2026 European Conference on Computing in Construction
Corfu, Greece
July 12–15, 2026

EXPLAINABLE AI FOR REQUIREMENT TRACEABILITY IN CONSTRUCTION SYSTEMS ENGINEERING

Timon Kuijper, Ranjith K. Soman, Tong Wang, and Sander van Nederveen
Delft University of Technology, Delft, the Netherlands

Abstract

Requirement verification in integrated contract models, such as Design and Construct, remains a manual, document-centric, and error-prone activity. While technical verification through Automated Code Compliance has advanced significantly, process requirements are typically verified through human inspection of textual documentation. This research addresses this gap by developing a novel artefact architecture that integrates general-purpose Large Language Models, the RASE mark-up language, and Knowledge Graphs to automate traceability between contractual obligations and contractor deliverables. Tested through iterative Design Science Research, the final prototype achieved a precision of 1.000 and accuracy of 0.968, offering a transparent, "white-box" framework for contractual auditability.

Introduction

Requirements traceability – the ability to track relationships between requirements and project deliverables – is essential in systems engineering and contract management for validating that project outcomes meet the specified requirements from start to finish (SixSigma, 2025). In large construction projects, the contractor must demonstrate that the delivered work complies with the Employer's requirements. Ensuring every requirement is satisfied (and providing evidence for it) is critical to project success in terms of cost, time, and quality (Hoehne, 2014). Traditionally, traceability of process-oriented requirements in construction is maintained manually through document management and version control systems (Guru, 2025). This manual approach is time-consuming, error-prone, and difficult to scale, especially given the growing volume of digital project documents and the involvement of many stakeholders. As a result, full end-to-end traceability is often impractical in current practice, and important links between requirements and evidence may be missed or lost.

Recent advances in Artificial Intelligence (AI), particularly Large Language Models (LLMs), offer new opportunities to automate requirements traceability by automatically recognizing and maintaining semantic links between textual artifacts. Research on AI-based traceability to date has focused largely on automated code compliance checking for technical requirements and on contract analysis tools, but there is a notable gap in applying AI to link process requirements with verification evidence in construction projects (Karim-Jallow et al., 2014; Ismail et al., 2017; Zheng et al., 2025). No prior work, to the authors' knowledge, has systematically explored a zero-shot LLM approach for automatically verifying process requirements in design-and-construct contracts by linking them to evidence in contractor documents. Moreover, the black-box nature of powerful LLMs raises concerns about transparency and explainability of AI-generated traceability links.

This paper addresses the above gap by presenting an AI-assisted requirement traceability approach that leverages a general-purpose LLM combined with a knowledge graph to automatically link contractual process requirements to corresponding evidence in project documents. The approach is designed to improve the efficiency and accuracy of traceability while also enhancing transparency (through a formal knowledge graph representation of the trace links). We developed and evaluated a prototype tool following a Design Science Research methodology, with iterative refinement and expert feedback. In a case study, the prototype achieved high accuracy in identifying valid requirement-evidence links, indicating that such an AI-assisted traceability system can significantly reduce manual effort and error in construction compliance verification.

The remainder of the paper is structured as follows. Background section reviews related literature on requirements traceability and AI applications in construction, identifying the research gap. Methods section explains the Design Science Research method used to develop and evaluate the solution. Implementation section describes the AI-assisted traceability tool, including its architecture and components. The following section presents the results of evaluating the tool's performance and usability. The final section discusses the findings, implications for research and practice, limitations, and future work

Literature review

Requirements traceability in construction

Requirements traceability refers to the ability to link requirements to corresponding artifacts that demonstrate their implementation or verification (Madni and Sievers, 2018). In construction projects, particularly within a Systems Engineering framework, traceability ensures

that contractual requirements are systematically verified against design documents, management plans, and other deliverables (SixSigma, 2025). Maintaining these links provides transparency, supports impact analysis, and reduces the risk of overlooked obligations.

In practice, traceability in construction remains largely manual. Engineers typically rely on traceability matrices, spreadsheets, or document annotations to map requirements to evidence (Guru, 2025). This process is time-consuming and prone to omissions, especially in large Design & Construct projects involving extensive documentation (Karim-Jallow et al., 2014). Process-oriented requirements, which describe how work must be conducted rather than what must be built, are particularly challenging to verify because evidence is often embedded in lengthy textual documents rather than structured models. As a result, complete and continuously updated traceability is difficult to achieve in current industry practice.

AI approaches to compliance and traceability

Recent advances in natural language processing and Large Language Models (LLMs) have stimulated research into automated traceability and compliance checking. In software engineering, information retrieval and machine learning techniques have been used to recover trace links between requirements and code or design artifacts (Rempel and Mäder, 2016). More recently, retrieval-augmented generation approaches using LLMs have demonstrated improved trace link recovery performance (Hey et al., 2025). In the architecture, engineering and construction (AEC) domain, AI applications have primarily focused on automated technical code compliance checking using rule-based systems, ontologies, or BIM-integrated workflows (Ismail et al., 2017; Yang et al., 2023). Parallel research has explored contract analysis using knowledge graph-enhanced LLMs (Zheng et al., 2025). While these studies show the potential of combining semantic representations with LLM reasoning, they do not directly address automated verification of contractual *process* requirements through evidence linking.

The RASE (Requirement-Applicability-Selection-Exception) methodology provides a structured way to extract normative statements from regulatory and contractual texts (Hjelseth and Nisbet, 2011). RASE has been shown to support automated processing of construction regulations by isolating enforceable requirements. In parallel, knowledge graph representations have been proposed to model requirements and compliance relationships in a transparent and queryable manner (Wang and El-Gohary, 2023). These approaches suggest that combining semantic structuring techniques with AI-driven text understanding could enable more robust verification workflows.

Research gap

Despite progress in automated compliance checking and LLM-supported document analysis, limited research has addressed the automated verification of contractual process requirements in construction projects. Existing approaches either focus on technical design compliance, require significant rule engineering, or rely on supervised training data. The use of a zero-shot LLM approach, combined with a knowledge graph to produce auditable requirement–evidence mappings for process requirements, remains largely unexplored.

This paper addresses this gap by developing and evaluating an AI-assisted verification tool that automatically links contractual process requirements to supporting evidence in contractor documentation, using a hybrid LLM and knowledge graph architecture.

Method

This study followed the Design Science Research Methodology (DSRM) (Peffers et al., 2007) to develop and evaluate the AI-assisted traceability tool. Design Science Research (DSR) is a problem-solving paradigm that iteratively designs, builds, and evaluates artifacts (in this case, a software tool) to address identified problems, thereby contributing new knowledge. The research process was structured according to the DSRM steps proposed by Peffers *et al.* (Peffers et al., 2007):

(1) Problem Identification and Motivation: Through preliminary literature review and expert interviews, we established that manual requirement traceability in large construction projects is inefficient and error-prone, and that no adequate automated solution exists. The need for an AI-based approach was justified by the increasing volume of documentation and the potential of LLMs to interpret complex texts. We defined the core problem as: "How can AI support automated traceability between contractual process requirements and contractor documentation in construction projects?"

(2) Define Objectives for the Solution: Based on the problem analysis, we set functional and non-functional objectives for the artifact. Key functional objectives included: automatically identifying requirement statements in contract documents; extracting evidence of compliance from contractor documents; and establishing trace links between them. Non-functional objectives included a high degree of accuracy and recall in link identification (targeting at least 90 % accuracy and 90% recall to be comparable with manual quality), high precision (to minimize false positives, as an incorrect assertion of compliance is unacceptable), and usability/transparency for end-users. These objectives were operationalized into measurable success criteria (e.g., performance metrics and a usability score). We also aimed for the solution to be zero-shot, meaning it should work without project-specific training, relying on prompt engineering and general LLM capabilities, to facilitate adoption in new projects.

(3) Design and Development: We designed an artifact architecture that integrates a general-purpose LLM with a knowledge graph. The design and implementation details are described in following section. Importantly,

we adopted an iterative development approach (typical in DSR): the artifact went through multiple design iterations, each adding improvements or new features to overcome shortcomings identified in the previous iteration. For example, after an initial prototype (Iteration 1) showed low recall, we refined the prompt and graph constraints to ensure all requirements would be captured (Iteration 2). Further iterations introduced a confidence scoring mechanism and automated consistency checks. Throughout this process, design decisions were informed by both literature (e.g., using RASE mark-up and SHACL constraints as derived from prior work) and practitioner input.

(4) Demonstration: We demonstrated the artifact using a real-world case study. The case involved a Design and Construct infrastructure project in the Netherlands. The input documents included a Project Requirements document (a contract appendix listing process requirements from the Employer) and a Project Management Plan (PMP) submitted by the contractor as evidence of how requirements are addressed. The tool was used to automatically generate traceability links between the requirements and the PMP. Domain experts (project engineers) observed the demonstration to validate that the identified links correspond to actual compliance evidence.

(5) Evaluation: We evaluated the artifact's performance and utility using both quantitative and qualitative methods. Quantitatively, we measured classical classification performance metrics for the trace link predictions: True Positives (TP), False Positives (FP), True Negatives (TN), False Negatives (FN), accuracy, precision, recall, and F_1 score (harmonic mean of precision and recall). These metrics were computed by comparing the AI-generated links against a ground truth traceability set established via manual verification of the case documents. We applied this evaluation after each major iteration of the artifact. Qualitatively, we gathered feedback on usability and usefulness through a focus group and a survey. After the final prototype was demonstrated, $N = 5$ construction engineering professionals used the tool on sample queries and then filled out a System Usability Scale (SUS) questionnaire. The SUS provides a standardized score out of 100 for perceived usability. We also collected open-ended feedback about what users liked and what could be improved. This multi-faceted evaluation approach follows guidance from DSR literature that emphasizes not only technical performance of artifacts but also their fit with user needs and contexts (Venable et al., 2016).

(6) Communication: Finally, we document the problem, artifact, and findings in this paper and in a companion thesis report, to disseminate the knowledge to both academic and practitioner audiences. The iterative DSR approach ensured that the research was grounded in solving a relevant practical problem while also contributing general insights (e.g., how LLMs can be combined with knowledge graphs for traceability).

Implementation of the AI-assisted traceability tool

System overview

The developed artifact is a prototype tool that supports verification of contractual process requirements by automatically producing an auditable *requirement–evidence mapping* (See Figure 1). The system combines (i) a Large Language Model (LLM) for natural-language understanding and evidence extraction and (ii) a knowledge graph (KG) to store requirement entities, evidence snippets, and their links in a structured and queryable form. The KG acts as a traceability backbone: each verification decision is represented as a link between a requirement node and one or more evidence nodes, including provenance metadata (e.g., document and section identifiers).

LLM-driven requirement extraction and evidence identification

The tool operates in two stages. First, it extracts requirement statements from the contractual requirements document. Second, for each extracted requirement, it searches for supporting evidence in the contractor document(s) (e.g., project management plan) and returns a quoted excerpt plus a document location reference. Prompting is designed to be conservative: the LLM is instructed to only propose a link when it can point to explicit textual evidence; otherwise, it returns ''no evidence found''. This design choice prioritizes avoiding false positives (incorrectly asserting compliance) over maximizing coverage.

Knowledge graph representation

All outputs are stored as RDF-like triples in a knowledge graph with a lightweight ontology. Core classes include `Requirement` and `Evidence`, with a predicate such as `hasEvidence` linking a requirement to an evidence excerpt (See Figure 2). Evidence nodes store provenance fields (document name, chapter/section identifiers) to support auditability and facilitate queries such as: ''Which requirements have no evidence?'' or ''Which document sections satisfy the most requirements?''

Design iterations (build–evaluate cycles)

The artifact was developed using a Design Science Research (DSR) approach with multiple **design iterations**. In this paper, an **iteration** is defined as one complete *build–evaluate cycle*: (1) implement a concrete change to the artifact (e.g., prompt structure, document chunking strategy, evidence provenance requirements, or KG validation rules), and (2) re-run the same case study evaluation against a fixed ground truth to quantify the impact on link-quality metrics (accuracy, precision, recall, and F_1). Each iteration was motivated by observed error patterns (false negatives/false positives) in the previous iteration and aimed to improve verification completeness while maintaining conservative, audit-friendly outputs. Table 1 summarizes the iterations and their key design changes.

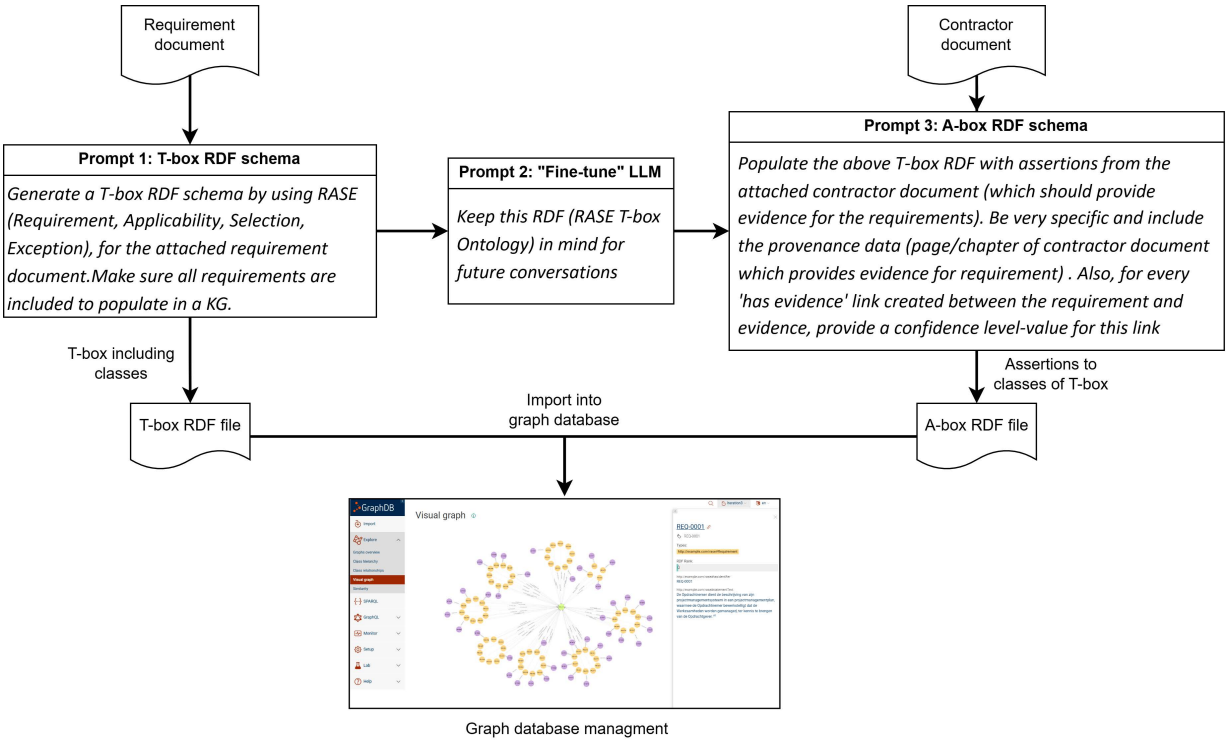

Figure 1: AI-assisted requirement traceability workflow combining T-box generation, LLM-based A-box population, and graph database integration.

Results

We evaluated the performance of the AI-assisted traceability tool on the case study project. The quantitative results demonstrate that through iterative refinement, the artifact achieved a high level of accuracy in identifying valid traceability links between requirements and evidence. Table 2 summarizes the classification performance metrics over four iterations of the artifact development. In the first iteration, the tool's recall was very low (around 17%), indicating it missed the majority of actual requirement-evidence links; accuracy was also low (17.2%). However, precision was perfect (100%) even in that iteration, meaning that although the tool found few links, those it did find were correct (no false positives). This high precision can be attributed to the conservative approach in the LLM prompting as described earlier. The low recall and accuracy in Iteration 1 highlighted the need to capture more links (reduce false negatives). By Iteration 2, after improving the prompt to ensure no requirements were skipped, recall jumped to 67.9% and accuracy to 73.4%. Further enhancements in Iteration 3 (such as better handling of partial matches and refining evidence extraction) yielded about 73.6% recall. Finally, Iteration 4 introduced a confidence scoring mechanism and additional knowledge graph constraints to catch missing links; the tool then achieved 96.6% recall and 96.8% accuracy, essentially meeting the target of $> 90\%$ on both. Precision remained 100% throughout all iterations, meaning the system did not produce any incorrect links in our test dataset. The F_1 score, which balances precision and recall, improved dramatically from 29.3% in Iteration 1 to 98.3% in Iteration 4, indicating a well-balanced and robust performance by the end.

In absolute terms, for the final iteration, out of 64 total requirements in the test contract, the tool correctly linked 57 requirements with relevant evidence found in the PMP (True Positives) and correctly identified that 3 requirements had no evidence in that document (True Negatives, which in this context means those requirements were indeed not addressed in the PMP and likely require evidence from other documents or were out of scope). There were 2 False Negatives, i.e., requirements for which the tool failed to find the existing evidence in the PMP. Notably, there were 0 False Positives in any iteration, so the precision is 1.0 (the tool never claimed an evidence link where there was none). By comparing with manual verification, we found that most of the false negatives in early iterations were due to the LLM not recognizing evidence phrased very differently from the requirement wording. After ontology enhancements and additional prompt context, only 2 requirements remained unlinked (false negatives) in the final iteration. Those cases were ones where the evidence was implicit or scattered in the document, indicating a limit to the LLM's ability to infer compliance without explicit statements (a topic for future improvement). Beyond the binary performance metrics, the tool provided qualitative insights. For example, as mentioned, it identified inconsistencies in the contractor document's chapter numbering while extracting evidence. In one case, the LLM correctly pointed to a section titled "Risk management"

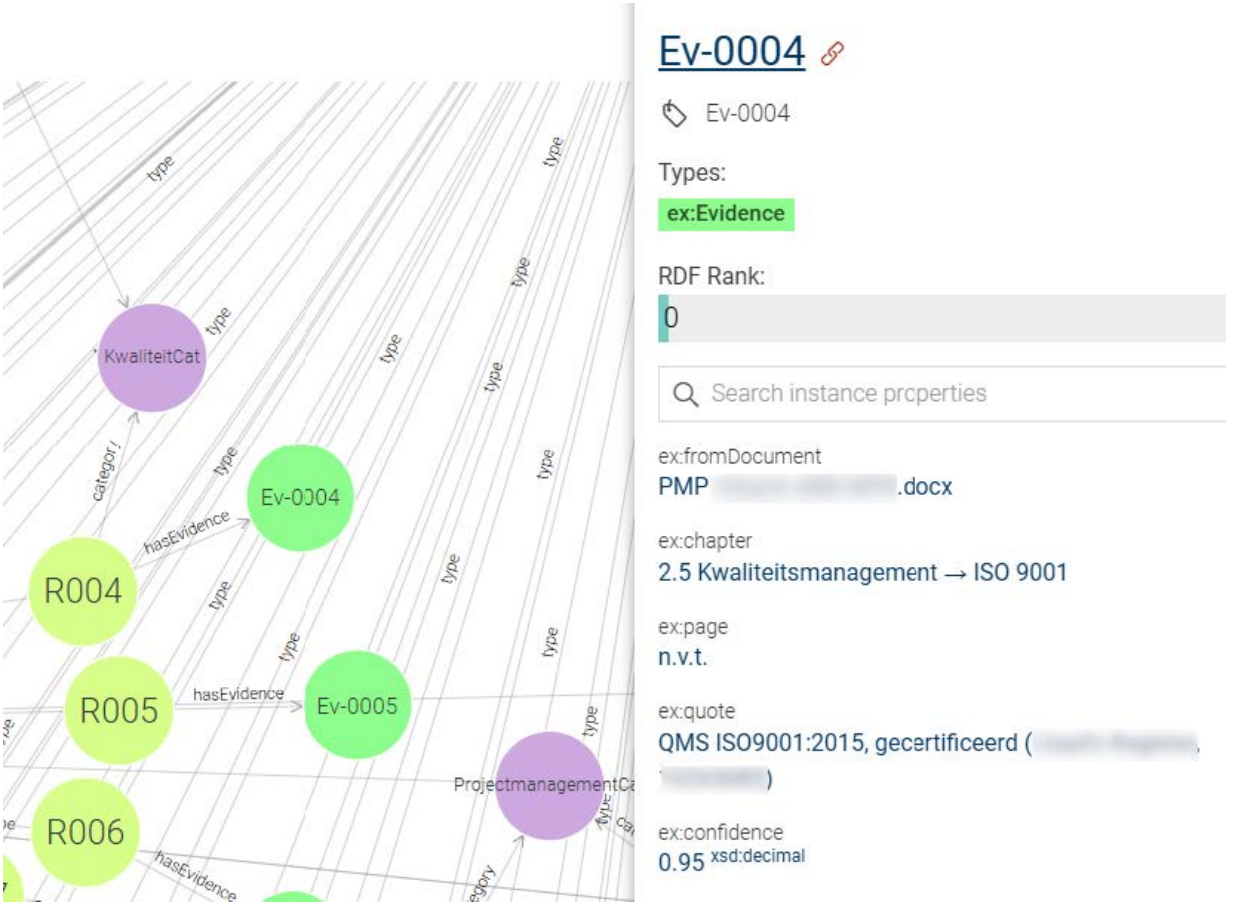

Figure 2: Knowledge Graph Representation

as evidence for a requirement on risk coordination, but flagged it under chapter 4 whereas the contractor's table of contents had it under chapter 3. This led to discovering a documentation error by the contractor. Such findings suggest the AI tool not only checks compliance but can also help improve document quality and consistency. We also evaluated the usability and acceptance of the tool via the focus group and SUS questionnaire. The average SUS score given by the five participants was **78.5**. According to standard SUS interpretations (Lewis, 2018; Sauro, 2018), this score falls in the 80th percentile, corresponding to a rating above "good" (approximately a "B+" on a grading scale). An SUS above 68 is generally considered above average; thus, our result indicates that users found the prototype reasonably easy to use. They particularly appreciated the ability to click a requirement and immediately see the evidence text, which one participant noted "saves a lot of time compared to flipping through PDFs manually." Some participants were initially skeptical about AI "reading" their documents, but after seeing the accuracy of results, they expressed increased trust. The focus group discussion revealed that users valued the knowledge graph's transparency — it reassured them that the AI wasn't making arbitrary guesses but actually linking to specific document sections. However, the participants also pointed out areas for improvement. In a few instances, the evidence text returned by the tool was quite lengthy; users suggested highlighting the exact sentences that matter. They also desired a clearer indication of confidence for each link (even though the system internally had confidence scores, the front-end only showed the link or no link). This feedback aligns with our own observation that a confidence threshold could be used to only show high-confidence links by default, with lower-confidence suggestions flagged for human review. Overall, the results demonstrate that the AI-assisted traceability tool can dramatically improve the traceability process: it achieved near-perfect precision and very high recall in linking requirements to evidence in a complex real document, and it did so in a matter of minutes (the end-to-end runtime for analyzing 64 requirements against a 150-page PMP was under 10 minutes on an LLM API, which is negligible compared to days of manual work). The high usability score further suggests that with a bit more refinement, practitioners would be willing to adopt such a tool in their workflow.

Discussion

The successful demonstration of the prototype confirms that leveraging LLMs for requirement traceability in construction is not only feasible but highly effective under the tested conditions. In this section, we reflect on the impli-

Table 1: Design iterations of the AI-assisted requirement verification tool.

Iter.	Implemented change	Design focus
1	Baseline pipeline combining LLM-based requirement extraction, evidence identification, and knowledge graph storage of requirement–evidence links.	Establish end-to-end feasibility of automated verification workflow.
2	Revised prompt structure to enforce complete requirement coverage and systematic section-wise evidence scanning.	Improve completeness of requirement handling and structured evidence search.
3	Refined evidence prompts to require explicit textual quoting and provenance references; improved document chunking strategy.	Strengthen grounding of links and standardize evidence extraction format.
4	Integrated additional knowledge graph validation constraints and final prompt refinements for handling implicit references and edge cases.	Increase structural robustness and consistency of requirement–evidence mapping.

Table 2: Traceability link classification performance per iteration (case study evaluation)

Metric	Iteration 1	Iteration 2	Iteration 3	Iteration 4
Accuracy	0.172	0.734	0.781	0.968
Precision	1.000	1.000	1.000	1.000
Recall	0.172	0.679	0.736	0.966
F_1-score	0.293	0.809	0.848	0.983

cations of these findings, discuss the contribution of this work in context, and outline limitations and future work.

Implications and contributions

The research provides evidence that an AI-assisted approach can significantly reduce the manual effort required for verifying process compliance in construction projects. By automatically generating traceability links, the tool addresses a pain point for contractors, who typically must allocate considerable time to compile compliance matrices for each requirement. The near-100% precision means that users can trust the links that are present, focusing their attention on verifying missing links (false negatives) rather than double-checking every provided link for correctness. In practice, this flips the verification process from a manual search for evidence to an AI-assisted review of suggested evidence, potentially saving time and catching errors that humans might overlook due to fatigue or oversight. One of the experts in our focus group mentioned that the tool "would allow junior engineers to do in hours what used to take an experienced engineer days," hinting at productivity gains and knowledge transfer benefits. From a scientific perspective, this work contributes a novel integration of technologies: it combines an advanced natural language understanding model (Open AI GPT-5) with semantic web techniques (RASE markup, RDF knowledge graph, SHACL constraints) to achieve a level of performance that neither approach could easily attain alone. Pure LLM-based solutions might achieve good recall but suffer from hallucinations, while pure rule-based/ontology solutions struggle with the ambiguity of natural language. Our hybrid approach demonstrates a synergistic effect: the knowledge graph and constraints guide the LLM and provide structure, while the LLM provides the flexible understanding needed to interpret diverse requirement and evidence texts. This approach is generalizable beyond the specific case; it could be applied to other domains where requirements (or regulations) need to be traced to evidence (e.g., safety certification, quality assurance in manufacturing, or even legal compliance in finance). Thus, a broader contribution is showing how zero-shot LLM capabilities can be harnessed in a controlled, explainable way for traceability tasks. Furthermore, this research adds to the Design Science knowledge base by illustrating the application of DSRM in the construction IT context. We followed a structured iterative design-evaluate cycle and have documented the evolution of the artifact. The improvement across iterations (Table 2) serves as a case example of how setting concrete objectives (like recall > 90%) and continuously

evaluating against them leads to a more effective artifact. It also underscores the importance of evaluation at multiple levels: our evaluation was not just technical (metrics) but also involved end-user feedback, which is crucial for any AI tool to be practically viable. The high SUS score in the final evaluation is as important a result as the high F_1-score; both technical performance and user acceptance must align for successful adoption.

Limitations

Despite the encouraging results, there are several limitations to acknowledge. First, the evaluation was conducted on a single case project (albeit a fairly large and representative one). While the results show the tool can work extremely well on that case, projects can vary. Different contract structures, requirement phrasing styles (especially if not in English, whereas our LLM was primarily English-trained), or evidence document types might pose new challenges. For instance, some projects might have requirements scattered across multiple documents or require linking not just one contractor report but a chain of documents (like design drawings or test reports). Our current prototype handles one requirements document and one evidence document; scaling to multiple heterogeneous sources will require further development. Second, the reliance on a third-party LLM (OpenAI's GPT 5) introduces practical issues such as cost and data privacy. Running GPT-5 on large documents can be expensive, though we tried to optimize prompts to stay within reasonable token limits. More critically, sending potentially sensitive project documents to an external API can be problematic for companies concerned with confidentiality. This limitation could be mitigated by on-premise LLM solutions or fine-tuned smaller models in the future. There is ongoing research to distill or adapt large models for specific tasks which could be explored to reduce dependency on external APIs. Another limitation is that, while precision was perfect in our tests, we cannot guarantee it will always remain so as we generalize. We tuned the system to be conservative to avoid false positives; this worked for the case data, but a very differently structured document might confuse the LLM to produce an incorrect link. If a false positive (hallucination) does occur, it could mislead users into thinking a requirement is satisfied when it is not, which is a serious risk. Therefore, deploying this in a real project would require thorough validation and perhaps maintaining a human in the loop who at least spot-checks the evidence for each requirement. Additionally, our method of splitting the evidence document by sections and sequentially querying the LLM might miss evidence that is, say, a combination of two sections or implied across the document. More sophisticated retrieval (like using vector embeddings to fetch the top relevant passages for a requirement) could enhance the evidence search phase. Finally, regarding the evaluation method: our ground truth traceability links were established via manual review by the researchers (and double-checked with the help of the domain experts). There is some degree of subjectivity in determining what counts as "evidence" for a requirement, especially for process-oriented requirements that might not have a binary satisfied/not satisfied state. We treated the contractor's own verification statements in the PMP as the evidence of compliance. It is possible that in some cases the contractor's documentation might claim compliance where an external auditor might disagree. In our metrics, we assumed the contractor's documentation combined with our judgment is the truth. A more rigorous validation in the future could involve independent auditors verifying the links as well.

Future work

Building on the prototype's success and the limitations identified, several avenues for future work are evident. One important next step is to generalize and test the tool on additional projects. We plan to apply the approach to other case studies, possibly in different domains (for example, rail or highway projects, or even outside construction such as aerospace systems engineering) to evaluate how well the LLM handles different writing styles and terminologies. This could be complemented by fine-tuning the LLM on a corpus of requirement-evidence pairs (if such data can be accumulated) to see if a fine-tuned model could achieve similar results more efficiently. Another future improvement is enhancing the explainability of the AI decisions. Integrating techniques from explainable AI (XAI) could allow the system to provide a justification for each trace link, beyond just the raw text. For example, the tool could highlight which keywords or phrases in the requirement matched with the evidence, or even output a natural language justification like "Requirement X is satisfied in section Y of the PMP where it states '...' which corresponds to the required process." Such features would further increase user trust and understanding of the AI's reasoning. Introducing a confidence threshold for suggestions is also worth exploring. We observed that by iteration 4, the few missed links could potentially be discovered by allowing the AI to be more "creative" (in iteration 4 it might have still erred on the side of caution). If we allow a slightly lower precision in exchange for 100% recall, we could then flag the uncertain links for human review. A hybrid approach could be: the tool marks some links as "high confidence" (auto-verified) and others as "low confidence – needs human check." This way, no potential evidence is missed, and the human verifier's effort is focused only on the questionable cases. From a technical standpoint, integrating a vector-based retrieval step could improve evidence discovery. Instead of or in addition to scanning document sections sequentially, we can use embeddings to find the most semantically similar passages to each requirement and present those to the LLM. This retrieval augmented generation approach (Hey et al., 2025) could help when evidence is not explicitly a near copy of requirement wording. Finally, expanding the knowledge graph to capture more context could be valuable. Currently, our KG links requirements to evidence found in one document. In

reality, evidence of compliance could span multiple documents (e.g., a test report plus a certification). The ontology could be extended to represent complex verification chains (for instance, a requirement is satisfied by a procedure which is evidenced by a test, which is documented in a report). This aligns with the broader vision of Model-Based Systems Engineering (MBSE) where every requirement can be traced to model elements, test cases, and results. Our work is a step in that direction by automatically populating a knowledge base; future work might integrate with MBSE tools or requirement management databases to pull in more types of trace links.

Conclusion

This research demonstrates a practical AI-assisted solution to requirement traceability in construction projects. By combining the strengths of LLMs in understanding unstructured text with the rigor of knowledge graphs for information organization and validation, we achieved a tool that can automate a large portion of the traceability process with high accuracy and user acceptance. The Design Science approach ensured the artifact was iteratively improved to meet the desired performance. The resulting prototype not only contributes a new method for managing requirements in construction but also provides a blueprint for tackling similar problems in other domains where requirements verification is crucial. As AI technologies continue to evolve, we expect such hybrid human-AI systems to become an integral part of the systems engineering toolkit, augmenting human capabilities in managing complexity and ensuring compliance in large-scale projects.

References

Guru (2025). Documentversiebeheer: Een uitgebreide gids voor projectmanagers. `https://www.getguru.com/nl/reference/document-version-control`.

Hey, T., Fuchss, D., Keim, J., and Koziolek, A. (2025). Requirements traceability link recovery via retrieval-augmented generation. In Proc. REFSQ (Requirements Engineering: Foundation for Software Quality), pages 381–397.

Hjelseth, E. and Nisbet, N. (2011). Capturing normative constraints by use of the semantic mark-up rase methodology. In Proc. CIB W78-W102 Conference, pages 1–10.

Hoehne, O. M. (2014). On motivating people to implement systems engineering: Getting from the necessary to the impossible. INCOSE International Symposium, 24(1):362–377.

Ismail, A. S., Ali, K. N., and Iahad, N. (2017). A review on bim-based automated code compliance checking system. In Proc. International Conference on Research and Innovation in Information Systems (ICRIIS), pages 1–6.

Karim-Jallow, A., Demian, P., Baldwin, A., and Anumba, C. (2014). An empirical study of the complexity of requirements management in construction projects. In Engineering, Construction and Architectural Management (Proc. ICE), volume 21, pages 505–531.

Lewis, J. R. (2018). The system usability scale: Past, present, and future. International Journal of Human-Computer Interaction, 34(7):577–590.

Madni, A. M. and Sievers, M. (2018). Model-based systems engineering: Motivation, current status, and research opportunities. Systems Engineering, 21(3):172–190.

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.

Rempel, P. and Mäder, P. (2016). Preventing defects: The impact of requirements traceability completeness on software quality. IEEE Transactions on Software Engineering, 43(8):777–797.

Sauro, J. (2018). 5 ways to interpret a sus score. `https://measuringu.com/interpret-sus-score/`.

SixSigma (2025). Matrix voor traceerbaarheid van vereisten: Een complete gids voor projectsucces.

Venable, J., Pries-Heje, J., and Baskerville, R. (2016). Feds: A framework for evaluation in design science research. European Journal of Information Systems, 25(1):77–89.

Wang, X. and El-Gohary, N. (2023). Deep learning-based relation extraction and knowledge graph-based representation of construction safety requirements. Automation in Construction, 147:104696.

Yang, M., Zhao, Q., Zhu, L., Meng, H., Chen, K., Li, Z., and Hei, X. (2023). Semi-automatic representation of design code based on knowledge graph for automated compliance checking. Computers in Industry, 150:103945.

Zheng, C., Wong, S., Su, X., Tang, Y., Nawaz, A., and Kassem, M. (2025). Automating construction contract review using knowledge graph-enhanced large language models. Automation in Construction, 175:106179.