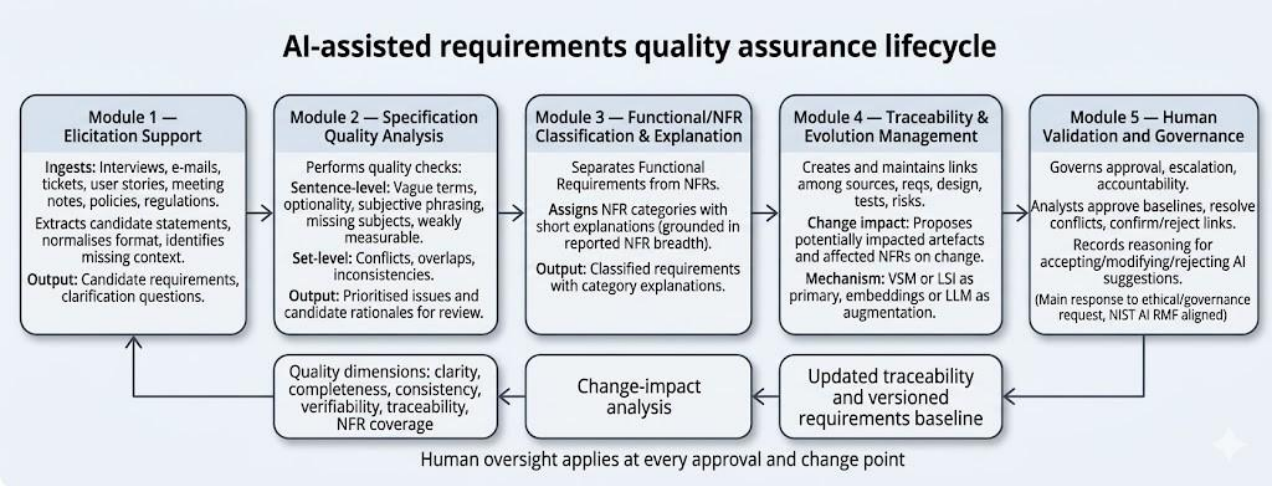

Figure 2. AI-assisted requirements quality assurance lifecycle.

5. Requirements quality model

The manuscript retains seven quality dimensions but now links them more explicitly to the framework modules: clarity, completeness, consistency, verifiability, traceability, NFR coverage, and change-impact visibility. Clarity is mainly supported by Modules 1 and 2; completeness by Modules 1, 2, and 3; consistency by Module 2; verifiability by Modules 2 and 3; traceability by Module 4; NFR coverage by Module 3; and change-impact visibility by Module 4 with approvals in Module 5.

Change-impact visibility deserves the most explicit clarification because the reviewer noted that it was less standard than the other dimensions. In this paper it means the degree to which a change in a source artefact

or requirement can be propagated into a visible and reviewable set of affected downstream artefacts, decisions, and quality constraints. It is therefore closely related to traceability but not identical to it: traceability stores or recovers the links, whereas change-impact visibility makes those links operational for review and decision-making.

Table 2 below uses recent NFR evidence to make the quality model more concrete. Rather than treating NFRs as a single residual category, the framework adopts six broad NFR classes synthesised by De Martino and Palomba (2025): accuracy, efficiency, maintainability, resiliency, sustainability, and usability. That classification is adapted here as a practical elicitation and analysis aid rather than copied wholesale.

6. Methodology and concrete prototype instantiation

To address the earlier criticism that the methodology was vague, the revised paper adopts an explicit design-science framing informed by Peffers et al. (2007) and a concrete, implementable prototype architecture. The methodology has four stages.

Stage 1 — Problem formulation and objective definition. The framework targets five concrete failure modes repeatedly highlighted in the literature and in practice: ambiguous wording, inconsistent requirements sets, under-specified NFRs, weak traceability, and poor change visibility. The design objective is not full automation, but measurable improvement in review efficiency and requirements quality under analyst oversight.

Stage 2 — Artefact design. A practical prototype can be instantiated with three complementary layers: (a) an instruction-tuned LLM for extraction, ambiguity explanation, and analyst-facing rationales; (b) a smaller task-specific classifier for FR/NFR multi-label classification and lightweight batch scoring; and (c) a traceability engine that combines classical IR baselines with embedding-based reranking. This blend is defensible because the literature suggests different strengths for different tasks: LLMs excel at explanation and heterogeneous text understanding, while IR baselines remain competitive and interpretable for trace-link recovery (Wan et al., 2025; Rejithkumar and Anish, 2025).

Stage 3 — Demonstration using published artefacts and evidence. Because this revised manuscript does not report new proprietary experiments, demonstration is provided through an illustrative banking scenario and a secondary empirical synthesis from published studies. That synthesis is not treated as proof that the full framework has already succeeded; rather, it grounds the feasibility of each module and makes the framework's assumptions explicit.

Stage 4 — Future evaluation blueprint. A future implementation should evaluate four core tasks: ambiguity/conflict detection, FR/NFR classification, traceability recovery, and change-impact recommendation. Suitable quantitative metrics include precision, recall, macro-F1, mean average precision, and precision@k. Suitable human factors include review effort, analyst agreement, explanation usefulness, and trust. Figure 3 summarises this evaluation logic.

The methodology is intentionally explicit about what has and has not been done. The artefact is designed and justified; a full end-to-end implementation remains future work. That honest scope definition is preferable to implying results that have not been produced.

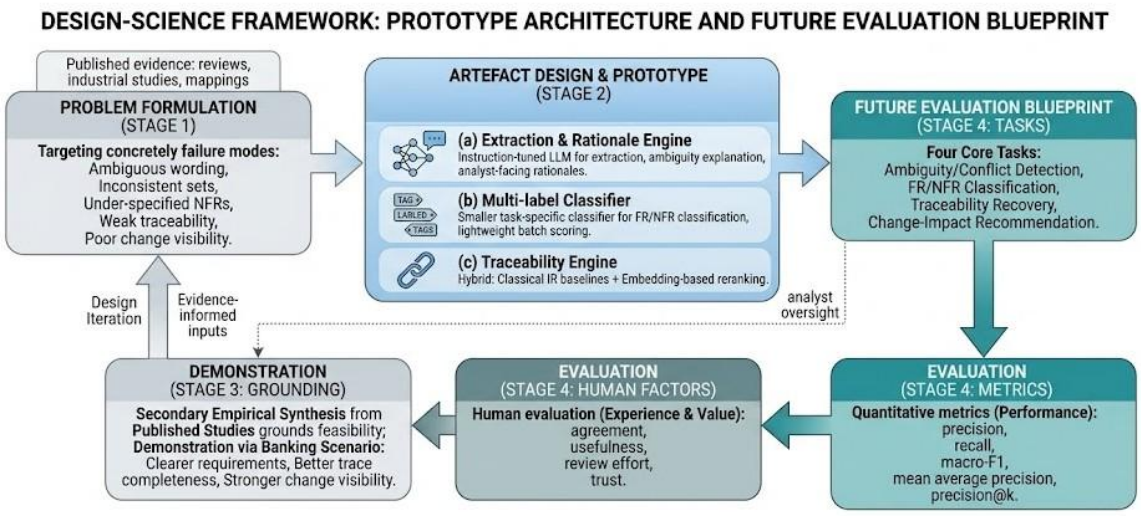

Figure 3. Design-Science Framework: Prototype Architecture and Future Evaluation Blueprint

7. Secondary empirical synthesis from published studies

This section directly addresses the reviewer's request for empirical grounding and for tables informed by published work. Table 1 synthesises the strongest recent evidence signals that motivate and constrain the framework. The key point is not that any one prior paper solves the full problem. Rather, the prior literature provides modular evidence that ambiguity support, NFR classification, traceability baselines, and analyst-facing explanations are all empirically active and relevant.

8. Illustrative scenario: regulated digital banking

Consider a regulated digital-banking programme whose requirements arise from compliance memos, business workshops, incident tickets, usability feedback, and release-planning meetings. The project faces several familiar RE problems: regulatory statements are abstract, user stories are concise to the point of incompleteness, NFRs such as auditability and availability are often implicit, and change requests arrive continuously as controls evolve.

In such a setting, Module 1 extracts candidate requirements and questions from source artefacts. Module 2 flags weak phrases such as "fast response", "appropriate security", or "intuitive dashboard", together with missing operational criteria. Module 3 classifies the hidden quality constraints into categories such as security, privacy, performance, availability, accountability, and usability, supplying short explanations so analysts can judge whether the category assignment is sensible. Module 4 connects regulatory clauses to requirements, tests, and change tickets, enabling impact analysis when a rule changes. Module 5 ensures that no requirement or trace update is committed without analyst approval and rationale capture.

This scenario is deliberately practical: it shows how the framework reduces ambiguity and increases change visibility without presuming that AI outputs are self-validating. It also clarifies why traceability and NFR-awareness are not optional add-ons but central to regulated software quality.

9. Discussion

The revised manuscript supports three discussion points. First, the literature now makes it difficult to argue that LLMs are merely speculative in RE. The evidence base has matured enough to justify concrete framework choices, especially for ambiguity analysis and NFR explanation. Secondly, the same literature also makes clear that task-level success is not the same as lifecycle maturity. Industrial evidence exists, but most studies still stop short of integrated workflow validation. Thirdly, governance remains a differentiator. A journal-ready framework paper should explain not only what AI can do, but also who remains accountable when requirement decisions are accepted into baselines, designs, tests, and release plans.

The framework therefore contributes by organising existing evidence into a lifecycle architecture rather than by introducing a single new algorithm. That is a legitimate and useful contribution when the evidence base is rich enough to support synthesis but still fragmented enough to need integration.

An additional implication concerns ethics and responsible use. Because the workflow may process interviews, e-mails, and policy documents, privacy and confidentiality controls are necessary. Because LLMs may hallucinate or overgeneralise, human review is mandatory at baseline and change decisions. Because training data may encode bias, explanation and audit trails should be retained when AI suggestions are used. The governance module explicitly operationalises these concerns rather than leaving them implicit.

10. Limitations and threats to validity

This paper is still limited by the absence of a new full end-to-end implementation. The secondary empirical synthesis cannot substitute for direct evaluation of the complete framework. In addition, the cited industrial studies operate in particular domains, and their results may not transfer directly to every RE context. Public datasets also underrepresent some enterprise requirements characteristics, while proprietary artefacts create reproducibility challenges. Finally, AI model capabilities and prompt behaviours evolve quickly, which means that tool choices described here should be interpreted as architectural roles rather than fixed product commitments.

Nevertheless, these limitations are now explicit and bounded. The manuscript no longer presents evaluation as if it had already been completed; instead, it presents a stronger and more honest artefact with evidence-backed design decisions and a clear evaluation path.

11. Conclusion

This revised manuscript responds to the major-review feedback by strengthening the literature review, clarifying the design-science methodology, adding concrete prototype choices, incorporating secondary empirical evidence from recent published studies, improving organisation and British-English style, expanding the reference section, and embedding figures and tables that make the argument easier to follow. The result is a more mature framework paper.

The central claim remains that requirements quality in AI-assisted environments should be organised as a lifecycle discipline rather than as a collection of disconnected automation tools. By integrating elicitation support, specification analysis, NFR-aware classification, traceability and evolution management, and explicit human governance, the framework offers a credible basis for future empirical work and for practical experimentation in industry. The next step is straightforward: implement the blueprint and test it in one or more industrial settings with task-level and workflow-level evaluation.

Table 1. Published empirical evidence motivating the framework

Study	Empirical signal	Implication for this manuscript
Zadenoori et al. (2025)	Systematic literature review of 74 primary studies (2023–2024). Most studies focus on elicitation and validation; most evaluations remain controlled; workflow integration and industrial use are limited.	Justifies a lifecycle framework that explicitly integrates modules rather than proposing another isolated task solution.
Hemmat et al. (2025)	Systematic review of LLM use in RE over 2020–2024. Reports growing adoption, but persistent issues with hallucination, domain specificity, validation, and practical guidance.	Supports the need for human-in-the-loop governance and a methodology section that is more concrete about validation and analyst oversight.
Bashir et al. (2025)	Industrial study on three datasets (179, 265, and 219 requirements). Ten-shot prompting improved ambiguity-classification performance by 20.2% over zero-shot prompting; practitioner explanation rating averaged 3.84/5.	Provides direct empirical support for explanation-augmented ambiguity analysis and for keeping explanations visible to analysts instead of returning only labels.
Wan et al. (2025)	Systematic mapping of IR-based requirements traceability methods using 40 primary studies selected from 2,052 publications. Identifies 32 enhancement strategies, 53 datasets, and 9 evaluation metrics.	Supports retaining interpretable IR baselines as the backbone of traceability recovery, with newer AI methods used as augmentation rather than wholesale replacement.
De Martino and Palomba (2025)	Systematic literature review of 130 studies. Identifies 31 NFRs grouped into six classes and 26 software-engineering challenges for ML-enabled systems.	Shows that NFR support must be richer than simple FR/NFR separation and that requirements-quality frameworks must be able to handle evolving and AI-specific NFRs.
Rejithkumar and Anish (2025)	Industrial NICE tool combines GPT-4o-generated explanations with smaller fine-tuned models for NFR identification, classification, and explanation; evaluation includes F1 and human assessment.	Supports combining LLM reasoning with lighter-weight classification components for scalable analyst support.

Table 2. Evidence-backed NFR classes adapted for the framework

NFR class	Representative requirements	Why the class matters for RE quality
Accuracy	System-level predictive correctness and threshold suitability	Useful when requirements concern acceptable error boundaries, acceptance criteria, and risk tolerances in AI-enabled systems.
Efficiency	Performance, capacity, stability, scalability	Important for clarifying measurable operational expectations that are often written vaguely in stakeholder language.
Maintainability	Replaceability, retrainability, reproducibility, transferability, reusability, traceability, adaptability	Relevant when requirements evolve and when models or components must be changed without destabilising the wider system.

Resiliency	Robustness, reliability, security, safety, privacy, availability	Supports explicit treatment of failure tolerance, protection, and continuity requirements rather than leaving them implicit.
Sustainability	Cost, energy consumption, fairness, ethics, accountability	Brings social, economic, and environmental trade-offs into elicitation and change analysis.
Usability	Interpretability, usability, imperceptibility / acceptable experience	Ensures the framework surfaces stakeholder-facing qualities such as understandability and trust, not just technical performance.

Note. The six classes are adapted from the synthesis reported by De Martino and Palomba (2025), which groups 31 NFRs into six broad clusters for ML-enabled systems. The present table rephrases that evidence for requirements-quality use rather than reproducing the source taxonomy verbatim.

Table 3. Concrete prototype choices and evaluation measures by framework module

Module	Concrete implementation choice	Primary outputs	Example evaluation measures
Elicitation support	Prompt-based extraction from source artefacts with schema-constrained output	Candidate requirements, clarification questions	Precision/recall of extraction; analyst acceptance rate
Specification quality analysis	LLM-assisted rule-and-rationale checking for ambiguity, inconsistency, and missing measurable criteria	Issue flags, explanations, review queue	Precision/recall of issue detection; explanation usefulness
FR/NFR classification	Smaller multi-label classifier with explanation support; optional LLM fallback for low-confidence cases	FR/NFR labels, NFR subclasses, explanations	Macro-F1; category coverage; reviewer agreement
Traceability and evolution	VSM/LSI or other IR baseline with embedding reranking and analyst confirmation	Candidate trace links, impact suggestions	MAP, precision@k, trace completeness, change-impact recall
Human validation and governance	Approval checkpoints, rationale capture, audit log	Accepted/rejected suggestions, governance evidence	Review effort, turnaround time, trust, override patterns

References

Bashir, S., Ferrari, A., Abbas, M., Saadatmand, M., Enoiu, E. P., Bohlin, M., and Lindberg, P. (2025). Requirements Ambiguity Detection and Explanation with LLMs: An Industrial Study. Proceedings of ICSME 2025 Industry Track.

Casamayor, A., Godoy, D., and Campo, M. (2010). Identification of non-functional requirements in textual specifications: A semi-supervised learning approach. Information and Software Technology, 52(4), 436–445. https://doi.org/10.1016/j.infsof.2009.10.010

Cleland-Huang, J., Gotel, O. C. Z., Hayes, J. H., Mäder, P., and Zisman, A. (2014). Software traceability: Trends and future directions. Proceedings of the on Future of Software Engineering, 55–69. https://doi.org/10.1145/2593882.2593891

Dalpiaz, F., Dell'Anna, D., Aydemir, F. B., and Çevikol, S. (2019). Requirements classification with interpretable machine learning and dependency parsing. 2019 IEEE 27th International Requirements Engineering Conference (RE), 142–152. https://doi.org/10.1109/RE.2019.00025

De Martino, V., and Palomba, F. (2025). Classification and challenges of non-functional requirements in ML-enabled systems: A systematic literature review. Information and Software Technology, 179, 107678.

Gotel, O. C. Z., and Finkelstein, A. C. W. (1994). An analysis of the requirements traceability problem. Proceedings of the First International Conference on Requirements Engineering, 94–101.

Hemmat, A. H. A., Sharbaf, M. S. M., Kolahdouz-Rahimi, S. K., Lano, K., and Tehrani, S. Y. (2025). Research directions for using LLM in software requirement engineering: a systematic review. Frontiers in Computer Science, 7, 1519437. https://doi.org/10.3389/fcomp.2025.1519437

ISO/IEC/IEEE. (2018). ISO/IEC/IEEE 29148:2018 Systems and software engineering — Life cycle processes — Requirements engineering. International Organization for Standardization / IEEE.

ISO/IEC. (2023). ISO/IEC 25010 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Product quality model. International Organization for Standardization.

Jia, H., Morris, R., Ye, H., Sarro, F., and Mechtaev, S. (2025). Automated Repair of Ambiguous Natural Language Requirements. arXiv:2505.07270.

NIST. (2023). Artificial Intelligence Risk Management Framework (AI RMF 1.0). National Institute of Standards and Technology. NIST AI 100-1.

Peffers, K., Tuunanen, T., Rothenberger, M. A., and Chatterjee, S. (2007). A design science research methodology for information systems research. Journal of Management Information Systems, 24(3), 45–77. https://doi.org/10.2753/MIS0742-1222240302

Petersen, K., Vakkalanka, S., and Kuzniarz, L. (2015). Guidelines for conducting systematic mapping studies in software engineering: An update. Information and Software Technology, 64, 1–18.

Porter, D., DeFranco, J. F., and Laplante, P. A. (2025). Requirements Specification Automated Quality Analysis: Past, Present, and Future. Computer, 58(1), 101–104. https://doi.org/10.1109/MC.2024.3480629

Rejithkumar, G., and Anish, P. R. (2025). NICE: Non-Functional Requirements Identification, Classification, and Explanation Using Small Language Models. Proceedings of the 2025 IEEE/ACM International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP).

Villamizar, H., Escovedo, T., and Kalinowski, M. (2021). Requirements Engineering for Machine Learning: A Systematic Mapping Study. 2021 47th Euromicro Conference on Software Engineering and Advanced Applications (SEAA). https://doi.org/10.1109/SEAA53835.2021.00013

Wan, H., He, X., Deng, Y., and Wang, B. (2025). A systematic mapping study of information retrieval-based requirements traceability methods. Information Processing & Management, 62(6), 104287.

Zadenoori, M. A., Dąbrowski, J., Alhoshan, W., Zhao, L., and Ferrari, A. (2025). Large Language Models (LLMs) for Requirements Engineering (RE): A Systematic Literature Review. arXiv:2509.11446.