Available Online: 31 July 2026 DOI: 10.54254/2977-3903/2026.35813

When may LLM outputs influence software requirements? A human-in-the-loop governance framework

Chuanjin Zhu

College of Artificial Intelligence and Computer Science, Northwest Normal University, Lanzhou, China

202331607248@nwnu.edu.cn

Abstract. Large Language Models (LLMs) are increasingly used to review, clarify, rewrite, and trace software requirements. These applications create a governance problem that output-quality assessment alone cannot resolve: a fluent proposal may rely on inadmissible evidence, alter stakeholder intent, introduce unsupported specificity, or imply an organizational commitment that the model has no authority to make. Existing work on retrieval-augmented generation, controlled natural language, formal verification, human oversight, and Artificial Intelligence (AI) governance supplies relevant controls, but it does not specify the procedural status of an individual LLM proposal relative to a controlled requirements artifact. This article develops an artifact-centered, human-in-the-loop framework in which the permitted influence of a proposal is the primary object of governance. The framework combines five governance functions—governed evidence, bounded context construction, controlled LLM analysis, pre-commit verification, and accountable human approval—with four artifact-influence states: A0 advisory observation, A1 evidence-linked candidate, A2 verified recommendation, and A3 approved and committed change. Its central theoretical claim is that output quality, evidential legitimacy, verification status, and authority to commit a change are distinct properties and should not be collapsed into a single confidence judgment. Seven falsifiable hypotheses translate the model into measurable comparisons involving source admissibility, context leakage, unsupported specificity, semantic drift, reviewer agreement, unreviewed changes, governance cost, and organizational maturity. Human review is treated as both a necessary decision boundary and a potential source of automation bias, anchoring, and fatigue. The framework is conceptual rather than empirically validated and provides a basis for controlled experiments, field studies, and longitudinal evaluation.

Keywords: large language models, requirements engineering, human-in-the-loop governance, software requirements, traceability, artifact influence

1. Introduction

Requirements Engineering (RE) defines what a software-intensive system is expected to achieve, the constraints under which it must operate, and the conditions by which its acceptability will be assessed. Requirements are typically derived from multiple sources, including stakeholder interviews, organizational policies, regulations, contracts, issue records, operational data, and prior design decisions. These sources may differ in authority, currency, scope, confidentiality, and approval status. Consequently, transforming source

material into a controlled requirements baseline is not simply a matter of improving textual clarity. It is a governed process in which proposed requirements must remain traceable to relevant evidence and consistent with established decision rights.

Large Language Models (LLMs) extend the range of RE activities that can be supported computationally. They can summarize stakeholder material, classify requirements, identify ambiguity, compare versions, formulate clarification questions, propose candidate requirements, and assist traceability [1-5]. Recent studies have examined their use in user-story assessment, requirements coverage review, ambiguity and completeness analysis, backlog-item analysis, requirements enhancement, and initial requirement generation [4-10]. These capabilities make LLMs plausible components of an RE toolchain, particularly where large volumes of heterogeneous natural-language material must be analyzed.

The same capabilities also create distinctive risks. An LLM can add a plausible but unsupported threshold, change the strength of an obligation, omit an exception, assign responsibility to the wrong actor, or apply a rule outside its valid scope. Such changes may be difficult to detect because the resulting text can be fluent, specific, and professionally written. Linguistic improvement therefore does not necessarily imply that a proposed requirement is evidentially legitimate or procedurally safe.

This distinction becomes consequential when model output moves beyond informal assistance and begins to affect a controlled artifact, such as a user story, acceptance criterion, contractual requirement, compliance record, trace link, or baselined specification. At this boundary, the relevant question is not merely whether the output is clear or plausible. The question is whether the proposed change is supported by admissible project evidence, has passed appropriate checks, and has been accepted by a person with the organizational authority to commit it.

Existing approaches address parts of this problem. Retrieval-Augmented Generation (RAG) can connect outputs to external sources; controlled-language techniques can constrain representation; formal methods can strengthen selected verification tasks; human review can retain decision authority; and general AI-governance frameworks can address provenance, privacy, accountability, and lifecycle risk [11-14]. However, these approaches do not, individually or collectively, define a requirement-level pathway through which an LLM-generated observation becomes eligible to modify a controlled baseline.

The governing question is: under what conditions may an LLM-generated proposal influence a controlled requirements artifact? The framework developed here answers that question by distinguishing five governance functions from four artifact-influence states. Its core claim is that a proposal may be linguistically strong yet remain ineligible to change a requirements baseline because evidence, verification, and organizational authority are separate conditions. Seven research hypotheses specify how this claim can be tested. The study is conceptual and normative; it defines constructs and expected mechanisms without claiming empirical validation.

2. Conceptual background

2.1. Requirements as governed artifacts

A requirement is more than a grammatically correct sentence. It is a claim about a system capability, stakeholder need, constraint, quality attribute, or acceptance condition. Its legitimacy depends not only on wording but also on the source from which it was derived, the owner of that source, its current status, its applicable scope, and the authority by which it was approved. Requirements engineering standards accordingly emphasize quality characteristics, traceability, and controlled change [15].

Consider the statement "the dashboard should load quickly." The statement identifies a performance concern but does not yet constitute a verifiable performance requirement. Workload, device class, network conditions, metric, percentile, and target must be specified before implementation and acceptance can be assessed. An LLM may identify these missing variables and formulate questions, but it cannot legitimately select their values unless an authorized source supports them.

This example reveals two dimensions that are often conflated. Textual quality concerns whether a requirement is clear, complete, consistent, and verifiable. Evidential legitimacy concerns whether its substantive content is supported by sources that are authorized and applicable. A statement may be precise yet unsupported, while a legitimate stakeholder statement may remain incomplete and unsuitable for direct implementation. A governance framework for LLM-supported RE must preserve both dimensions rather than treating fluent output as a proxy for justified content.

2.2. LLM capabilities and limits

Transformer-based language models generate text from statistical patterns learned during training and from the prompt and context supplied at inference time [16]. This mechanism supports classification, summarization, question generation, defect identification, trace-link ranking, and structured rewriting. Earlier RE research established automated methods for non-functional-requirement classification [17], interpretable requirements classification [18], reusable public requirements datasets [19], user-story quality assessment [20], and terminological-ambiguity detection [21]. More recent work has examined LLM-supported ambiguity and completeness analysis, backlog-item analysis, requirements enhancement, requirement generation, and AI-based requirements-quality assessment [6].

These studies justify treating LLMs as potentially useful analytical components, but they do not remove the need for governance. Model outputs can vary with prompt wording, context order, model release, decoding settings, retrieval configuration, and connected tools. Repeated agreement may indicate stability without establishing correctness. RAG may retrieve an obsolete or unauthorized source. Controlled syntax may preserve a false premise. A structurally valid JSON object may contain unsupported semantics, and a citation may exist without supporting the claim to which it is attached.

The resulting governance problem is therefore located at the relationship between the proposed artifact change and its evidential and organizational basis. Model capability, output fluency, format validity, source retrieval, and human approval are related but distinct properties. A defensible process must make these distinctions explicit.

2.3. The artifact-level governance problem

RAG, controlled language, verification, human oversight, and AI-governance guidance regulate different parts of an LLM-supported workflow [11]. Their controls remain incomplete at the artifact boundary because they do not assign a procedural status to an individual proposal. Retrieval can identify a source without establishing that the source is admissible. Verification can detect selected defects without authorizing a change. Human review can preserve authority while still being affected by anchoring, automation bias, or fatigue.

A requirement-level model must therefore make five decisions explicit: what may count as project evidence; what evidence is relevant and permitted for the task; what the model may produce when support is incomplete; what checks a candidate must pass; and who may commit the change. The five governance layers implement these decisions, while the A0-A3 states record the influence currently permitted for each proposal. Table 1 defines the constructs required for this separation.

Table 1. Core constructs

Construct	Definition in this article
Controlled requirements artifact	A requirement, user story, acceptance criterion, trace link, model element, or specification record that is versioned, reviewed, or subject to change control.
Governed evidence	A source whose owner, version, lifecycle status, access classification, effective date, and applicable scope are known.
Artifact influence	The permitted procedural effect of an LLM output on a controlled artifact, ranging from an advisory observation to an approved and committed change.
Prompt contract	A task-specific specification of authorized inputs, prohibited behavior, required output fields, uncertainty handling, and evidence references.
Pre-commit verification	Checks performed before a proposed change may be submitted for baseline approval.
Accountable approval	A recorded decision by a person with the relevant organizational role to accept, reject, modify, defer, or request clarification.

3. Conceptual research design

This study adopts a model-oriented conceptual research design. Conceptual articles should explain how prior knowledge is selected, combined, and used to construct new concepts or relationships [22]. Conceptual framework analysis likewise emphasizes explicit definitions and a coherent network of relationships among constructs [23]. Following these principles, the present study synthesizes knowledge from requirements engineering, LLM-supported software engineering, evidence retrieval, controlled representation, verification, and AI governance to construct a requirements-specific governance model.

The study is not a systematic literature review. It does not claim exhaustive coverage, estimate effect sizes, or report a formal search-and-screening protocol. Sources were selected for their conceptual role from five domains: RE standards and quality principles; studies of LLM-supported requirements tasks; RAG and controlled-language research; formal methods for requirements; and human-oversight and AI-governance guidance [1]. Each domain contributes a different component. RE standards establish artifact-quality and change-control needs. Task-level studies identify capabilities and failure modes. RAG and controlled-language research inform evidence and representation controls. Formal methods inform verification. AI-governance guidance informs accountability, privacy, provenance, and lifecycle management.

Framework construction proceeded through five analytical stages. First, the unit of analysis was defined as an LLM-generated proposal capable of influencing a controlled requirements artifact. Second, recurring failure conditions were abstracted from the selected literature and established RE practice. Third, each failure condition was translated into a design principle describing what a defensible process should preserve. Fourth, the principles were organized into governance functions and linked to influence states that specify what a proposal is permitted to do. Fifth, the framework was examined for construct clarity, internal consistency, boundary conditions, and empirical testability.

The derivation follows a single conceptual chain. A recurring problem condition identifies what can go wrong; a design principle identifies the property to be protected; a governance function specifies where that property is operationalized; and an influence state defines the procedural effect currently permitted for the proposal. Seven research hypotheses translate the expected mechanisms and trade-offs into empirical comparisons. Figure 1 summarizes this logic.

The epistemic status of the resulting framework is deliberately limited. It is a normative conceptual model intended to organize controls and generate testable claims. It is not an implemented architecture, a validated maturity model, or evidence that the proposed controls necessarily produce net benefits in practice.

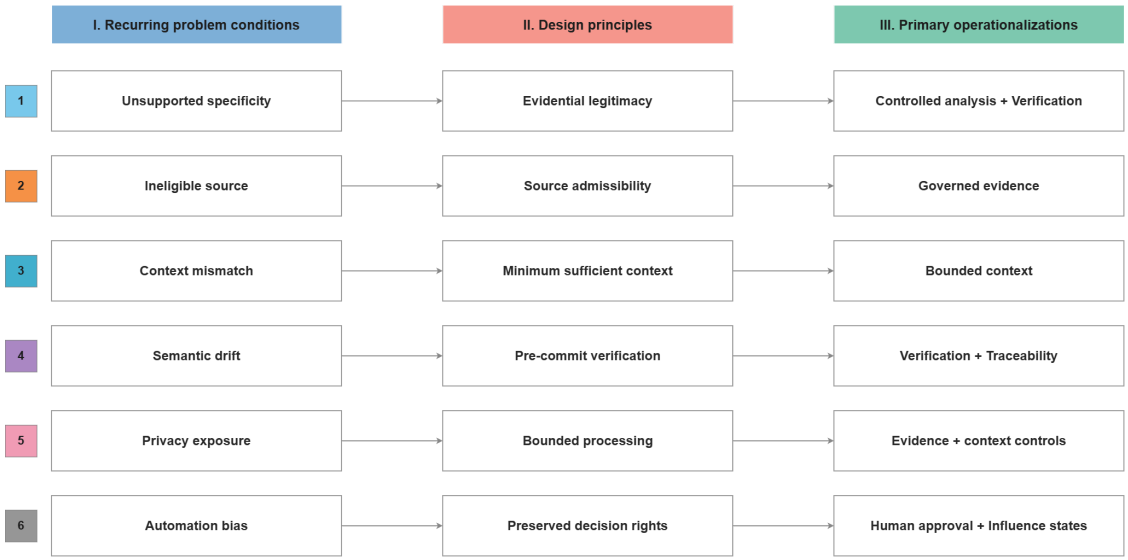

Figure 1. Conceptual derivation from recurring problem conditions to design principles, governance functions, influence states, and research hypotheses

4. From recurring risks to design principles

4.1. Unsupported specificity

LLMs can make incomplete statements appear actionable by adding plausible actors, thresholds, interfaces, or obligations. In RE, these additions may convert unresolved uncertainty into a false organizational commitment. The corresponding principle is evidential legitimacy: a specific value or rule may enter a candidate requirement only when admissible evidence supports it. When support is insufficient, the appropriate output is an observation, a clarification question, or an explicit abstention rather than an invented completion.

4.2. Inadmissible sources

Requirements repositories contain drafts, superseded decisions, expired policies, local exceptions, and restricted material. Relevance-based retrieval does not reliably distinguish among these statuses. The corresponding principle is source admissibility: any source used as project fact should have identifiable ownership, version, lifecycle status, access classification, effective date, and applicable scope. A draft may remain visible for discussion while being ineligible to justify a baseline value.

4.3. Context mismatch

A larger context window is not automatically safer. Irrelevant material can introduce conflicting terminology, historical records can be mistaken for current policy, and restricted information can cross access boundaries. The corresponding principle is minimum sufficient context: the system should use the smallest authorized

evidence package that is adequate for the task. The principle does not require context minimization at any cost; it requires a controlled balance among coverage, relevance, permission, and temporal applicability.

4.4. Semantic drift

A rewrite can preserve the topic of a requirement while changing its obligation strength, actor, scope, conditions, exceptions, timing, or numerical values. Equivalent prompts may also produce materially different revisions. The corresponding principle is pre-commit verification: before a proposal is submitted for approval, the process should check structure, source support, terminology, modality, numerical values, conflicts, and semantic preservation. Formal methods may strengthen this function where suitable models and properties exist, but they are not assumed to be universally available.

4.5. Privacy and access exposure

Requirements repositories may contain personal data, commercial plans, vulnerabilities, architectural details, and contractual terms. The corresponding principle is bounded processing: classification, minimization, redaction, access control, approved deployment environments, and retention rules should constrain retrieval and generation. A semantically correct output remains unacceptable if it was produced through unauthorized disclosure.

4.6. Automation bias, review fatigue, and diffused responsibility

Fluent and highly specific model output can anchor reviewers, encourage automation bias, and receive less scrutiny than incomplete human notes. Repeated low-value warnings can also reduce attention over time, turning a nominal human gate into routine confirmation. Responsibility becomes further diluted when prompt designers, tool operators, domain experts, and artifact owners occupy different roles. The corresponding principle is preserved decision rights: models may produce observations, candidates, and verified recommendations, but an identifiable person with the relevant authority must decide whether a baseline changes. Preserving that right requires more than recording a final click; it requires review conditions that support independent judgment, manageable workload, explicit rationale, and traceable responsibility. As shown in Table 2, each recurring risk is linked to a design principle and a primary governance function.

Table 2. Recurring risks, design principles, and primary operationalizations

Risk condition	Requirements consequence	Design principle	Primary operationalization (s)
Unsupported specificity	Invented values or rules become false commitments.	Evidential legitimacy; clarification or abstention when support is insufficient.	Controlled analysis; pre-commit verification
Obsolete or unauthorized source	Correct-looking text relies on ineligible evidence.	Status-aware source admissibility.	Governed evidence
Context mismatch or leakage	Irrelevant, stale, or restricted material shapes the output.	Minimum sufficient, permission-aware context.	Bounded context construction
Semantic drift or instability	A rewrite changes modality, actor, scope, condition, timing, value, or exception.	Verification before commitment.	Pre-commit verification; traceability

Table 2. Continued

Privacy or access exposure	Sensitive information leaves an approved boundary.	Bounded processing through classification, minimization, redaction, and access control.	Governed evidence; bounded context
Automation bias, fatigue, or unclear responsibility	Reviewers anchor on fluent output, overlook defects, or approve without clear ownership.	Independent and accountable human decision rights under manageable review conditions.	Human approval; risk-tiered review; artifact-influence states

5. The five-layer governance framework

The six principles are operationalized through five governance functions. Each function narrows the proposals that may proceed: governed evidence establishes admissible sources; bounded context selects the authorized evidence package; controlled analysis constrains model behavior; pre-commit verification tests candidate eligibility; and accountable approval determines whether the baseline changes. These functions may be implemented in separate components or distributed across tools and roles. A cross-cutting traceability spine connects evidence, context, prompt, model, output, checks, decision, and baseline version. Figure 2 depicts the framework.

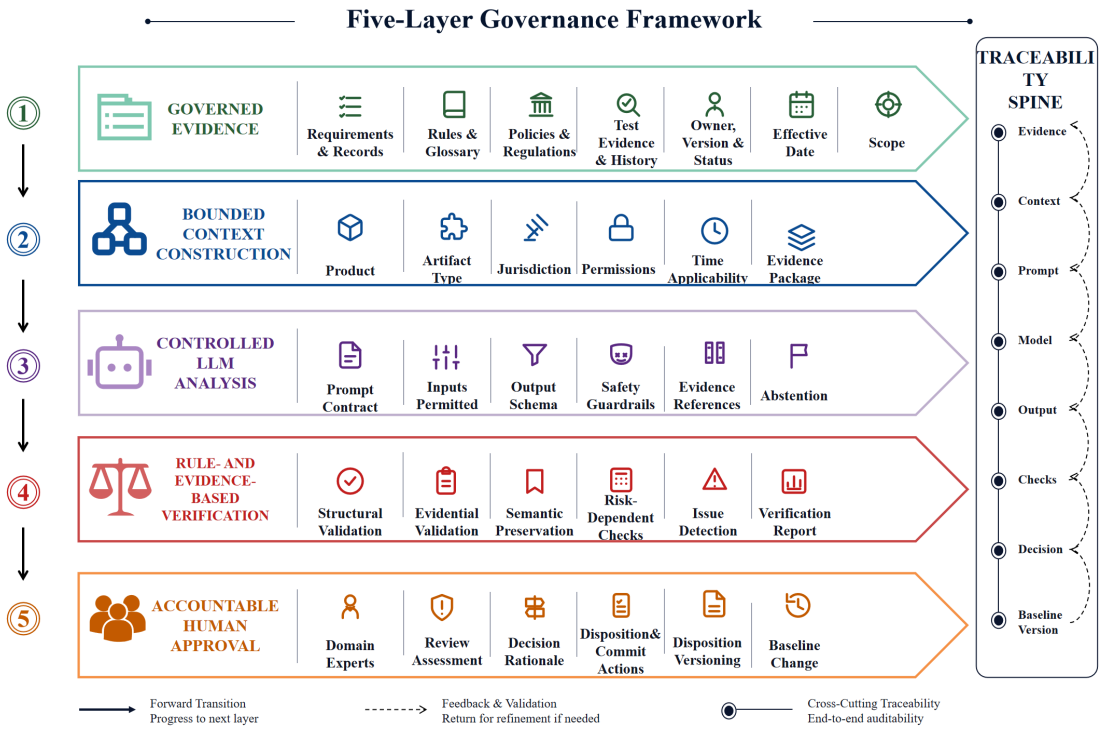

Figure 2. Five-layer governance workflow, feedback paths, and cross-cutting traceability

5.1. Layer 1: governed evidence

The first layer contains requirements, stakeholder records, business rules, glossary entries, policies, regulations, test evidence, and historical decisions that may support requirements work. Each record should have an owner, version, lifecycle status, access classification, effective date, and applicable scope. Draft, obsolete, revoked, and superseded material should be distinguishable from authoritative evidence. The

repository therefore acts as a control plane for what the model may treat as project fact rather than merely as a document store.

A practical evidence record may also include jurisdiction, product or release scope, confidentiality level, approval role, and exact citable spans. Admissibility is purpose-dependent. A draft performance target may support a clarification discussion while remaining ineligible to justify a baseline threshold. This distinction preserves useful visibility without confusing visibility with authority.

5.2. Layer 2: bounded context construction

The second layer constructs a task-specific context from governed evidence. Selection should consider project, product version, artifact type, jurisdiction, confidentiality, user permissions, and temporal applicability. Each context item should preserve its source identifier, exact span, version, and date so that later outputs can be checked against the evidence actually available to the model.

Bounded context serves two purposes. It reduces the influence of irrelevant, stale, or unauthorized material, and it makes the evidential basis of the analysis inspectable. The context package should therefore be versioned. A material change in retrieval should be treated as a new analytical run because it changes the basis on which the proposal was produced.

5.3. Layer 3: controlled LLM analysis

The model should receive a task-specific prompt contract rather than an open-ended instruction. The contract defines the model role, permitted inputs, defect taxonomy, required output fields, prohibited behavior, uncertainty handling, and evidence-reference requirements. Typical prohibitions include inventing numerical values, business rules, stakeholder roles, interfaces, jurisdictions, regulations, or security controls. The model should be allowed to abstain when the evidence does not support a completed revision.

Useful output fields include requirement identifier, issue type, problematic span, evidence references, explanation, proposed revision, clarification question, unresolved assumptions, confidence category, and verification flags. Structured output facilitates deterministic checks and exposes missing information. The raw output should be retained even when rejected because rejected proposals can reveal recurring failure modes and support audit reconstruction.

5.4. Layer 4: pre-commit verification

The fourth layer checks a candidate before it may be presented as a verified recommendation. Structural validation checks required fields, identifiers, schemas, and data types. Evidential validation checks whether cited sources exist, are admissible, and support the relevant claim. Semantic-preservation checks compare actor, modality, scope, conditions, exceptions, terminology, timing, and numerical values with the source requirement and supporting evidence. Optional formal analysis may be added when a project maintains invariants, contracts, state machines, temporal properties, or executable models [12, 13].

These checks should not be conflated. A valid schema does not imply semantic validity. A citation does not imply that the cited text entails the claim. A formally valid formula does not establish that the formula reflects stakeholder intent. Layer 4 is therefore intended to block predictable failure modes and produce explicit warnings, rejection reasons, or unresolved questions. It supports human judgment rather than replacing it.

5.5. Layer 5: accountable human approval

Requirements engineers, product owners, domain experts, security specialists, compliance personnel, and other accountable roles review proposals according to their consequences. Low-risk terminology

normalization may use a streamlined individual approval workflow, but every proposal that reaches A3 must receive a recorded decision from an authorized person. Changes involving safety, security, privacy, contracts, finance, or regulation should require role-appropriate specialist approval. A model confidence score cannot substitute for organizational authority.

The approval record should connect the original input, retrieved evidence, prompt and model version, raw output, verification result, reviewer identity and role, rationale, disposition, and final artifact version. The reviewer may accept, reject, modify, defer, or request clarification. Only an authorized acceptance may create a baseline change.

5.6. Traceability and feedback

Traceability is not a sixth sequential layer. It is a cross-cutting mechanism that supports reproducibility, audit, rollback, and organizational learning. Failed checks may return a case to context construction, controlled analysis, or stakeholder clarification. Human review may reveal that an evidence record is obsolete or that two approved rules conflict, causing the process to return to Layer 1. These feedback paths make the framework iterative without making the approval boundary ambiguous. Table 3 summarizes the objective, required records, and typical failure response for each governance layer.

Table 3. Layer objectives, required records, and typical responses

Layer	Primary objective	Required record	Typical failure response
1. Governed evidence	Establish admissible, current, access-controlled sources.	Owner, version, status, access, dates, scope, and citable spans.	Correct metadata; quarantine or exclude the source.
2. Bounded context	Construct a sufficient, relevant, authorized evidence package.	Retrieved items, filters, access decision, ranking rationale, and context version.	Refine the query, remove an item, or request missing evidence.
3. Controlled analysis	Constrain task, output, uncertainty, and prohibited behavior.	Prompt contract, structured raw output, evidence links, assumptions, and abstention status.	Revise the contract, regenerate, or escalate for clarification.
4. Verification	Reject malformed, unsupported, or semantically risky proposals.	Validation report, warnings, rejection reasons, and optional formal results.	Return to Layers 1-3, downgrade the state, or create a clarification item.
5. Human approval	Preserve stakeholder authority and accountability.	Decision, reviewer role, rationale, disposition, and baseline version.	Reject, modify, defer, or request further evidence.

6. Artifact-influence states and transition logic

6.1. States A0-A3

The five layers describe governance functions, whereas the A0-A3 label attaches to an individual proposed-change record. The states specify what that proposal is currently permitted to do relative to a controlled

artifact. Separating layers from states prevents model capability, workflow position, verification status, and organizational authority from being collapsed into a single score. The letter "A" denotes artifact influence.

A0 is an advisory observation. It may identify an issue, summarize evidence, suggest an exploratory interpretation, or formulate a clarification question, but it is not yet a formal change candidate. A1 is an evidence-linked candidate whose substantive elements are connected to admissible evidence and whose assumptions are exposed. A2 is a verified recommendation that has passed the configured structural, evidential, semantic, and risk-dependent checks. A3 is an approved and committed change accepted by an authorized person and incorporated into a new controlled artifact version. As shown in Table 4, the four states differ in both their minimum conditions and their permitted effects on the controlled artifact.

Table 4. Artifact-influence states A0-A3

State	Meaning	Minimum conditions	Permitted effect
A0 - Advisory observation	An issue, question, summary, or exploratory suggestion that is not yet a change candidate.	Task identity and raw output are recorded; evidence may be incomplete.	May inform discussion or create a clarification task; cannot modify the artifact.
A1 - Evidence-linked candidate	A proposed modification linked to admissible evidence and explicit assumptions.	Authorized sources are cited; required structure is present; unsupported additions are absent.	May enter formal review and verification; cannot enter the baseline.
A2 - Verified recommendation	A candidate that has passed the configured structural, evidential, semantic, and risk-dependent checks.	Verification report is complete; blocking warnings are resolved; traceability is intact.	May be submitted for accountable approval; no autonomous baseline write is allowed.
A3 - Approved and committed change	A verified recommendation accepted and committed by an authorized person.	Reviewer decision, role, rationale, and new baseline version are recorded.	The approved proposal is committed; the resulting artifact version becomes governed evidence for later work.

6.2. Forward transitions

An A0 observation becomes an A1 candidate only when the proposed change is within task scope, linked to admissible evidence, expressed in the required structure, and explicit about unresolved assumptions. Better wording alone is not sufficient. An A1 candidate becomes an A2 recommendation only after the required verification stack has been completed. The stack may vary by risk but should at least check source identity and status, terminology, modality, numerical values, and semantic consistency.

An A2 recommendation becomes an A3 approved and committed change only through an explicit decision by an authorized person. No confidence threshold, agent vote, or automated validator may substitute for this gate. The A2-to-A3 transition is therefore the point at which analytical support becomes an organizational commitment.

6.3. Downgrade, rejection, and revalidation

State progression is reversible. An A2 recommendation should be downgraded when a cited source is superseded, access rights change, a conflict is discovered, or the context package changes materially. If a

reviewer edits an A2 recommendation in a way that changes its meaning, scope, modality, value, condition, or exception, the revised proposal should return to A1 and be verified again.

A failed verification may return a repairable candidate to A1 or an unsupported proposal to A0. Human rejection closes or defers the proposal while preserving the evidence and rationale. Time-sensitive requirements should carry revalidation triggers linked to policy expiry, release changes, or regulatory updates.

An A3 record should not be silently overwritten. If later evidence requires correction, the organization should create a new controlled version and preserve the prior decision record. This rule protects auditability and prevents later changes from rewriting historical responsibility.

6.4. Risk-tier controls

Organizations should define which states and verification controls are appropriate for each artifact class. Low-risk clerical tasks, such as terminology normalization, may move quickly from A1 to A2 and use streamlined individual approval at the A2-to-A3 gate. Medium-risk functional requirements should require complete evidence and semantic checks. High-risk safety, security, privacy, contractual, or regulatory requirements may remain at A0 or A1 until specialist review and, where appropriate, formal analysis are available. The state model therefore supports differentiated control rather than imposing one cost profile on every task. Table 5 consolidates the forward-transition, downgrade, rejection, and revalidation rules.

Table 5. State-transition, downgrade, and revalidation rules

Transition or event	Required condition	Result
A0 to A1	Authorized evidence link, task scope satisfied, structured candidate, and exposed assumptions.	Candidate enters verification.
A1 to A2	All mandatory structural, evidential, semantic, and risk-dependent checks pass.	Recommendation becomes eligible for approval.
A2 to A3	Authorized reviewer accepts and records rationale and baseline version.	Proposal reaches A3 and the change is committed.
Verification failure	Blocking error, unsupported value, modality change, conflict, or invalid source.	Return to A1 for repair or A0 for clarification.
Evidence invalidation	A source becomes obsolete, revoked, inaccessible, or outside scope.	Downgrade dependent A1/A2 items and trigger revalidation.
Substantive human edit	Reviewer changes meaning, scope, modality, value, condition, or exception.	Return the revised proposal to A1 and verify again.
Post-approval correction	New evidence or a defect affects an A3 record or its committed artifact change.	Create a new version and preserve the historical record.

7. Illustrative application

7.1. Scenario and assumed evidence

This section uses a hypothetical order-management project to clarify the framework. The example is illustrative rather than empirical. Its evidence records, roles, requirement statements, and outcomes are assumed for conceptual demonstration and should not be interpreted as evidence that the framework improves performance or requirements quality.

The project begins with three stakeholder statements. R1 states that "the system shall generate order reports quickly." R2 states that "managers can approve large orders and the system sends an alert." R3 states that "customer information must be protected according to applicable regulations." Each statement reflects a legitimate concern, but none is sufficiently precise for implementation and verification without further evidence.

Five evidence records are assumed to be available. Their labels are descriptive rather than project identifiers. Their status is central to the example because the same content may be useful for discussion while remaining ineligible to support a baseline change. The assumed records and their conceptual roles are shown in Table 6.

Table 6. Assumed evidence records used in the illustrative scenario

Assumed record	Assumed status	Illustrative content	Conceptual role
Requirements-quality guideline	Approved	Performance requirements should identify conditions, workload, metric, and target.	Supports defect detection but does not supply a target value.
Performance target note	Draft	Suggests a response-time target under a peak-load profile.	May support discussion but cannot justify a baseline threshold.
Order-approval business rule	Approved	Orders above 500,000 currency units require approval by a regional sales manager; the order owner and finance queue should be notified within 60 seconds after approval or rejection.	Supports the actor, threshold, recipients, and timing for R2.
Security policy	Approved	Customer records are confidential; controls should address access, logging, encryption, and retention.	Supports the control scope but not a jurisdiction-specific legal obligation.
Compliance applicability note	Pending legal review	The applicable jurisdiction and regulatory instruments have not been confirmed.	Blocks regulation-specific baseline text.

7.2. Bounded context and prompt contract

A task-specific context package would include the approved requirements-quality guideline, the approved order-approval rule, the approved security policy, and the pending compliance-applicability note. The draft performance target could remain visible for discussion but would not be permitted to justify a baseline threshold. The context package would preserve source identifiers, exact spans, versions, statuses, and access decisions.

The prompt contract would instruct the model to review R1-R3 for ambiguity, singularity, verifiability, unsupported assumptions, and evidence conflicts. It would prohibit the invention of numbers, roles, interfaces, business rules, jurisdictions, regulations, and controls. When the evidence is insufficient, the model would be required to leave the proposed revision incomplete and formulate a clarification question. Required fields could include requirement label, issue type, problematic span, evidence reference, explanation, proposed revision, clarification question, assumptions, and verification flags.

7.3. Applying the A0-A3 states

For R1, the model may identify "quickly" as unverifiable and note the missing workload, metric, and approved target. Because the only target is contained in a draft record, the framework does not permit a numerical

rewrite. The output remains an A0 advisory observation and may create a clarification task for the product owner and performance engineer.

For R2, the approved business rule supplies an actor, threshold, recipients, and timing. The framework therefore permits two evidence-linked candidates: "Orders above 500,000 currency units shall require approval by a regional sales manager," and "Within 60 seconds after approval or rejection, the system shall notify the order owner and the finance queue." These proposals are A1 candidates because their substantive elements are linked to assumed admissible evidence.

For R3, the approved security policy supports the need for access control, logging, encryption, and retention. The pending compliance note states that the applicable jurisdiction and legal instruments have not been confirmed. The framework therefore permits a clarification question but not a regulation-specific rewrite. The proposal remains at A0.

Verification of the R2 candidates would compare their structure, evidence links, actor, threshold, recipients, timing, and obligation strength with the approved business rule. If the elements are preserved, the candidates may advance to A2. A variant that changes "shall notify" to "should notify" fails the modality check and returns to A1. A variant of R3 that names a specific regulation fails evidential validation and returns to A0 because the assumed evidence does not support the legal claim.

At the human gate, an authorized requirements engineer and product owner may approve the verified R2 candidates, move the proposal records to A3, and commit the changes. R1 remains unresolved until an approved performance target exists. R3 remains an open clarification until legal applicability has been established. The example demonstrates how evidence status and influence state jointly determine what the output is permitted to do. Table 7 summarizes the resulting state assignments and transitions.

Table 7. Illustrative state assignments and transitions

Requirement or variant	Initial classification	Framework-based assessment	Disposition	Resulting state
R1	A0 observation	No approved performance target is available.	Request evidence and clarification.	A0
R2 candidates	A1 candidates	Actors, threshold, recipients, timing, and modality are supported.	Verify and submit to authorized roles.	A2, then A3 if approved
R2 weakened variant	A1 candidate	"Shall" is changed to "should."	Return for correction and reverification.	A1
R3	A0 clarification	Jurisdiction and applicable instruments remain unresolved.	Assign to compliance and security owners.	A0
R3 regulation-specific variant	A1 candidate	The evidence does not support the legal claim.	Reject or reformulate as a clarification question.	A0

8. Research hypotheses and measurement

The framework makes directional claims that can be rejected by observation. Each hypothesis specifies a comparison, a measurable outcome, and a result that would count against the predicted relationship. Tests should evaluate complete workflows under a common evidence corpus, task set, model configuration, and approval policy. Model accuracy alone is insufficient because the dependent variables concern artifact influence, reviewer behavior, and governance cost.

H1 (source admissibility). Requirements candidates produced with status-aware evidence grounding will have a lower inadmissible-source rate than candidates produced with relevance-only retrieval. The inadmissible-source rate is the proportion of cited or substantively used sources that are draft, obsolete, unauthorized, outside scope, or outside their effective period. H1 is not supported if the status-aware condition produces an equal or higher rate.

H2 (bounded context). Permission-aware minimum-sufficient context will reduce unauthorized-source exposure and irrelevant-source use relative to broad-context retrieval while maintaining defect recall within a prespecified non-inferiority margin. Unauthorized exposure is measured by the number of restricted spans made available to the model; irrelevant-source use is the proportion of output claims influenced by sources judged inapplicable to the task. H2 is not supported if exposure or irrelevant use does not decrease, or if defect recall falls beyond the stated margin.

H3 (observation-first analysis). An observation-first workflow that requires abstention when evidence is insufficient will produce a lower unsupported-specificity rate than a direct-rewrite workflow. Unsupported specificity is the proportion of generated actors, thresholds, interfaces, jurisdictions, obligations, or exceptions that cannot be entailed from admissible evidence. H3 is not supported if the two workflows do not differ or if the observation-first workflow produces more unsupported additions.

H4 (pre-commit verification). Candidates processed by structural, evidential, and semantic verification will have a lower semantic-drift rate among proposals submitted for approval than candidates controlled only by prompting. Semantic drift includes changes to actor, modality, scope, condition, timing, numerical value, or exception that are not supported by evidence. H4 is not supported if the verification stack fails to reduce this rate.

H5 (explicit influence states). Reviewers using explicit A0-A3 labels and transition records will show higher agreement about a proposal's permitted effect and greater audit-reconstruction accuracy than reviewers using an equivalent workflow without explicit state labels. Agreement may be measured with Krippendorff's alpha or Fleiss' kappa; reconstruction may be measured by the proportion of evidence, verification, decision, and baseline links correctly recovered within a fixed period. H5 is not supported if neither outcome improves.

H6 (human A3 gate). A workflow that reserves the A2-to-A3 transition for an authorized human will produce fewer unreviewed baseline changes than an automated-commitment workflow, but it will require more reviewer minutes per committed change and a longer median decision cycle. H6 is not supported if unreviewed changes do not decrease; its predicted cost trade-off is not supported if review effort and cycle time do not increase.

H7 (organizational moderation and net value). The effect of the full governance framework on net governance value will be more positive in organizations with higher evidence-governance maturity, clearer approval authority, and stronger tool integration, and the difference will be larger for high-consequence artifact classes. Net governance value is defined as estimated avoided loss and audit benefit minus setup, verification, review, delay, and maintenance costs. H7 is not supported if the interaction between framework use and organizational readiness is absent, negative, or unrelated to artifact consequence. Table 8 consolidates the comparison conditions, primary outcomes, and results that would be inconsistent with each hypothesis.

Table 8. Research hypotheses and operational tests

Hypothesis	Primary comparison	Primary outcome (s)	Result inconsistent with the hypothesis
H1	Status-aware grounding versus relevance-only retrieval	Inadmissible-source rate; stale-source rate	Equal or higher inadmissible-source rate under status-aware grounding
H2	Minimum-sufficient context versus broad context	Unauthorized spans exposed; irrelevant-source use; defect recall	No reduction in exposure/use, or recall below the non-inferiority margin
H3	Observation-first with abstention versus direct rewriting	Unsupported-specificity rate; clarification yield	Equal or higher unsupported-specificity rate
H4	Verification stack versus prompt-only control	Semantic-drift rate; blocking-error escape rate	No reduction in drift or escaped blocking errors
H5	Explicit A0-A3 states versus implicit workflow status	Inter-rater agreement; audit-reconstruction accuracy and time	No improvement in agreement or reconstruction
H6	Human-only A3 gate versus automated commitment	Unreviewed changes; reviewer minutes; cycle time	No reduction in unreviewed changes
H7	Full framework across different readiness and risk levels	Net governance value; escaped defects; total governance cost	No positive readiness interaction or no stronger effect for high-consequence artifacts

A controlled experiment can test H1-H5 with seeded requirements defects and adjudicated evidence labels. H6 requires workflow-level comparison because its benefits and costs occur at the approval boundary. H7 is better examined through multisite field studies or longitudinal deployments in which readiness, artifact consequence, cost, and escaped defects vary meaningfully. All studies should report both error reduction and the resources required to achieve it.

9. Discussion

9.1. The core theoretical claim

The framework treats artifact influence, not output quality, as the object of governance. Four properties must remain distinct: linguistic quality, evidential legitimacy, verification status, and organizational authority. A proposal may be clear yet unsupported, supported yet unverified, or verified yet unauthorized. The five layers perform the required control functions; the A0-A3 states record what an individual proposal is currently permitted to do. This separation is the paper's central theoretical claim. The remaining elements—traceability, transition rules, risk tiers, and hypotheses—derive from it.

9.2. Human judgment as both control and risk

The human A3 gate preserves organizational decision rights, but human involvement does not automatically make a workflow safe. Reviewers may anchor on the first model-generated formulation, accept fluent text as evidence of correctness, seek evidence that confirms the recommendation, or defer to a displayed confidence score. They may also treat approval as shared responsibility when several roles have touched the proposal. These effects can convert formal oversight into nominal oversight.

The framework therefore requires review design that protects independent judgment. For consequential changes, reviewers should assess the evidence and original requirement before seeing or accepting the proposed revision, record a reason for the decision, and identify unresolved assumptions. Separation of duties is appropriate when the same person configures the prompt, selects evidence, and approves the artifact. Independent second review, disagreement escalation, and random audit should be reserved for risk classes where the expected consequence justifies their cost. Human-control quality can be measured through acceptance of seeded unsupported proposals, detection of modality or scope changes, override patterns, rationale completeness, and agreement among reviewers.

9.3. Review fatigue and the effective strength of the human gate

Review capacity is finite. A workflow that routes every ambiguity, low-confidence observation, and duplicate warning to the same approval queue can reduce attention and increase mechanical acceptance. The relevant quantity is not simply the number of human reviews but the probability that a reviewer detects a consequential defect under the actual alert burden. Low precision, repeated warnings, long evidence packages, and specialist bottlenecks can weaken that probability over a review session.

Risk-tiered routing is therefore a substantive governance control rather than an efficiency convenience. A0 observations should usually create discussion or clarification tasks rather than approval work. Duplicate findings should be consolidated, low-risk clerical changes may be sampled or batched, and high-consequence proposals should receive protected review capacity. Organizations should monitor alerts per reviewer, false-positive burden, median evidence-reading time, detection accuracy by queue position, deferral rates, and the frequency of approvals without substantive rationale. A decline in detection across a review sequence is direct evidence that the nominal human gate is stronger than the effective gate.

9.4. Governance cost and net value

The framework imposes fixed and variable costs. Fixed costs include evidence metadata, access policies, role definitions, tool integration, validation rules, and reviewer training. Variable costs include context construction, automated checks, human review, clarification, decision delay, and revalidation after source or model changes. Specialist scarcity can create an additional opportunity cost when security, compliance, or domain experts become approval bottlenecks.

Governance should therefore be evaluated at the workflow level. Total cost per committed change can be represented as the amortized setup cost plus context, verification, review, delay, and maintenance costs. Benefits include fewer unsupported commitments, fewer escaped semantic changes, improved audit reconstruction, and lower remediation exposure. The framework has positive net value only when these expected benefits exceed its total cost. This condition explains why a high-control workflow may be justified for safety, privacy, contractual, or regulatory requirements while being disproportionate for reversible clerical edits. Cost allocation also matters: a control may benefit the organization while concentrating work on a small reviewer group, thereby encouraging local resistance or superficial compliance.

9.5. Organizational implementation conditions

Implementation depends on capabilities that the framework cannot create by itself. First, evidence must have identifiable owners, versions, status, scope, and access rules. Second, approval authority must be explicit enough to determine who may perform the A2-to-A3 transition. Third, repositories and workflow tools must preserve links among evidence, context, output, verification, decision, and baseline version. Fourth, the organization needs a risk taxonomy that determines which checks, reviewers, and revalidation triggers apply to

each artifact class. Fifth, reviewers need domain competence, sufficient capacity, and incentives to reject unsupported proposals rather than maximize throughput.

These conditions suggest staged adoption. An organization should first assess evidence and role readiness, then deploy A0 tasks such as search, summarization, terminology checks, and clarification generation. A1 candidate generation should follow only when admissible evidence can be identified reliably. A2 use requires stable verification rules and audit records. High-consequence A3 changes require specialist capacity, escalation paths, and periodic review of acceptance patterns. Expanding influence faster than these capabilities mature increases procedural appearance without increasing effective control. As shown in Table 9, implementation requires evidence governance, clear decision rights, workflow integration, risk classification, reviewer capacity, and organizational learning mechanisms.

Table 9. Organizational conditions for implementation

Condition	Minimum operational evidence	Failure mode when absent
Evidence governance	Owners, versions, lifecycle status, scope, effective dates, and access classifications	Retrieval treats visible material as authoritative project fact
Decision-right clarity	Role-specific authority for artifact classes and recorded delegation	Approval becomes symbolic or responsibility is diffused
Workflow integration	Persistent links across evidence, context, output, checks, decision, and baseline	Audit reconstruction and revalidation become manual or incomplete
Risk classification	Defined artifact tiers, mandatory checks, escalation rules, and revalidation triggers	Controls are either excessive for low-risk work or inadequate for high-risk work
Reviewer capability and capacity	Domain training, workload limits, specialist availability, and calibration feedback	Bias, fatigue, delay, and rubber-stamping weaken the human gate
Learning and assurance	Error taxonomy, random audits, override analysis, and corrective-action ownership	Repeated failures remain invisible and governance does not improve over time

9.6. Boundary conditions and empirical implications

The framework is most applicable where requirements have identifiable owners, versions, and approval processes. It will be less effective in informal settings where decisions are undocumented or source status cannot be determined. Multi-organizational projects complicate authority and confidentiality; multilingual requirements complicate modality and terminology checks; rapidly changing policies increase revalidation cost; and hosted-model changes may reduce reproducibility. These are not peripheral implementation issues because they alter the relationship between the proposed change and its evidential basis.

Empirical evaluation should therefore compare sociotechnical workflows rather than isolated model outputs. Controlled studies can estimate causal effects on source use, unsupported specificity, semantic drift, and reviewer agreement. Field studies can examine cost, fatigue, role adaptation, and integration failures. Longitudinal studies can test whether evidence metadata, reviewer calibration, and downgrade rules remain effective as repositories, policies, and models change.

10. Limitations

The framework is normative and has not been validated through a controlled experiment or field deployment. The literature synthesis is representative rather than exhaustive, and the order-management scenario illustrates

construct relationships without demonstrating effectiveness. The seven hypotheses identify observable consequences, but their operational definitions and thresholds will require adaptation to domain risk, artifact type, and organizational context.

The framework cannot guarantee correct human decisions. Reviewers may misunderstand evidence, anchor on model output, approve unsupported changes, or become fatigued. Evidence metadata and automated validators may also be wrong. Moreover, the framework may impose costs that outweigh its benefits, particularly where requirements are informal, consequences are low, or specialist capacity is limited. These limitations make empirical measurement of both control performance and governance cost essential.

The A0-A3 model deliberately does not prescribe a universal risk taxonomy, approval hierarchy, or technical architecture. Its generality supports adaptation but reduces predictive precision until organizations define local artifact classes, authority rules, verification requirements, and revalidation triggers.

11. Conclusion

The paper's central claim is that an LLM output should not acquire authority over a requirements baseline merely because it is fluent, plausible, cited, or structurally valid. Linguistic quality, evidential legitimacy, verification status, and organizational authority are separate properties.

The five governance layers control evidence, context, model behavior, verification, and approval; the A0-A3 states record the influence permitted for each proposal. Together, they define a reversible path from advisory observation to approved change while preserving a human A3 decision boundary. The seven hypotheses make the model empirically contestable and require evaluation of human bias, fatigue, workflow cost, and organizational readiness alongside defect reduction. The framework should be retained only where those tests show that its reduction in unsafe artifact influence exceeds the cost of operating it.

References

[1] Borg, M. (2024). Requirements engineering and large language models: Insights from a panel. *IEEE Software, 41*(2), 6–10.

[2] Vogelsang, A. (2024). From specifications to prompts: On the future of generative large language models in requirements engineering. *IEEE Software, 41*(5), 9–13.

[3] Hou, X., Zhao, Y., Liu, Y., Yang, Z., Wang, K., Li, L., Luo, X., Lo, D., Grundy, J., & Wang, H. (2024). Large language models for software engineering: A systematic literature review. ACM Transactions on Software Engineering and Methodology, 33(8), 1–79. https: //doi.org/10.1145/3695988

[4] Ronanki, K., Cabrero-Daniel, B., & Berger, C. (2022). ChatGPT as a tool for user story quality evaluation: Trustworthy out of the box? In *International Conference on Agile Software Development* (pp. 173–181). Cham: Springer Nature Switzerland.

[5] Zhang, Z., Rayhan, M., Herda, T., Goisauf, M., & Abrahamsson, P. (2024). LLM-based agents for automating the enhancement of user story quality: An early report. In D. Šmite, E. Guerra, X. Wang, M. Marchesi, & P. Gregory (Eds.), *Agile processes in software engineering and extreme programming: XP 2024* (pp. 117–126). Springer. https: //doi.org/10.1007/978-3-031-61154-4_8

[6] Mahbub, T., Dghaym, D., Shankarnarayanan, A., Syed, T., Shapsough, S., & Zualkernan, I. (2024). Can GPT-4 aid in detecting ambiguities, inconsistencies, and incompleteness in requirements analysis? A comprehensive case study. *IEEE Access, 12*, 171972–171992. https: //doi.org/10.1109/ACCESS.2024.3464242

[7] van Can, A. T., & Dalpiaz, F. (2025). Locating requirements in backlog items: Content analysis and experiments with large language models. *Information and Software Technology, 179*, 107644.

[8] Ellsel, C., & Stark, R. (2025). Advancing requirements engineering with large language models. *Procedia CIRP, 136*, 701–706.

[9] Paiva, G., Canedo, E. D., & Filho, G. P. R. (2026). From issue titles to requirements: An empirical study of large language models and prompt engineering strategies. *Requirements Engineering, 31*(1), 7.

[10] Wolf, E., Trendowicz, A., & Siebert, J. (2026). Quality assessment of software requirements using artificial intelligence methods: A systematic literature review. *Information and Software Technology, 191*, Article 107979.

[11] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 34 (NeurIPS 2020)* (Vol. 33, Article 793, pp. 9459–9474). Curran Associates.

[12] Mavin, A., Wilkinson, P., Harwood, A., & Novak, M. (2009). Easy approach to requirements syntax (EARS). In *2009 17th IEEE International Requirements Engineering Conference* (pp. 317–322). IEEE. https: //doi.org/10.1109/RE.2009.9

[13] Ferrari, A., & Spoletini, P. (2025). Formal requirements engineering and large language models: A two-way roadmap. *Information and Software Technology, 181*, 107697.

[14] NIST. (2024). *Artificial intelligence risk management framework: Generative artificial intelligence profile: NIST AI 600-1*. Gaithersburg, MD: National Institute of Standards and Technology.

[15] ISO/IEC/IEEE 29148: 2018. (2018). *Systems and software engineering—Life cycle processes—Requirements engineering*.

[16] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems* (Vol. 30, pp. 5998–6008). Curran Associates.

[17] Cleland-Huang, J., Settimi, R., Zou, X., & Solc, P. (2007). Automated classification of non-functional requirements. *Requirements Engineering, 12*(2), 103–120. https: //doi.org/10.1007/s00766-007-0045-1

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

[19] Ferrari, A., Spagnolo, G. O., & Gnesi, S. (2017). Pure: A dataset of public requirements documents. In *2017 IEEE 25th International Requirements Engineering Conference (RE)* (pp. 502–505). IEEE.

[20] Lucassen, G., Dalpiaz, F., Van Der Werf, J. M. E. M., & Brinkkemper, S. (2016). Improving agile requirements: The Quality User Story framework and tool. *Requirements Engineering, 21*(3), 383–403. https: //doi.org/10.1007/s00766-016-0250-x

[21] Dalpiaz, F., van der Schalk, I., Brinkkemper, S., Aydemir, F. B., & Lucassen, G. (2019). Detecting terminological ambiguity in user stories: Tool and experimentation. *Information and Software Technology, 110*, 3–16. https: //doi.org/10.1016/j.infsof.2018.12.007

[22] Jaakkola, E. (2020). Designing conceptual articles: Four approaches. *AMS Review, 10*(1), 18–26.

[23] Jabareen, Y. (2009). Building a conceptual framework: Philosophy, definitions, and procedure. *International Journal of Qualitative Methods, 8*(4), 49–62.