

ReqGenX: An Empirical Study of Atomic Decomposition, Artifact Regeneration, and Reconstruction for Legacy SRS Documents

Ragib Shahariar Ayon ✉️🏠^{id}

Texas State University, San Marcos, TX, USA

Rayed Fahmi ✉️^{id}

Texas State University, San Marcos, TX, USA

Sumon Biswas ✉️🏠^{id}

Case Western Reserve University, Cleveland, OH, USA

Shibbir Ahmed ✉️🏠^{id}

Texas State University, San Marcos, TX, USA

Abstract

Background. Evaluating automated Software Requirements Specification (SRS) generation is challenging because few datasets provide fine-grained traceability between source requirements, intermediate elicitation artifacts, and generated specifications.

Aims. We study whether legacy SRS documents can serve as controlled reference evidence for constructing traceable synthetic pre-SRS artifacts that enable fine-grained evaluation of LLM-based requirements transformations.

Method. We conduct an empirical study using *ReqGenX*, a controlled pipeline that decomposes SRS sections into atomic statements grounded in the SRS content, routes atoms to standards-inspired artifact types through multi-LLM plurality voting, and generates artifacts using constrained prompts with iterative judge-guided refinement. We evaluate *ReqGenX* on seven PURE SRS documents using grounding, quality, information retention, and downstream reconstruction analyses.

Results. *ReqGenX* produces highly grounded, well-formed atoms that score strongly on our atomicity and downstream-utility criteria, with median AlignScore values typically between 0.96 and 0.99 and Prometheus scores ranging from 4.34 to 4.85. Generated artifacts remain strongly grounded in their source atoms, with AlignScore values typically between 0.80 and 0.94 and judge pass rates near 100%; stricter Prometheus evaluation yields pass rates ranging from 54.8% to 97.1%. In a downstream SRS reconstruction case study, artifact-backed atoms remain recoverable from generated SRSs, with SBERT means between 0.69 and 0.75 and AlignScore medians between 0.76 and 0.84.

Conclusions. Traceable synthetic pre-SRS artifacts can support fine-grained evaluation of LLM-based SRS generation, while exposing tradeoffs among faithfulness, information retention, and artifact completeness. These artifacts are intended as controlled evaluation resources rather than substitutes for primary elicitation evidence.

2012 ACM Subject Classification Software and its engineering → Requirements analysis

Keywords and phrases Requirements, large language model, software engineering

Digital Object Identifier 10.4230/LIPIcs.ESEM.2026.35

Category Technical Track Paper

Related Version *Previous Version:* <https://arxiv.org/abs/2607.16564>

Supplementary Material *Software:* <https://zenodo.org/records/20267334>



© Ragib Shahariar Ayon, Rayed Fahmi, Sumon Biswas, and Shibbir Ahmed;
licensed under Creative Commons License CC-BY 4.0

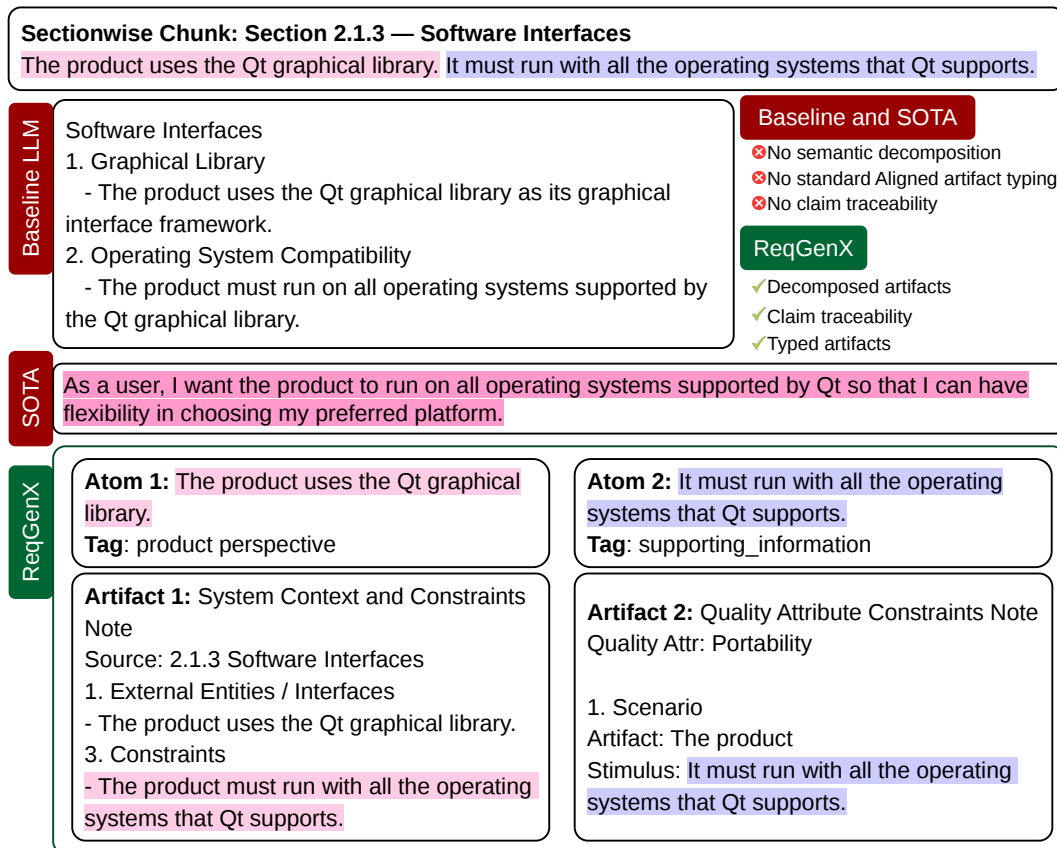
20th International Symposium on Empirical Software Engineering and Measurement (ESEM 2026).

Editors: Robert Feldt, Maria Paasivaara, Daniel Mendez, Stefan Wagner, and Marvin Muñoz Barón;
Article No. 35; pp. 35:1–35:20



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



■ **Figure 1** Example of section-to-atom decomposition and artifact routing. The illustrative direct-generation branch merges claims, whereas *ReqGenX* decomposes and routes them to typed, traceable artifacts. This branch illustrates the traceability problem and is not an experimentally evaluated baseline or SOTA comparison. Colored highlights (■: requirement content, ■: supporting information content) map source statements to artifact content.

1 Introduction

Large language models (LLMs) are increasingly used in requirements engineering (RE) for SRS generation, requirement extraction and validation, and intermediate artifacts such as user stories and acceptance criteria [25, 40, 34]. Related work has also sought to make software behavior and assumptions explicit through behavioral contracts and inferred data preconditions [5, 4], while recent LLM-based approaches extend this direction to formal and executable specification generation, intent-aware development workflows, and structured representations of functional requirements derived from developer discussions [7, 9]. At the SRS level, however, an LLM-generated document may omit source information, merge distinct concerns, or introduce unsupported requirements. Evaluating LLM-based SRS compilation, therefore, requires more than document-level quality: it requires evidence about which source claims support generated content and where information is preserved or lost. Prior NLP4RE work addresses requirements classification, extraction, quality analysis, and traceability [39]; PURE provides public requirements documents [18], and traceability research links information across lifecycle artifacts [19]. However, existing public resources do not jointly provide source requirements, intermediate RE artifacts, generated specifications,

and their provenance links. This limits researchers’ ability to localize failures across multi-stage LLM-supported requirements transformations. We therefore study where information is preserved, transformed, or lost when legacy SRSs are used as controlled source evidence.

In this study, *pre-SRS artifacts* are typed intermediate representations of requirements that can precede or support SRS authoring, including use-case briefs, quality-attribute constraints, context notes, acceptance criteria, terminology/reference artifacts, and assumption/dependency records. *ReqGenX* generates synthetic proxy artifacts from legacy SRS content; it does not recover the historical elicitation materials behind the original SRSs or address greenfield elicitation from interviews, stakeholder discussions, issue reports, regulations, or other primary evidence. Our primary audience is researchers evaluating LLM-supported RE transformations.

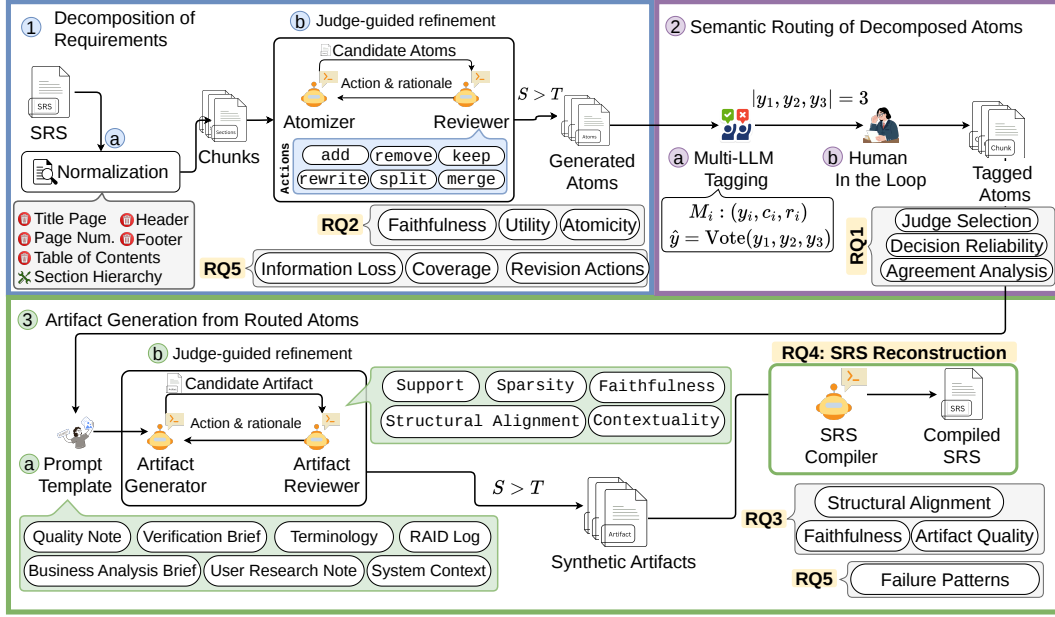
Figure 1 illustrates the traceability problem using a *QHeadache* section from PURE [18]. The section contains a Qt library dependency and an operating system compatibility constraint. An illustrative direct-generation branch merges these claims into one generic artifact, whereas *ReqGenX* decomposes them into separate atoms and routes them to typed artifacts while preserving claim-level provenance. To study this problem, *ReqGenX* uses each SRS as controlled evidence and operates in three stages: 1 normalization and decomposition into atomic statements grounded in the observed SRS chunk; 2 semantic routing through multi-LLM plurality voting with human review; and 3 artifact-specific generation with iterative quality checks. Stable identifiers preserve provenance from SRS sections through atoms, semantic tags, and generated artifacts. We evaluate grounding, atomicity, downstream utility, information retention, decision reliability, artifact quality, and downstream reconstruction across seven PURE SRSs.

Our results show that atoms are strongly grounded, with median AlignScore values typically between 0.96 and 0.99, although source coverage varies substantially across datasets and models. Generated artifacts also remain strongly grounded in their source atoms, with AlignScore values typically between 0.80 and 0.94, while stricter Prometheus evaluation reveals wider variation in artifact quality. In the downstream reconstruction case study, artifact-backed atoms remain partially recoverable, with SBERT means of 0.690–0.753 and AlignScore medians of 0.760–0.842. These results show that strong grounding at one stage of transformation does not imply complete information preservation downstream. We do not position *ReqGenX* as a production SRS-generation system; rather, we use it as a traceable experimental setting for studying information preservation, evidence granularity, artifact quality, and LLM-based intermediate decisions. This setting exposes intermediate failure modes that final-document evaluation alone cannot localize.

Our contributions are:

- We characterize LLM-based atomization across seven PURE SRSs, showing that strong atom-level faithfulness can coexist with substantial variation in source coverage.
- We evaluate typed artifact generation and show that strong source grounding does not necessarily imply artifact completeness or structural quality.
- We analyze stage-wise information loss across SRS→Atoms→Artifacts and evaluate downstream reconstruction separately as a controlled recoverability case study.
- We evaluate semantic routing and LLM-based quality judgment against human-reviewed labels, including a human-calibrated judge-selection procedure.
- We release a traceable replication package containing processed SRSs, generated atoms and artifacts, prompts, codebooks, provenance metadata, and evaluation outputs [8].

Sections 2–4 present the pipeline, experimental design, and results; Sections 5–8 discuss lessons learned, related work, threats to validity, and conclusions.



■ **Figure 2** End-to-end *ReqGenX* study pipeline for deriving and evaluating traceable synthetic pre-SRS artifacts from legacy SRS documents.

2 Study Pipeline

Terminology. We use *section* for a logical SRS unit after normalization, *chunk* for the text associated with that section, and *atom* for a minimal proposition extracted from and supported by a chunk. Throughout this paper, *source* refers to the observed legacy SRS content used as evidence, not to the unknown historical elicitation materials from which the SRS may have been created. An atom should express one coherent statement and remain interpretable in isolation, consistent with the requirements-quality notion of atomicity [20].

ReqGenX is intentionally a multi-stage experimental pipeline rather than a lightweight or deployment-ready RE tool. We assess its stages through manual normalization checks, human-reviewed routing labels, candidate-model screening, reconciled human annotations for judge selection, grounding metrics, and external Prometheus evaluation. Figure 2 summarizes three stages: 1 decomposition into atoms grounded in the observed SRS content; 2 semantic routing of atoms to artifact types; and 3 generation and refinement of typed pre-SRS artifacts.

2.1 Decomposition of Requirements

The pipeline begins with an SRS document, typically provided as a PDF. We convert each document into a text representation that preserves headings, lists, and basic structure, then normalize it by removing presentation-oriented material and repairing common PDF-conversion errors [10, 28]. The normalized text is manually checked against the original SRS to ensure that requirements content is preserved and parsing artifacts are not introduced. Although PURE also provides parsed XML representations, our implemented pipeline operates on the original SRS documents and includes PDF-to-text normalization as preprocessing; using PURE’s XML could reduce this effort in future replications.

■ **Table 1** Routing of atom tags to pre-SRS artifact types.

IEEE 29148 Artifact	Atom Tags
Business Analysis Brief	Purpose, Scope, Product Functions
User Research	User Characteristics
System Context and Constraints	Product Perspective, Limitations, Supporting Information, Other
Use Case Brief	Functional
Quality Attribute Constraints	Availability, Fault Tolerance, Operability, Performance, Portability, Scalability, Security, Legal, Usability, Look and Feel, Maintainability
Acceptance Criteria Sheet	Verification
Terminology and References	Definitions, References, Acronyms and Abbreviations
RAID Log Entry	Assumptions and Dependencies

The normalized SRS is segmented into section-wise chunks that retain section and source-document metadata. An LLM-based atomizer then separates independent propositions while preserving qualifiers, conditions, bounds, and scope. In the running example of Figure 1, the software-interface section is decomposed into separate atoms for the Qt dependency and operating-system compatibility.

Each atom is evaluated for three criteria: *faithfulness*, requiring every claim to be supported by the source chunk; *atomicity*, requiring one self-contained proposition rather than bundled ideas or structural fragments; and *downstream utility*, which we operationalize as whether the atom contains sufficient requirements-relevant information to instantiate at least one artifact type in Table 1. Because LLM-generated atoms may still be ambiguous, infeasible, or unsupported [33], a separate judge scores these criteria and returns **keep**, **rewrite**, **split**, **merge**, **remove**, or **add** actions. Failed atom sets are revised before routing.

2.2 Semantic Routing of Decomposed Atoms

After decomposition, *ReqGenX* assigns each atom an IEEE 29148-inspired semantic tag [2, 39, 21]. Table 1 maps these tags to pre-SRS artifact families. In the running example, the Qt-library atom is routed to a System Context and Constraints artifact, whereas the operating-system compatibility atom supports a portability-related Quality Attribute Constraint. Routing follows an independent proposal-then-aggregate strategy. The atomizer proposes an initial tag, three LLM agents independently label the atom without observing peer outputs, and plurality voting selects the final tag [35, 41]. Unresolved cases receive human-in-the-loop (HITL) review before artifact generation. Atoms tagged as **requirements** undergo a second classification as *functional*, *quality*, or *other*. Functional atoms are routed to Use Case Briefs, whereas quality atoms are routed to Quality Attribute Constraints and further assigned a PROMISE-NFR subtype, such as availability, performance, portability, security, or usability [27]. These subtypes guide artifact expression rather than changing the artifact family. The same plurality-voting and human-review process is used for this second classification.

2.3 Artifact Generation and Review

After routing, *ReqGenX* transforms each validated atom into an artifact using a type-specific template. In the running example, the two routed atoms are expressed using their corresponding artifact structures while preserving their source identifiers. The `atom_text` is

the primary source of truth: substantive artifact content must be supported by the atom, while semantic tags, section metadata, and chunk context are used only for structure or minor disambiguation. Generated artifacts retain an identifier linking them to their source atoms, and post-processing removes residual formatting or rationale text. Templates are deliberately conservative: they prohibit unsupported enrichment, omit unsupported fields, and permit empty outputs when the available evidence cannot support a valid artifact. A rubric-based judge evaluates artifacts for *faithfulness*, *support*, *contextuality*, *structural alignment*, and *sparsity*. Failed artifacts are revised using judge feedback until they satisfy the acceptance threshold or the refinement budget is exhausted. The eight artifact families in Table 1 cover the principal information roles considered in this study and draw on established requirements and architecture documentation practices [13, 2, 11, 1].

2.4 Traceability and Provenance Model

ReqGenX maintains provenance across transformations so that generated outputs can be audited against their SRS evidence. Chunks retain document and section metadata, while each generated artifact receives a stable trace identifier linked to its source atom, semantic label, and relevant source context. The resulting chain connects the SRS document and section to the chunk, atom, semantic tag, generated artifact, and, for the reconstruction case study, matched SRS sentence windows. In the running example, this chain links the original QHeadache section to each atom, semantic label, and generated artifact. We use these links for grounding checks, information retention analysis, and downstream recoverability evaluation. This metadata is not intended as a complete industrial traceability solution; it provides the provenance required for the controlled analyses in this study.

3 Experimental Design

We evaluate *ReqGenX* as a controlled empirical pipeline for studying information change across transformations from legacy SRS content to synthetic pre-SRS artifacts and downstream reconstruction. We examine five complementary aspects: 1 the reliability of routing and quality decisions that determine how atoms proceed through the pipeline; 2 atomization quality and information preservation; 3 artifact grounding and structural quality; 4 downstream recoverability in an SRS reconstruction case study; and 5 stage-wise information loss and associated failure patterns. Together, these dimensions expose behavior that final-document evaluation alone cannot localize. In RQ1, *RE-specific intermediate decisions* refer to semantic routing, which determines which artifact family an atom can support, and quality judgment, which determines whether generated outputs proceed downstream.

- **RQ1:** How reliable are the RE-specific intermediate decisions made by LLMs within *ReqGenX* for semantic routing and quality judgment, and how does reliability vary across models and decision types?
- **RQ2:** How does atomization quality vary across LLMs and SRS documents in terms of faithfulness, atomicity, downstream utility, and source-information coverage?
- **RQ3:** How does the quality of generated pre-SRS artifacts vary across LLMs and SRS documents in terms of grounding, sparsity, structural alignment, and rubric-based quality?
- **RQ4:** To what extent are artifact-backed source atoms recoverable after downstream SRS reconstruction in a controlled case study?
- **RQ5:** At which transformation stages is source information lost, and which revision and failure patterns explain the observed losses?

RQ1 uses PROMISE-NFR candidate-model screening, Cohen’s κ agreement with human-reviewed routing labels, and human-calibrated judge selection using F1, false-positive rate (FPR), and false-negative rate (FNR). RQ2 compares seven SRSs and three atomizer models using AlignScore, atomicity, downstream utility, Prometheus, and source coverage. RQ3 evaluates artifacts using source-atom AlignScore, internal-judge acceptance, Prometheus, and artifact-specific quality criteria. RQ4 uses a four-dataset reconstruction case study with Claude-generated artifacts, GPT-4o compilation, and SBERT/AlignScore recoverability. RQ5 combines stage-wise retention, revision-action frequencies, and artifact-generation failure traces.

3.1 Dataset

We construct the benchmark from PURE [18] following the document-selection protocol of Okamoto et al. [28]. Because *ReqGenX* operates on ordered text, we retain documents explicitly identified as SRSs whose requirements can be represented as textual sections, paragraphs, and lists. We exclude other specification types and SRSs containing substantive figures or non-revision-history tables that lack a stable linear reading order; revision-history tables are retained because they can be normalized without changing their meaning. The resulting benchmark contains seven SRSs: Get Real, Library, Puget Sound, QHeadache, VUB, Libra, and Home, comprising 31,454 words, 1,408 sentences, and 369 sections. This selection favors text-oriented SRSs and may underrepresent failures arising in figure-heavy, table-heavy, or multimodal specifications.

3.2 Implementation and Large Language Models

We implemented *ReqGenX* in Python 3.11 with LLM interactions orchestrated through LangChain. Experiments ran on a 28-core Intel® Xeon® W-3465X CPU (2.50 GHz), 256 GB RAM, and an NVIDIA RTX 6000 Ada Generation GPU under Ubuntu 24.04.2 LTS. We use a temperature of 1.0 to encourage output diversity. The evaluated model set spans closed- and open-source families and includes Claude Sonnet 4.6, GPT-4o, GPT-OSS (20.9B), Gemma 3 (27.4B), Phi-4 (14.7B), Mistral (7.2B), Llama 3.1 (8.0B), Qwen3 (30.5B and 8.2B), and LLaMA 3 (8.0B), used across atom generation, semantic routing, artifact generation, and review.

3.3 Evaluation Procedure

3.3.1 Judge Calibration and Selection

For quality judgment, we sample 291 triples from 1,195 Claude Sonnet 4.6-generated artifact instances. Two annotators independently score atom and artifact quality using the shared 1–5 rubric, with disagreements adjudicated into reconciled reference labels. Candidate judges are compared against these labels using F1, FPR, and FNR. Because false acceptance can propagate low-quality outputs downstream, judge selection prioritizes low FPR.

3.3.2 Reconstruction Case-Study Procedure

RQ4 is evaluated on four datasets using an additional pipeline for generation, quality assurance, and traceability matching. We use Claude Sonnet 4.6-generated artifacts for their strong upstream performance, and GPT-4o as a separate compiler model. GPT-4o compiles the artifacts into SRS drafts under ISO/IEC/IEEE 29148-inspired constraints while

preserving artifact-level traceability, then refines them for ambiguity, singularity, verifiability, and syntactic correctness. Each reconstructed SRS is then matched to its accepted artifact-backed source atoms using the reconstruction metrics defined below.

3.4 Evaluation Metrics

We use grounding and rubric-based quality measures as the primary indicators of output quality, while coverage, agreement, revision actions, and reconstruction metrics serve as complementary diagnostics for localizing information loss, model disagreement, and downstream recoverability.

AlignScore. We use AlignScore [37] to measure semantic support. For atoms, the source chunk is the context and the atom is the claim; for artifacts, the source atom is the context and the artifact is the claim. We use 0.80 as the support threshold for unsupported-rate analyses.

Judge LLM and Prometheus Evaluation. A rubric-based judge scores atom and artifact quality on a 1–5 scale, with scores ≥ 4 treated as passing. Atoms are assessed for faithfulness, atomicity, and downstream utility; artifacts for faithfulness, support, contextual use, structural alignment, and sparsity. Prometheus [24] provides an external complementary rubric-based assessment.

Embedding-Based Information Retention. We estimate source retention using cosine similarity over `text-embedding-3-large`. Source SRS text is divided into overlapping two-sentence units and matched to final atoms; a unit is covered when its best-match similarity is at least 0.75. We use this measure for atomization coverage and stage-wise SRS→Atoms→Artifacts retention.

Downstream SRS Reconstruction Metrics. We measure the recoverability of accepted artifact-backed atoms from reconstructed SRSs. Each SRS is segmented into overlapping sentence windows, and source atoms are matched to their closest windows using SBERT cosine similarity [31]. AlignScore is then applied with the matched SRS window as context and the source atom as the claim. SBERT captures semantic similarity, whereas AlignScore estimates factual support.

4 Results and Evaluation

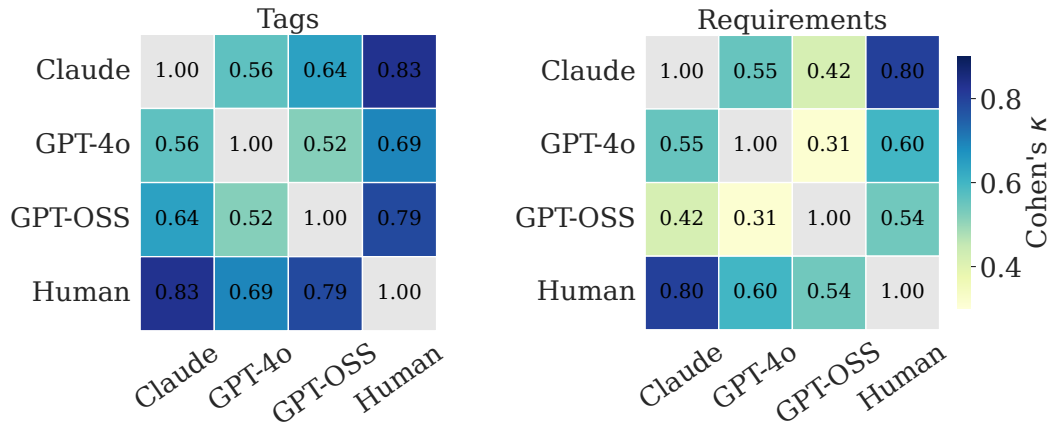
4.1 RQ1: Reliability of LLM-Based RE Decisions

Finding 1: Reliability of LLM-based RE-specific intermediate decisions varies across models and decision types. Semantic tagging is more stable than fine-grained requirement classification, while judge performance also differs across models. These results indicate that intermediate LLM decisions in RE pipelines should be validated independently rather than treated as uniformly reliable, particularly when their errors can propagate to downstream artifacts.

■ **Table 2** Performance of candidate LLMs on the PROMISE dataset for plurality voting selection. Metrics are reported as support-weighted per-class precision, recall, and F1; weighted F1 is averaged from per-class F1 values.

Model	Precision	Recall	F1
Claude Sonnet 4.6	0.80	<u>0.84</u>	0.81
GPT-4o	0.78	0.86	0.81
GPT-OSS	<u>0.84</u>	0.68	<u>0.70</u>
Phi-4	<u>0.84</u>	0.55	0.59
Qwen3 (30B)	0.92	0.43	0.50
Qwen3 (8B)	0.80	0.46	0.50
LLaMA 3 (8B)	0.44	0.37	0.36

Note: Bold values indicate the best value in each metric column; underlined values indicate the second-best value. Ties are marked for all tied values.



■ **Figure 3** Pairwise Cohen's κ agreement among LLM-generated and human-reviewed labels for semantic tag assignment and requirement classification. Diagonal cells denote self-agreement, while off-diagonal cells show agreement between annotator/model pairs.

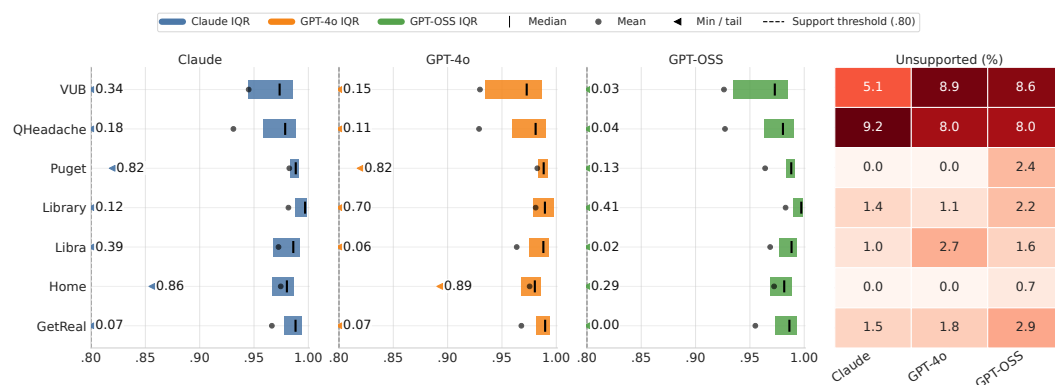
RQ1 shows that reliability varies across models and decision types. On the re-labeled PROMISE-NFR dataset¹, Claude Sonnet 4.6 and GPT-4o achieve the strongest weighted F1 among closed-source models, while GPT-OSS is the strongest open-source model. Because PROMISE-NFR does not cover the full *ReqGenX* routing schema, Table 2 should be interpreted as evidence of general requirement-classification ability rather than direct routing accuracy. Based on this screening, we use Claude Sonnet 4.6, GPT-4o, and GPT-OSS 20B as complementary voters for plurality-based routing. On the actual *ReqGenX* routing schema, agreement with human-reviewed labels is consistently higher for semantic tags than for fine-grained requirement classification. The analysis includes 2,054 semantic-tag instances and 1,177 requirement-label instances; human review corrected 353 semantic tags (17.2%) and 58 requirement labels (4.9%). Figure 3 shows that Claude has the strongest agreement with human-reviewed labels, reaching $\kappa = 0.83$ for tags and $\kappa = 0.80$ for requirements, while model-to-model agreement is weakest for requirement classification, where GPT-4o

¹ <https://zenodo.org/records/14330485>

■ **Table 3** Comparison of candidate judge LLMs for atom- and artifact-level assessment against reconciled human labels. F1 is higher-is-better. FPR and FNR are lower-is-better.

LLM	Atoms			Artifacts		
	F1 ↑	FPR ↓	FNR ↓	F1 ↑	FPR ↓	FNR ↓
Gemma 3	0.80	0.33	0.20	0.88	0.29	0.13
GPT-4o	0.77	<u>0.28</u>	0.27	0.88	0.63	<u>0.03</u>
Phi-4	0.79	0.46	<u>0.16</u>	<u>0.87</u>	<u>0.47</u>	0.09
Mistral	0.74	0.89	0.12	0.84	1.00	0.01
Llama 3.1	0.66	0.23	0.44	0.83	0.90	0.04
GPT-OSS	0.81	0.49	0.12	0.76	0.77	0.21

Note: Bold values indicate the best value in each metric column; underlined values indicate the second-best value. Ties are marked for all tied values.



■ **Figure 4** Chunk-level atom faithfulness across datasets and atomizer models. Most atomized outputs achieve high central AlignScore values, with medians generally above 0.95, but several dataset-model pairs exhibit low-scoring tail cases. Unsupported rates are computed using the AlignScore support threshold of 0.80 and show that high typical faithfulness can still coexist with occasional unsupported atoms.

and GPT-OSS fall to $\kappa = 0.31$. For quality judgment, the 291 human-annotated triples yield strong inter-annotator agreement, with Cohen’s $\kappa = 0.84$ for atoms and $\kappa = 0.97$ for artifacts. Table 3 shows that Gemma 3 provides the best overall tradeoff under the judge-selection criteria defined in Section 3, with F1 scores of 0.80 for atoms and 0.88 for artifacts while avoiding the substantially higher false-positive rates of GPT-OSS for atoms and GPT-4o for artifacts. We therefore use Gemma 3 as the automated judge for large-scale evaluation.

4.2 RQ2: Quality of Generated Atoms

RQ2 evaluates whether LLMs can decompose legacy SRS sections into atoms that are faithful, atomic, and useful for downstream artifact generation. Overall, the results show that LLM-generated atoms are usually well grounded, but atomization is not equivalent to lossless extraction: usable decomposition requires balancing faithfulness, source coverage, atomicity, and downstream utility.

Figure 4 shows that generated atoms are generally faithful to their source chunks. Across datasets and models, median AlignScore values typically range from 0.96 to 0.99, while unsupported rates remain below 3% for most dataset-model pairs. The strongest cases

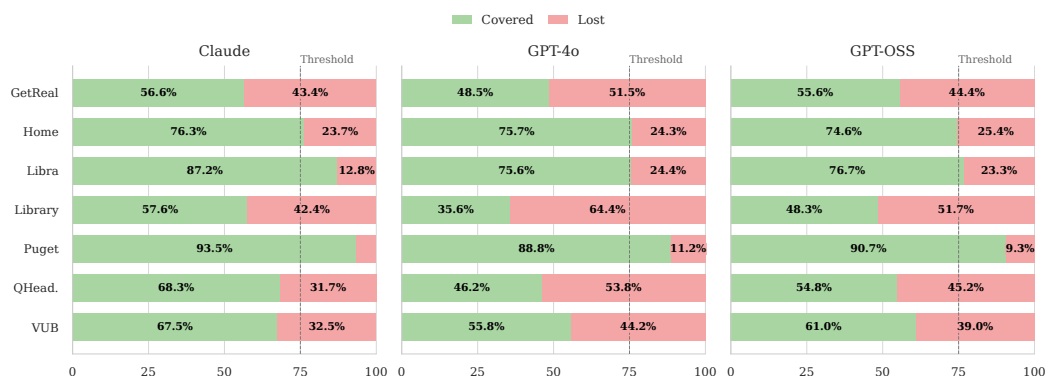


Figure 5 Information coverage and loss across datasets using overlapping window-2 source units with a similarity threshold of 0.75. Each horizontal stacked bar shows the percentage of covered and lost source information units for a given model and dataset.

Table 4 Final atom quality across SRSs and atomizer models. Atomicity and usability pass rates report the percentage of final atoms receiving judge scores ≥ 4 ; average Judge (Gemma 3) and Prometheus scores are reported on a 1–5 scale.

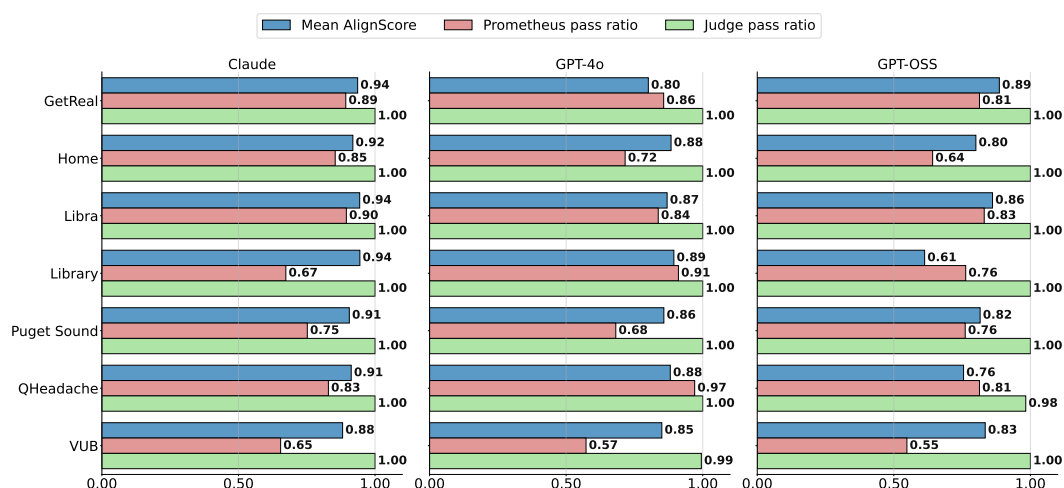
SRS	Atomicity Pass %			Usability Pass %			Avg. Judge Score			Avg. Prometheus Score		
	Claude	GPT-4o	GPT-OSS	Claude	GPT-4o	GPT-OSS	Claude	GPT-4o	GPT-OSS	Claude	GPT-4o	GPT-OSS
VUB	98.1	92.5	<u>96.4</u>	96.2	85.6	<u>89.8</u>	4.84	4.70	<u>4.75</u>	4.72	<u>4.69</u>	4.67
QHeadache	98.0	76.0	<u>87.5</u>	93.9	70.7	<u>75.0</u>	4.84	4.65	<u>4.66</u>	<u>4.77</u>	<u>4.77</u>	4.80
Puget Sound	<u>96.2</u>	96.3	94.5	82.6	<u>84.6</u>	87.4	4.67	4.69	4.69	4.34	4.44	<u>4.37</u>
Library	90.0	85.6	<u>87.6</u>	87.9	<u>82.2</u>	80.3	4.87	<u>4.76</u>	4.75	<u>4.81</u>	4.77	4.85
Libra	93.0	<u>94.1</u>	95.3	<u>91.0</u>	89.8	92.2	4.78	4.80	4.80	4.72	4.75	4.72
Home	98.5	100	<u>99.3</u>	97.0	90.3	<u>94.6</u>	4.90	<u>4.86</u>	4.85	<u>4.70</u>	4.69	4.72
Get Real	95.4	93.0	95.7	81.5	<u>82.5</u>	85.7	<u>4.64</u>	4.62	4.73	4.46	4.60	<u>4.59</u>

Note. **Bold** values indicate the best-performing model for each dataset and metric group; underlined values indicate the second-best model. Ties are marked for all tied models.

occur in Puget Sound and Home, where unsupported rates are 0.0% for Claude Sonnet 4.6 and GPT-4o, and remain low for GPT-OSS. The highest unsupported rates appear for QHeadache with Claude-generated atoms (9.2%) and for VUB with GPT-4o and GPT-OSS atoms (8.9% and 8.6%, respectively). These low-scoring tails show that median faithfulness alone is insufficient: occasional unsupported atoms can still appear and may propagate into routing and artifact generation. Model-level differences are small and should be interpreted cautiously. A paired bootstrap over the seven SRSs shows that Claude and GPT-4o are not clearly separated on mean AlignScore, while Claude is only slightly higher than GPT-OSS; for unsupported rate, Claude is lower than GPT-OSS by 1.13 percentage points, but not clearly separated from GPT-4o. We therefore treat model rankings as descriptive rather than conclusive.

Finding 2: LLM atomization is generally faithful in the information it retains, but high faithfulness does not imply high source coverage. This distinction matters for RE evaluation pipelines: grounding and information retention should be assessed separately, since an atom can be well supported by its source while still omitting relevant source content.

Figure 5 shows a sharper tradeoff for source coverage. Puget Sound is consistently best preserved, with 88.8–93.5% coverage, followed by Libra with 75.6–87.2%. In contrast, GPT-4o covers only 35.6% of Library and 46.2% of QHeadache. This suggests that atomization is



■ **Figure 6** Artifact quality over generated artifacts. Mean AlignScore, Prometheus pass ratio, and judge pass ratio are reported on a 0–1 scale. All models show high atom-level grounding and high judge acceptance, while Prometheus reveals stricter variation in artifact completeness and quality.

often faithful in what it retains, but may omit or transform source information depending on the dataset and model. Some loss is expected: judge-guided revision can remove redundant, structural, vague, or low-utility content to produce cleaner atoms for downstream use. Thus, coverage should be interpreted alongside faithfulness rather than as a standalone failure signal. Table 4 shows that final atoms are generally well-formed after revision. Atomicity pass rates range from 76.0% to 100.0%, with most dataset-model pairs above 90%, while usability pass rates range from 70.7% to 97.0%. Average Gemma 3 judge scores remain high, ranging from 4.62 to 4.90, and Prometheus scores range from 4.34 to 4.85. These results indicate that LLMs can produce strong atomic decompositions, but downstream utility is harder to achieve than atomicity and benefits from explicit judge-guided refinement.

4.3 RQ3: Quality of the Generated Artifacts

Finding 3: Generated pre-SRS artifacts can remain strongly grounded in their source atoms while still varying substantially in completeness, sparsity, and structural alignment. For RE evaluation pipelines, grounding alone is therefore insufficient; artifact-specific quality checks are necessary to assess whether generated artifacts are structurally adequate for downstream use.

Across datasets and models, generated artifacts remain strongly grounded in their source atoms, but stricter quality assessment reveals substantially greater variation than the internal judge. Figure 6 summarizes these results. Mean AlignScore values typically range between 0.80 and 0.94, indicating that generated artifacts are largely supported by their source atoms. Claude Sonnet 4.6 shows the strongest and most stable grounding performance, reaching 0.94 on Library, while GPT-OSS exhibits larger variation, dropping to 0.61 on the same dataset. Judge pass ratios are nearly saturated after iterative refinement, remaining close to 1.00 across almost all dataset-model pairs. This suggests that the review-and-repair loop reliably produces artifacts that satisfy the pipeline’s acceptance criteria. However, Prometheus reveals substantially greater variation, with pass ratios ranging from 0.55 to 0.97. The strongest

■ **Table 5** Downstream reconstruction retention scores. SBERT measures semantic similarity between source atoms and generated SRS, while AlignScore estimates factual support.

Dataset	SBERT			AlignScore		
	Mean	Median	IQR	Mean	Median	IQR
PugetSound	0.697	0.717	[0.622–0.782]	0.596	0.785	[0.081–0.988]
Library	0.698	0.720	[0.640–0.764]	0.643	0.842	[0.307–0.962]
Libra	0.690	0.694	[0.634–0.755]	0.551	0.785	[0.040–0.974]
Home	0.753	0.770	[0.706–0.821]	0.633	0.760	[0.261–0.984]

Prometheus result appears on QHeadache using GPT-4o (0.97), while lower scores occur for VUB and Home, particularly for GPT-OSS and GPT-4o. This gap indicates that atom-level grounding alone does not guarantee artifact completeness, sparsity, or structural quality. Taken together, these results show that generated artifacts can remain strongly grounded in their source atoms while still varying substantially under stricter artifact-quality assessment.

4.4 RQ4: Downstream SRS Reconstruction

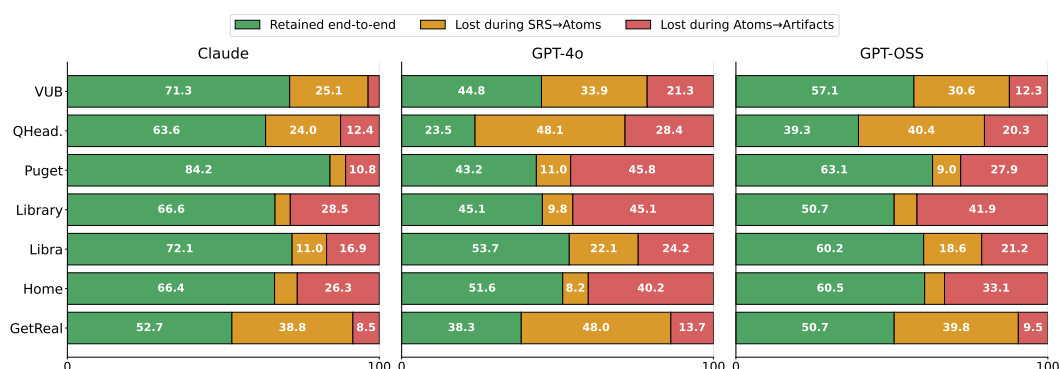
RQ4 evaluates downstream recoverability as a controlled four-dataset case study rather than practitioner-ready SRS generation. Table 5 reports retention across 473 accepted artifact-backed atoms. SBERT means range from 0.690 to 0.753, indicating moderate-to-high semantic recoverability. AlignScore medians range from 0.760 to 0.842, while lower means and wider IQRs show that factual support is more variable than semantic similarity. These low-support cases may reflect abstraction, rephrasing, or additional content introduced during SRS compilation rather than direct omission alone.

Finding 4: LLM-generated pre-SRS artifacts can support partial downstream SRS reconstruction, but semantic similarity does not guarantee factual recoverability. For RE evaluation, reconstructed SRSs should therefore be treated as traceable drafts whose retained source support must be checked explicitly, rather than as complete or practitioner-ready specifications.

4.5 RQ5: Information Loss and Failure Patterns

Information loss is distributed across both SRS→Atoms and Atoms→Artifacts transformations rather than concentrated in a single step. Claude retains the most source information end-to-end, preserving 52.7–84.2% of source content, while GPT-4o shows the largest cumulative loss, dropping to 23.5% retention on QHeadache; GPT-OSS falls between these ranges, retaining 39.3–63.1%. Loss also varies by dataset: QHeadache and Get Real lose substantial information during SRS→Atoms conversion, suggesting that implicit or weakly structured statements are difficult to stabilize as atoms, whereas Library and Puget Sound lose more information during Atoms→Artifacts conversion, indicating that valid atoms may still be too narrow to instantiate constrained artifact templates.

Revision and artifact diagnostics help explain these patterns. Most atom-level revisions involve granularity, context, or downstream usefulness rather than unsupported hallucination. The most frequent buckets are bundled or non-atomic claims (30.4%), context-dependent wording (20.4%), and weak or non-actionable wording (8.5%) [8]. Rewrite and remove actions



■ **Figure 7** Stage-wise information retention across the *ReqGenX* pipeline. Green indicates information retained end-to-end, while yellow and red indicate estimated information loss during atomization and artifact generation, respectively.

dominate, while split actions remain visible for dense SRS sections; merge and add actions are rare. This suggests that the atomizer more often over-produces weak or underspecified atoms than under-produces missing ones.

Finding 5: Information loss is stage-specific and often reflects evidence granularity rather than generation failure. Decomposition can discard implicit, redundant, or weakly actionable source content, while artifact generation can fail when a valid atom is too narrow to instantiate a complete typed artifact. For RE evaluation pipelines, information retention should therefore be assessed at multiple transformation stages rather than only at the final output.

Artifact-level failures show a related evidence-granularity problem: missing or incomplete artifacts are rarely due to parsing or generation crashes, but usually occur because the selected atom lacks enough evidence to fill the target artifact structure. For Claude, sparse artifacts are often associated with *chunk-only information*, while for GPT-4o and GPT-OSS the dominant issue is *template fields omitted*.

5 Lessons Learned

① *LLM-based RE decisions need uncertainty-aware evaluation.* RQ1 shows that LLMs can support semantic routing and quality judgment, but reliability varies across decision types. Because routing determines artifact assignment and judging controls downstream acceptance, disagreement can reveal ambiguous requirements, weak artifact boundaries, and cases needing human review. LLM4RE researchers should therefore evaluate routing confidence and disagreement patterns, and assess false-acceptance risk, while practitioners should review low-confidence and automatically accepted outputs.

② *Not every SRS statement is immediately artifact-ready.* RQ2 shows that LLM-generated atoms can be faithful and well-formed, while source coverage still varies across datasets and models. Legacy SRSs mix concrete requirements with context, vague notes, structural fragments, and partially specified ideas. Some statements can directly seed pre-SRS artifacts, while others require additional context or aggregation. Researchers should therefore study artifact readiness, while practitioners should avoid forcing every SRS statement into an artifact template.

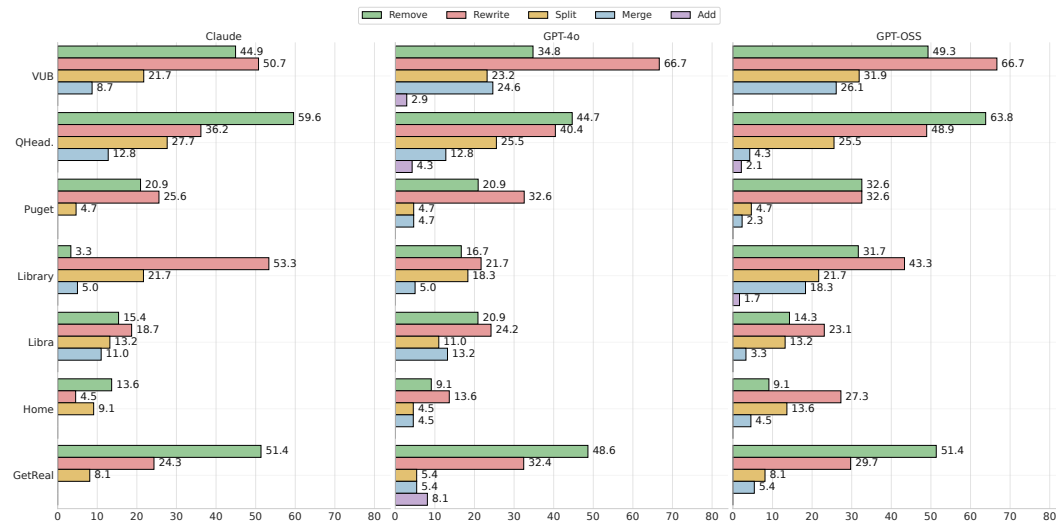


Figure 8 Revision action rates across datasets and atomizer models. Rewrite and remove actions dominate across most dataset-model pairs, while split actions remain visible for structurally dense SRS sections.

3 *Artifact generation requires balancing completeness and evidence control.* RQ3 shows that artifacts can remain grounded in their source atoms, but grounding alone does not guarantee completeness or usefulness. Adding synthetic details may improve completeness but risks unsupported content, while strict atom-grounding can produce sparse artifacts. Evidence-aware generation should therefore support full, partial, or deferred artifacts depending on available evidence, with missing fields treated as signals of insufficient source support.

4 *SRS reconstruction is feasible, but factuality must be checked across artifact chains.* RQ4 shows that LLM-generated pre-SRS artifacts can support SRS reconstruction with partial source recoverability. However, synthetic details may propagate into fluent, structurally valid text that remains only partially supported by the original atoms. Future work should therefore evaluate factuality across the full source SRS→Atoms→Artifacts→reconstructed SRS chain and review generated drafts through their trace links.

5 *The main bottleneck is evidence granularity.* RQ5 shows that information loss often occurs because evidence is too implicit, bundled, or too narrow for the next stage of transformation. Context-dependent statements may be difficult to express as standalone atoms, while valid atoms may still be too narrow to instantiate complete artifacts. Future LLM4RE systems should preserve context at appropriate granularity through related-atom grouping, context-only evidence, and partially specified artifacts.

6 Related Work

Recent work shows that LLMs can support requirements engineering (RE) activities such as elicitation, user-story generation, acceptance-criteria generation, functional design specification, and SRS drafting [33, 30, 34, 29, 26, 25, 36, 40]. Broader Large Language Models for Software Engineering (LLM4SE) surveys also report increasing use of LLMs across software engineering tasks [16, 22, 38]. These studies demonstrate that LLMs can generate useful requirements artifacts, whereas our work studies a different problem: how legacy SRS content can be decomposed into traceable atomic evidence, routed to typed pre-SRS artifacts, and evaluated for downstream reconstruction.

NLP4RE has long studied requirements classification, extraction, ambiguity analysis, traceability, and quality assessment [39, 3, 21, 14, 12, 20]. Recent work also explores extracting models or requirements from agile backlogs, app reviews, and other textual sources [6, 15], as well as using NLI for RE classification and defect or conflict analysis [17]. Closest to our setting, Okamoto et al. [28] study LLM-based restructuring of organically written SRSs into ISO/IEC/IEEE 29148-compliant documents. Our work shares the goal of making legacy SRSs more structured and analyzable, but operates at a finer granularity by decomposing sections into atomic propositions before routing them to pre-SRS artifact types.

Public datasets such as PURE [18] support reproducible NLP4RE research, while recent work calls for reusable benchmarks, clearer reporting, and stronger traceability resources [3, 23]. Traceability recovery research further emphasizes linking information across heterogeneous lifecycle artifacts [19]. *ReqGenX* contributes to this direction by using public SRSs as seed evidence for constructing traceable synthetic pre-SRS artifacts.

Taken together, these strands intersect with *ReqGenX* along several dimensions but address different transformation goals. LLM-based requirements and SRS-generation studies [25, 36, 40] primarily generate or assess requirements artifacts or documents, whereas *ReqGenX* studies provenance-preserving transformations across intermediate representations. SRS-restructuring work, such as Okamoto et al. [28], transforms existing specifications toward standardized document structures, while *ReqGenX* first decomposes legacy SRS content into claim-level atoms and routes those atoms into typed synthetic pre-SRS artifacts. Traceability-recovery research [19] links information across existing lifecycle artifacts, whereas *ReqGenX* constructs an explicit SRS-to-atom-to-artifact provenance chain so that information preservation and loss can be measured during generation. We therefore position *ReqGenX* as a controlled empirical evaluation pipeline rather than a general-purpose SRS-generation method.

7 Threats to Validity

Construct validity. Our metrics assess the quality of traceability-grounded proxies rather than the completeness of requirements. AlignScore measures factual support, SBERT semantic recoverability, and judge/Prometheus scores rubric-based quality; none establishes full SRS coherence, ISO conformance, or practitioner usefulness. Likewise, *downstream utility* indicates whether an atom can instantiate an artifact type, not whether practitioners would find it useful. Human assessment primarily supports routing review and judge calibration; the 291-triple Claude Sonnet 4.6 sample is not a broad human evaluation across models and datasets. Public PURE and PROMISE-NFR data may also introduce benchmark-exposure effects [32]. We mitigate these threats through human-reconciled labels, external Prometheus assessment, and complementary coverage, agreement, and retention analyses.

Internal validity. Multiple LLM-based stages introduce sensitivity to prompts, model updates, decoding, and nondeterminism, while using LLMs for both generation and evaluation may introduce correlated bias. We mitigate these risks through plurality voting, human-calibrated judge selection, external Prometheus evaluation, multiple model families, model-independent grounding measures, fixed prompts and codebooks, traceability metadata, and manual input checks. Nevertheless, automated judgments remain calibrated proxies rather than human ground truth. Moreover, at temperature 1.0 and with one generation run per configuration, our results do not demonstrate stability across repeated runs.

External validity. Our evaluation covers seven selected text-oriented PURE SRSs and a fixed set of synthetic artifact types; results may differ for proprietary, safety-critical, multimodal, table-heavy, or actively maintained industrial requirements. PURE also does not capture the evolving and negotiation-heavy evidence of active projects. Because our study begins with authored SRSs, it does not evaluate elicitation from interviews, stakeholder discussions, issue reports, app reviews, regulations, or other primary evidence. The generated pre-SRS artifacts are therefore synthetic proxies, not recovered practitioner-authored artifacts, and reconstruction should not be interpreted as end-to-end SRS generation from raw stakeholder information. Future work should evaluate industrial and primary elicitation data with broader practitioner assessment.

Conclusion validity. Our analyses involve a few SRS documents, nested evaluation units, fixed thresholds, and one generation run per model. We use dataset-level aggregation, distributional summaries, paired bootstrap comparisons where applicable, and cautious interpretation of model differences. We do not conduct component-level ablations or a full baseline comparison, so we cannot isolate the effects of plurality voting, judge-guided refinement, or artifact-specific templates. Existing end-to-end baselines also lack the intermediate provenance required for several analyses. Future work should compare single-model versus plurality routing, refinement versus no refinement, and generic versus artifact-specific generation. Accordingly, our conclusions concern feasibility, tradeoffs, and failure modes rather than statistical superiority, practitioner readiness, or deployment effectiveness.

8 Conclusion

We conducted a controlled empirical study of how information changes when legacy SRS content is decomposed into atomic evidence, routed to synthetic pre-SRS artifacts, and used for downstream reconstruction, using *ReqGenX* as a traceable evaluation pipeline. Across seven selected text-only PURE SRSs, the results show that LLMs can produce well-formed, strongly grounded atomic statements and typed artifacts, but these transformations are not lossless. Semantic routing and quality judgment require human-grounded model selection, strong artifact grounding does not guarantee structural completeness, and downstream reconstruction preserves source information only partially. These findings show that traceable synthetic artifacts can support fine-grained evaluation of LLM-based requirements transformations while exposing tradeoffs among faithfulness, coverage, sparsity, and recoverability. Future work should extend the study to primary elicitation sources, industrial and multimodal requirements, broader practitioner evaluation, simpler component-level baselines and ablations, and loss-aware decomposition or more flexible artifact schemas for partially specified evidence. Overall, the results support *ReqGenX* primarily as traceable experimental infrastructure for studying LLM-based requirements transformations, rather than as evidence of an end-to-end or deployment-ready SRS-generation solution.

Data Availability Statement

To support reproducibility, we provide a stable repository [8] containing processed inputs, generated artifacts, traceability metadata, evaluation outputs, scripts, prompts, templates, configurations, and documentation.

References

- 1 IEEE recommended practice for software requirements specifications, 1994. doi:10.1109/IEEESTD.1994.121431.
- 2 ISO/IEC/IEEE international standard - systems and software engineering – life cycle processes – requirements engineering, 2018. doi:10.1109/IEEESTD.2018.8559686.
- 3 Sallam Abualhaija, Fatma Basak Aydemir, Fabiano Dalpiaz, Davide Dell’Anna, Alessio Ferrari, Xavier Franch, and Davide Fucci. Replication in requirements engineering: The NLP for RE case. *ACM Trans. Softw. Eng. Methodol.*, 33(6):151, June 2024. doi:10.1145/3658669.
- 4 Shibbir Ahmed, Hongyang Gao, and Hridesh Rajan. Inferring data preconditions from deep learning models for trustworthy prediction in deployment. In *Proceedings of the 46th IEEE/ACM International Conference on Software Engineering, ICSE 2024, Lisbon, Portugal, April 14-20, 2024*, ICSE ’24, pages 38:1–38:13, New York, NY, USA, 2024. ACM. doi:10.1145/3597503.3623333.
- 5 Shibbir Ahmed, Sayem Mohammad Imtiaz, Samantha Syeda Khairunnesa, Breno Dantas Cruz, and Hridesh Rajan. Design by contract for deep learning APIs. In *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/FSE 2023*, pages 94–106, New York, NY, USA, 2023. Association for Computing Machinery. doi:10.1145/3611643.3616247.
- 6 Sathurshan Arulmohan, Marie-Jean Meurs, and Sébastien Mosser. Extracting domain models from textual requirements in the era of large language models. In *2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, pages 580–587. IEEE Press, 2023. doi:10.1109/MODELS-C59198.2023.00096.
- 7 Ragib Shahariar Ayon and Shibbir Ahmed. AutoReSpec: A framework for generating specification using large language models. In *Proceedings of the 2026 IEEE/ACM Third International Conference on AI Foundation Models and Software Engineering, FORGE ’26*, pages 29–39, New York, NY, USA, 2026. Association for Computing Machinery. doi:10.1145/3793655.3793731.
- 8 Ragib Shahariar Ayon, Rayed Fahmi, Sumon Biswas, and Shibbir Ahmed. Replication package for research paper titled "ReqGenX: An empirical study of atomic decomposition, artifact regeneration, and reconstruction for legacy srs documents", May 2026. doi:10.5281/zenodo.20267334.
- 9 Ragib Shahariar Ayon, Mohammad Wardat, and Shibbir Ahmed. From discussion to execution: Replicating buggy and correct data science code. *CoRR*, abs/2607.16569, 2026. doi:10.48550/arXiv.2607.16569.
- 10 Alberto Bacchelli, Anthony Cleve, Michele Lanza, and Andrea Mocci. Extracting structured data from natural language documents with island parsing. In *2011 26th IEEE/ACM International Conference on Automated Software Engineering (ASE 2011)*, pages 476–479. IEEE, 2011. doi:10.1109/ASE.2011.6100103.
- 11 Kevin Brennan et al. *A guide to the Business Analysis Body of Knowledge (BABOK Guide)*. IIBA, 2009.
- 12 Jane Cleland-Huang, Raffaella Settini, Xuchang Zou, and Peter Solc. Automated classification of non-functional requirements. *Requirements engineering*, 12(2):103–120, 2007. doi:10.1007/s00766-007-0045-1.
- 13 Alistair Cockburn. Basic use case template. *Humans and Technology, Technical Report*, 96:28, 1998.
- 14 Fabiano Dalpiaz, Davide Dell’Anna, Fatma Basak Aydemir, and Sercan Çevikol. Requirements classification with interpretable machine learning and dependency parsing. In *2019 IEEE 27th International Requirements Engineering Conference (RE)*, pages 142–152. IEEE, 2019. doi:10.1109/RE.2019.00025.
- 15 Adailton Ferreira de Araújo and Ricardo Marcondes Marcacini. RE-BERT: automatic extraction of software requirements from app reviews using bert language model. In *Proceedings of the 36th Annual ACM Symposium on Applied Computing, SAC ’21*, pages 1321–1327, New York, NY, USA, 2021. Association for Computing Machinery. doi:10.1145/3412841.3442006.

- 16 Angela Fan, Beliz Gokkaya, Mark Harman, Mitya Lyubarskiy, Shubho Sengupta, Shin Yoo, and Jie M Zhang. Large language models for software engineering: Survey and open problems. In *2023 IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FoSE)*, pages 31–53. IEEE, 2023. doi:10.1109/ICSE-FoSE59343.2023.00008.
- 17 Mohamad Fazelnia, Viktoria Koscinski, Spencer Herzog, and Mehdi Mirakhorli. Lessons from the use of natural language inference (nli) in requirements engineering tasks. In *2024 IEEE 32nd International Requirements Engineering Conference (RE)*, pages 103–115. IEEE, 2024. doi:10.1109/RE59067.2024.00020.
- 18 Alessio Ferrari, Giorgio Oronzo Spagnolo, and Stefania Gnesi. Pure: A dataset of public requirements documents. In *2017 IEEE 25th international requirements engineering conference (RE)*, pages 502–505. IEEE, 2017. doi:10.1109/RE.2017.29.
- 19 Hui Gao, Hongyu Kuang, Wesley KG Assunção, Christoph Mayr-Dorn, Guoping Rong, He Zhang, Xiaoxing Ma, and Alexander Egyed. Triad: Automated traceability recovery based on biterm-enhanced deduction of transitive links among artifacts. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*, pages 1–13, 2024. doi:10.1145/3597503.3639164.
- 20 Gonzalo Génova, José M Fuentes, Juan Llorens, Omar Hurtado, and Valentín Moreno. A framework to measure and improve the quality of textual requirements. *Requirements engineering*, 18(1):25–41, 2013. doi:10.1007/s00766-011-0134-z.
- 21 Tobias Hey, Jan Keim, Anne Koziol, and Walter F Tichy. Norbert: Transfer learning for requirements classification. In *2020 IEEE 28th international requirements engineering conference (RE)*, pages 169–179. IEEE, 2020. doi:10.1109/RE48521.2020.00028.
- 22 Xinyi Hou, Yanjie Zhao, Yue Liu, Zhou Yang, Kailong Wang, Li Li, Xiapu Luo, David Lo, John Grundy, and Haoyu Wang. Large language models for software engineering: A systematic literature review. *ACM Transactions on Software Engineering and Methodology*, 33(8):1–79, 2024. doi:10.1145/3695988.
- 23 Xing Hu, Feifei Niu, Junkai Chen, Xin Zhou, Junwei Zhang, Junda He, Xin Xia, and David Lo. Assessing and advancing benchmarks for evaluating large language models in software engineering tasks. *ACM Trans. Softw. Eng. Methodol.*, December 2026. Just Accepted. doi:10.1145/3786771.
- 24 Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. Prometheus 2: An open source language model specialized in evaluating other language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4334–4353, 2024. doi:10.18653/v1/2024.emnlp-main.248.
- 25 Madhava Krishna, Bhagesh Gaur, Arsh Verma, and Pankaj Jalote. Using llms in software requirements specifications: An empirical evaluation. In *2024 IEEE 32nd International Requirements Engineering Conference (RE)*, pages 475–483. IEEE, 2024. doi:10.1109/RE59067.2024.00056.
- 26 Christopher Lazik, Charlotte Kauter, Inês Nunes, Aaron Ziglowski, Alina Pryma, Christopher Katins, Lars Grunske, and Thomas Kosch. The good, the bad, and the uncanny: Investigating diversity aspects of llm-generated personas for requirements engineering. In *2025 IEEE 33rd International Requirements Engineering Conference (RE)*, pages 244–256. IEEE, 2025. doi:10.1109/RE63999.2025.00031.
- 27 F.-L. Li, J. Horkoff, J. Mylopoulos, A. Borgida, R. Guizzardi, G. Guizzardi, L. Liu, and Y. Peng. Dataset for a requirements modeling language and the desiree framework, 2024. Data set. doi:10.5281/zenodo.14330485.
- 28 Ryu Okamoto and Shinji Kusumoto. Towards the Automatic Restructuring of Software Requirements Specifications to Conform to Standards Using Large Language Models. In *2025 IEEE 33rd International Requirements Engineering Conference (RE)*, pages 467–475, Los Alamitos, CA, USA, September 2025. IEEE Computer Society. doi:10.1109/RE63999.2025.00056.

- 29 Liliana Pasquale, Azzurra Ragone, Emanuele Piemontese, and Armin Amiri Darban. Exploring the use of llms for requirements specification in an it consulting company. In *2025 IEEE 33rd International Requirements Engineering Conference (RE)*, pages 389–399. IEEE, 2025. doi:10.1109/RE63999.2025.00045.
- 30 Giovanni Quattrocchi, Liliana Pasquale, Paola Spoletini, and Luciano Baresi. Can llms generate user stories and assess their quality? *IEEE Transactions on Software Engineering*, 2026. doi:10.1109/TSE.2026.3670612.
- 31 Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 3982–3992, 2019. doi:10.18653/v1/D19-1410.
- 32 Martin Riddell, Ansong Ni, and Arman Cohan. Quantifying contamination in evaluating code generation capabilities of language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14116–14137, 2024. doi:10.18653/v1/2024.acl-long.761.
- 33 Krishna Ronanki, Christian Berger, and Jennifer Horkoff. Investigating chatgpt’s potential to assist in requirements elicitation processes. In *2023 49th Euromicro conference on software engineering and advanced applications (SEAA)*, pages 354–361. IEEE, 2023. doi:10.1109/SEAA60479.2023.00061.
- 34 Tjerk Spijkman, Bente Molenkamp, Steffen Beudeker, Sietse Overbeek, and Fabiano Dalpiaz. Llm-assisted requirements engineering in agile mdd: Industry insights and validation. In *2025 IEEE 33rd International Requirements Engineering Conference (RE)*, pages 366–377. IEEE, 2025. doi:10.1109/RE63999.2025.00043.
- 35 Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. arXiv:2203.11171.
- 36 Danning Xie, Byoungwoo Yoo, Nan Jiang, Mijung Kim, Lin Tan, Xiangyu Zhang, and Judy S Lee. How effective are large language models in generating software specifications? In *2025 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 1–12. IEEE, 2025. doi:10.1109/SANER64311.2025.00014.
- 37 Yuheng Zha, Yichi Yang, Ruichen Li, and Zhiting Hu. Alignscore: Evaluating factual consistency with a unified alignment function. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11328–11348, 2023. doi:10.18653/v1/2023.acl-long.634.
- 38 Quanjun Zhang, Chunrong Fang, Yang Xie, Yaxin Zhang, Shengcheng Yu, Weisong Sun, Yun Yang, and Zhenyu Chen. A survey on large language models for software engineering. *Science China Information Sciences*, 69(4):141102, 2026. doi:10.1007/s11432-025-4670-0.
- 39 Liping Zhao, Waad Alhoshan, Alessio Ferrari, Keletso J Letsholo, Muideen A Ajagbe, Erol-Valeriu Chioasca, and Riza T Batista-Navarro. Natural language processing for requirements engineering: A systematic mapping study. *ACM Computing Surveys (CSUR)*, 54(3):1–41, 2021. doi:10.1145/3444689.
- 40 Taohong Zhu, Lucas C Cordeiro, and Youcheng Sun. Requinone: A large language model-based agent for software requirements specification generation. In *2025 IEEE 33rd International Requirements Engineering Conference (RE)*, pages 449–457. IEEE, 2025. doi:10.1109/RE63999.2025.00054.
- 41 Xiaochen Zhu, Caiqi Zhang, Tom Stafford, Nigel Collier, and Andreas Vlachos. Conformity in large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3854–3872, 2025. doi:10.18653/v1/2025.acl-long.195.