Semantic Traceability in Requirements Engineering: Integrating Text, Interface, and Tests

Suelen S. Pereira
Universidade Federal do Amazonas (UFAM)
Manaus, Brazil
suelen.pereira@icomp.ufam.edu.br

Bruno Freitas Gadelha
Universidade Federal do Amazonas (UFAM)
Manaus, Brazil
bruno@icomp.ufam.edu.br

Rayfran Rocha Lima
Sidia Institute of Science and Technology
Manaus, Brazil
rayfran.rocha.lima@gmail.com

Ana Carolina Oran
Universidade Federal do Amazonas (UFAM)
Manaus, Brazil
ana.oran@icomp.ufam.edu.br

Abstract

Software functionalities are frequently distributed across heterogeneous development artifacts, such as textual requirements and Visual Interface Artifacts (VIA). Although previous studies increasingly automate Requirements Engineering, traceability, and Verification and Validation (V&V) activities using textual artifacts, existing approaches predominantly treat interfaces as operational targets for GUI automation rather than as complementary semantic sources for traceability and validation activities. In this context, this work proposes a multimodal workflow for semantic traceability that integrates textual requirements analysis and interface-aware validation through VIA. The proposed approach combines semantic extraction from User Stories using RoBERTa-based Named Entity Recognition with Large Language Models capable of processing both textual requirements and VIA screenshots extracted from implemented systems. Semantic correspondence between requirement-derived and VIA-derived artifacts is investigated through cosine similarity using the all-mpnet-base-v2 model. The workflow was evaluated through an exploratory empirical study involving 15 software projects, 318 US, and 140 VIA screenshots. The results suggest that VIA-oriented analysis may expand validation coverage by identifying interaction-oriented validation scenarios not explicitly represented in textual specifications. In several projects, VIA-aware analysis generated validation artifacts even when textual requirements could not support automated test derivation due to structural inconsistencies or insufficient semantic information. In addition, the multimodal traceability analysis revealed potential semantic gaps, undocumented functionalities, and inconsistencies between business specifications and implemented interfaces. The findings provide initial evidence that VIA may act as complementary semantic artifacts for multimodal traceability and interface-aware V&V activities, suggesting the potential of integrating textual and visual perspectives in software requirements engineering.

Keywords

Requirements Traceability, Multimodal Requirements Engineering, Interface-aware Verification and Validation

1 Introduction

Software functionalities are rarely fully represented in a single development artifact. In modern software projects, business rules, interaction constraints, validation behaviors, and operational flows are frequently distributed across multiple artifacts throughout the software lifecycle [4]. While textual requirements describe business goals and expected behaviors, relevant interaction details often emerge through visual representations of the system interface. This fragmentation is particularly critical in Requirements Engineering (RE) and Verification and Validation (V&V), where incomplete or ambiguous specifications may lead to inconsistencies, rework, and increased development costs [8]. Ambiguity arising from communication issues and documentation inconsistencies can affect software quality, project deadlines, and customer satisfaction [11, 16].

To address these challenges, previous studies have explored automation techniques for semantic extraction, traceability, and test generation from textual artifacts using Natural Language Processing (NLP) [10], Named Entity Recognition (NER) and Machine Learning [6], and Large Language Models (LLMs) Wolf et al. [19]. In parallel, GUI-oriented testing approaches employ deep learning and multimodal models to automate interface interaction through Visual Interface Artifacts (VIA), such as screenshots and mockups [9, 15, 17]. However, these approaches primarily treat VIA as operational targets for GUI automation rather than as semantic artifacts capable of complementing textual requirements in traceability and V&V activities.

This limitation is relevant because VIA often preserve validation rules, interaction constraints, navigation flows, and operational behaviors that are absent or only partially represented in textual specifications [12]. Consequently, text-centered approaches may overlook interface-level behaviors, undocumented functionalities, and validation scenarios present in implemented systems [3]. Despite advances in requirements automation and GUI testing, empirical evidence on combining textual requirements and VIA for multimodal traceability remains limited.

This work investigates whether VIA provide complementary functional semantics capable of supporting RE and V&V beyond what is explicitly documented in textual requirements. We propose a multimodal workflow integrating semantic extraction from User Stories (US) with interface-aware analysis to support semantic traceability and validation. The empirical evaluation involved 15 real-world software projects, comprising 318 US and 140 VIA screenshots extracted from implemented systems. The results suggest that interface-aware analysis expands validation coverage, particularly

when textual requirements are incomplete or structurally inconsistent, while revealing semantic gaps, undocumented functionalities, and interface-level validation scenarios not captured by textual artifacts alone.

The main contributions of this work are: (i) an exploratory perspective on VIA as complementary semantic artifacts for multimodal traceability and V&V activities; (ii) a multimodal workflow integrating textual requirements and interface-aware analysis for semantic traceability and automated validation; (iii) an empirical investigation of how VIA complement text-centered approaches by identifying validation scenarios, traceability gaps, and undocumented functionalities; and (iv) initial evidence that multimodal analysis reveals interface-level validation scenarios and semantic inconsistencies not captured through textual artifacts alone.

2 Background and Related Work

Previous studies increasingly explore automation techniques to support RE, traceability, and V&V activities from textual artifacts. Khan et al. [10] discuss the use of Natural Language Processing (NLP) techniques for semantic extraction and automated requirements analysis. Das et al. [6] employ Named Entity Recognition (NER) and Machine Learning methods to identify actors, actions, and contextual information from textual specifications. In addition, Wolf et al. [19] investigate the application of Large Language Models (LLMs) for requirements validation, ambiguity detection, and automated test generation from US.

Requirements traceability and automated test generation are similarly explored through text-centered approaches. Mustafa et al. [14] use Model-Driven Testing (MDT) techniques to derive test cases from US and textual requirements. Antoniol et al. [2] investigate traceability recovery and semantic alignment between software artifacts. Luttmer et al. [13] analyze the impact of inconsistencies and semantic noise on automated requirements extraction. Xu et al. [20] propose techniques to improve the organization and refinement of US before automated processing. Similarly, Behutiye et al. [3] discuss quality requirements documentation practices to reduce communication failures and implementation inconsistencies. Overall, these studies demonstrate that current automation and traceability approaches remain predominantly centered on textual artifacts as the primary semantic source for RE and V&V activities.

In parallel, GUI-oriented testing approaches increasingly employ deep learning, computer vision, and multimodal models to automate interface interaction and graphical testing activities through VIA. Khaliq et al. [9] propose a deep learning-based framework for functional User Interface testing. Patel [17] discuss AI-driven approaches for automated software quality engineering. In addition, Nie et al. [15] highlight the growing adoption of graphical interface analysis in mobile application testing through a systematic mapping study. Other studies investigate interface exploration and GUI-oriented validation workflows through automated interaction analysis [5]. However, these approaches predominantly treat VIA as operational targets for automation rather than as complementary semantic sources for traceability and V&V activities.

This limitation becomes relevant because software functionalities are frequently distributed across multiple development artifacts [4]. While textual requirements describe business goals and expected behaviors, Kolthoff et al. [12] argue that VIA often preserve complementary operational semantics, including validation rules, interaction constraints, navigation flows, and interface-level workflows that are absent or only partially represented in textual specifications. Consequently, approaches relying exclusively on textual artifacts may overlook interface-level behaviors and validation scenarios present in implemented systems but absent from written requirements [3].

Despite advances in automated requirements analysis and GUI testing, limited empirical evidence exists regarding how VIA may complement textual requirements in multimodal traceability and interface-aware V&V activities. Unlike approaches that depend on direct access to executable GUIs, VIA-based analysis can also be applied when only screenshots, prototypes, mockups, or archived interface documentation are available. This characteristic makes the approach technology-independent and suitable for scenarios in which executable interfaces or source code are inaccessible. In particular, limited attention has been given to investigating how combining textual specifications and VIA may contribute to identifying semantic inconsistencies, undocumented functionalities, and validation scenarios not captured through text-centered approaches.

Unlike the studies discussed in this section, which independently focus on textual requirements analysis or GUI-based testing, the proposed workflow analyzes User Stories and Visual Interface Artifacts (VIA) as complementary sources of software requirements. Rather than using VIA only for interface testing or textual artifacts only for requirements analysis, this study investigates how combining these complementary artifacts may contribute to semantic traceability and Requirements Engineering verification and validation (V&V) activities.

3 Proposed Method

This work proposes a multimodal workflow for semantic traceability by integrating textual requirements analysis and interface-aware test generation. The methodological workflow illustrated in Figure 1 combines US and VIA as complementary semantic artifacts to support V&V activities. In this study, VIA were instantiated as screenshots extracted from systems already deployed in production environments.

The approach is structured into two complementary analysis flows: Flow A, focused on textual requirements, and Flow B, focused on VIA. Both flows converge into a semantic concordance analysis stage responsible for identifying relationships between requirements-derived and VIA-derived artifacts.

The workflow begins with a multimodal data collection stage composed of US and VIA screenshots extracted from implemented systems. These artifacts are independently processed to extract complementary semantic perspectives associated with software functionalities and interaction behaviors.

Flow A performs semantic extraction from US using Natural Language Processing (NLP) techniques. Initially, textual artifacts undergo normalization and noise reduction procedures. Subsequently, a RoBERTa-based Named Entity Recognition (NER) approach Abdullah et al. [1] identifies actors, actions, and contextual entities

associated with software functionalities. Additional validation procedures verify the structural consistency of US and support the identification of incomplete or duplicated requirements artifacts.

After semantic structuring, Large Language Models (LLMs) are employed to generate requirement-oriented test scenarios derived from business specifications. This stage focuses on transforming textual requirements into validation artifacts aligned with user intentions and described functionalities.

Flow B focuses on extracting interaction semantics directly from VIA screenshots. The screenshots are processed to identify visual components, interaction elements, validation constraints, and navigation behaviors associated with implemented interfaces. Based on these VIA, LLM-supported processing generates interface-oriented test scenarios associated with visible interaction flows and operational constraints.

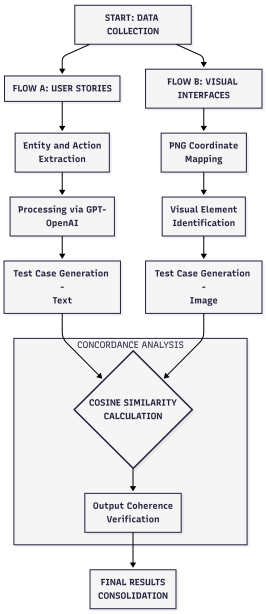

Figure 1: Methodological workflow

After the independent generation of textual and VIA-oriented validation artifacts, both flows converge into a concordance analysis stage. In this stage, semantic similarity computation is performed using the all-mpnet-base-v2 model [7] to investigate alignment relationships between requirements-derived artifacts and VIA-derived artifacts. The model was selected due to its strong performance in semantic textual similarity tasks and widespread adoption in sentence-level semantic embedding studies.

The similarity analysis supports the identification of semantic correspondence, coverage gaps, and inconsistencies between textual specifications and interface-level behaviors represented in VIA. Subsequently, an output coherence verification stage evaluates the semantic consistency of the generated artifacts before consolidating the final traceability results.

4 Experimental Evaluation

The evaluation was designed as an exploratory feasibility assessment rather than a controlled effectiveness experiment. The objective was to investigate whether multimodal semantic traceability could reveal complementary validation perspectives and semantic relationships between textual requirements and Visual Interface Artifacts (VIA). Therefore, the analysis focused on observing semantic behavior patterns, traceability relationships, and multimodal validation scenarios generated by the proposed workflow across heterogeneous software artifacts.

4.1 Study Context and Dataset

The empirical evaluation was conducted using artifacts produced in a Software Engineering course offered at a Brazilian higher education institution. The dataset comprises 15 software projects containing 318 User Stories (US) and 140 VIA screenshots extracted from systems deployed in production environments.

The projects spanned different application domains, including education, household management, productivity, cultural preservation, tourism, healthcare, and career-oriented applications, providing heterogeneous software artifacts for the evaluation.

Throughout the course, students collaboratively develop real-world software projects covering the complete software engineering lifecycle, including requirements elicitation, analysis, implementation, testing, deployment, and maintenance activities. The course emphasizes hands-on project execution in environments that simulate professional software development practices, requiring teams to work with agile methods, version control systems, continuous integration and delivery pipelines, cloud-based environments, and collaborative project management tools.

Although the artifacts were produced in an academic context, the projects represent operational software systems developed under realistic software engineering practices. This setting enabled an observational evaluation of the proposed multimodal workflow in scenarios involving heterogeneous requirements artifacts, VIA evolution, and iterative validation activities.

The User Stories and VIA screenshots analyzed in this study were collected from these projects. The proposed workflow and all experimental analyses were conducted exclusively by the authors.

The experimental workflow was implemented through Python scripts executed in a Google Colab environment and investigated semantic extraction, multimodal test generation, and traceability analysis across textual artifacts and VIA.

4.2 Processing and Validation of User Stories

The textual analysis stage investigated semantic extraction from 318 US to support automated traceability and test generation activities.

Initially, textual artifacts underwent normalization and noise reduction procedures using regular expressions to standardize input data and remove semantic noise. Subsequently, a hybrid approach combining RoBERTa-large Zhao et al. [21] and the *spaCy* library (*pt_core_news_sm*) performed Named Entity Recognition (NER) and morphological analysis to identify entities associated with systems, modules, and software functionalities described in US.

To refine entity identification, the workflow combined entities extracted by RoBERTa with regular expression patterns and heuristic validation procedures based on contextual filtering and stopword analysis. Additional linguistic rules validated word size, structure, and grammatical consistency. At the end of the extraction process, identified entities received confidence scores according to extraction source and validation consistency.

In parallel, the workflow analyzed the structural consistency of US by identifying Actors, Actions, and Benefits. Requirements containing all mandatory elements were classified as compliant and considered eligible for subsequent traceability and testing activities.

After semantic validation, textual similarity analysis was performed to identify duplicated requirements and conflicting actions associated with the same actor. Finally, the structured dataset was exported in JSON format for subsequent processing stages.

4.3 Requirement-Oriented Test Generation

After semantic extraction and validation, compliant US were processed to derive requirement-oriented test cases. For each eligible requirement, the workflow identified associated entities and contextual information extracted during the semantic analysis stage.

Requirements associated with entities above the confidence threshold ($\geq$ 0.7) or satisfying default classification rules were considered eligible for test generation. Based on these structured artifacts, GPT-4o-mini generated test cases containing Titles, Preconditions, Steps, Inputs, and Expected Results aligned with business requirements.

After generation, regex-based routines normalized the responses and separated each generated test case into individual structured artifacts.

4.4 VIA-Oriented Test Generation

The VIA analysis stage investigated the generation of validation artifacts directly from interface semantics. A total of 140 VIA screenshots extracted from deployed systems were individually processed to identify interaction-oriented testing information from visible interface elements.

Before submission to the OpenAI GPT-4o-mini API, a preprocessing stage filtered low-resolution images and converted valid VIA screenshots into base64 format. Subsequently, the workflow employed LLM-based analysis to identify visible interface components, including fields, buttons, interaction controls, icons, and navigation elements.

The prompt instructed the model to objectively describe visible interface structures and generate test scenarios containing identifiers, interaction types, target interface elements, descriptions, and expected results. Based on these VIA, the model generated structured validation scenarios associated with interaction flows, validation behaviors, and expected interface responses.

To reduce output variation and preserve structural consistency across generated artifacts, the model operated with deterministic parameters: *temperature* = 0.0 and standardized JSON outputs.

4.5 Multimodal Traceability Mapping

The final stage integrates the textual and VIA-oriented flows through multimodal traceability analysis. Semantic similarity computation was performed using the sentence transformer model *all-mpnet-base-v2* to investigate alignment relationships between requirement-derived artifacts and VIA-derived artifacts.

Before similarity computation, preprocessing routines removed noisy terms from both the original US and the generated test cases to reduce semantic bias during similarity analysis. The matching process computes cosine similarity between user story actions and generated validation artifacts.

Based on semantic similarity scores, the workflow classifies multimodal traceability relationships into three semantic correspondence categories: **Validated** (> 0.65), representing strong semantic correspondence with aligned business-oriented and operational intent; **Potential** ($0.40 - 0.65$), representing partial semantic overlap involving vocabulary divergence, incomplete contextual alignment, or operational variations; and **No Match** (< 0.40), representing insufficient semantic evidence of correspondence between textual and VIA-derived artifacts.

The classification thresholds were defined through exploratory calibration cycles involving manual inspection of representative artifact pairs to identify intervals capable of representing distinct levels of semantic correspondence. Table 1 illustrates representative examples of the three traceability relationship categories identified by the workflow.

Table 1: Examples of multimodal traceability relationship categories

Relationship	Example
Validated	"Create account" ↔ "Register user form"
Potential	"Submit request" ↔ "Open ticket"
No Match	"Reset password" ↔ "Generate invoice"

Similarity thresholds were refined through exploratory evaluation cycles and manual inspection of generated artifacts by two researchers to identify intervals capable of representing meaningful semantic correspondence relationships.

In addition to cosine similarity, representative artifact pairs were manually inspected by the authors to verify whether the identified semantic correspondences were consistent with the original User Stories. This qualitative assessment was used to interpret representative traceability relationships and support the analysis presented in the experimental evaluation.

Finally, the workflow consolidates a comparative matrix indicating whether each user demand was covered by the textual approach, the VIA-oriented approach, or both.

5 Results and Discussion

The experimental results suggest that textual artifacts and Visual Interface Artifacts (VIA) provide complementary semantic perspectives for requirements traceability and Verification and Validation (V&V) activities. While textual analysis preserves stronger alignment with business-oriented specifications, interface-aware analysis expands validation coverage by identifying interaction elements and operational behaviors not explicitly represented in User Stories (US).

From the 318 analyzed US, 205 satisfied the structural validation criteria required for automated semantic extraction. These compliant artifacts enabled the generation of 203 requirement-oriented test cases aligned with business specifications. Table 2 summarizes the results obtained from textual and VIA-oriented analysis across the evaluated projects, including requirement compliance, generated validation artifacts, and multimodal traceability outcomes.

The multimodal traceability mapping stage computed semantic similarity relationships between requirement-derived and VIA-derived artifacts. From the final results, 54 relationships were classified as *Validated*, 159 as *Potential*, and 07 as *No Match*. The predominance of *Potential* relationships suggests that the workflow frequently identified semantic correspondence between textual requirements and VIA-derived validation artifacts despite vocabulary divergence between business-oriented specifications and interface implementation terminology.

The results also indicate that textual traceability strongly depends on the structural quality and semantic consistency of US. Requirements containing clear Actors, Actions, and Benefits were more effectively transformed into validation artifacts, suggesting that the quality of textual specifications directly influences automated traceability outcomes.

In contrast, the VIA-oriented analysis substantially expanded validation coverage by generating interaction-oriented validation scenarios directly from implemented interface elements. Across the 140 analyzed VIA screenshots, the workflow generated 779 interface-oriented validation scenarios, averaging 5.56 scenarios per VIA. The generated artifacts describe interaction paths, validation elements, navigation behaviors, and interface constraints that are frequently summarized or omitted in textual requirements.

This behavior becomes more evident when comparing the distribution of textual and VIA-oriented validation artifacts across projects. Figure 2 illustrates the quantitative contrast between requirement-oriented and interface-oriented validation scenarios generated by the workflow.

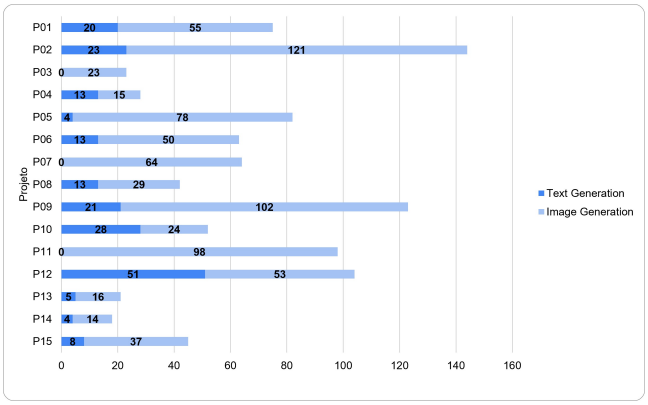

Figure 2: Quantitative comparison between validation scenarios generated through textual and VIA-oriented analysis per project.

Although the study was not designed to establish causal relationships through controlled experimentation, the observed results suggest that VIA-aware analysis may reveal complementary semantic information not explicitly represented in business-oriented specifications.

A particularly relevant finding emerged in Projects 03, 07, and 11, where the US were not classified as compliant during the structural validation stage. Because these requirements did not contain the mandatory semantic elements required by the workflow, no requirement-oriented test cases were generated from the textual artifacts. Nevertheless, the VIA-oriented workflow independently generated validation scenarios directly from visible interaction elements, enabling traceability and validation activities even in scenarios with insufficient textual specification quality.

This behavior suggests that VIA may preserve operational semantics even when textual requirements are incomplete or insufficiently structured for automated processing. In this context, VIA-aware analysis demonstrated the potential to complement text-centered approaches by exposing interaction semantics embedded in implemented interfaces.

Qualitative inspection of the generated artifacts further reinforces this behavior. In Project 13, VIA-oriented analysis identified operational interface elements such as the “Create Report” button and search fields that did not present explicit correspondence with textual requirements, revealing potential undocumented functionalities and interface-level validation scenarios. Similarly, in Project 14, VIA-derived artifacts exposed operational interaction elements such as filters, login fields, and spreadsheet export actions that were represented only partially in textual specifications.

Artifacts classified as **No Match** reveal potential inconsistencies between documented requirements and implemented interface elements, including orphan interface components and functionalities without explicit traceability relationships. These findings reinforce the potential of multimodal approaches to reduce semantic gaps between requirements documentation and implemented systems, particularly in scenarios involving incomplete or inconsistent textual specifications.

6 Threats to Validity

This section discusses potential threats to validity following the perspectives proposed by Wohlin et al. [18]. **Internal validity.** Variability in LLM-based semantic interpretation and inconsistencies in User Stories (US) may influence semantic extraction and the generated validation artifacts. To mitigate these effects, all experiments adopted deterministic configurations and a standardized preprocessing pipeline. **Construct validity.** Semantic traceability relies on embedding-based similarity and threshold calibration, which may influence the classification of correspondence relationships. Although thresholds were defined through exploratory calibration and representative manual inspection, different embedding models or similarity configurations may produce different mappings. **External validity.** The evaluation was conducted on fifteen software projects developed in a realistic educational environment. Although the projects comprise operational software systems from different domains, the findings should be interpreted as exploratory evidence. Further studies in industrial environments are required to assess the generalizability of the proposed workflow.

Table 2: Detailed Results of Mapping and Traceability

Project	User Story			VIA Analysis		Text Gen.	Mapping		
	Total	Conforming	Non-Conforming	VIA	Cases		Validated	Potential	No Match
Project 01	20	20	0	10	55	20	6	14	1
Project 02	25	23	2	22	121	23	19	6	0
Project 03	7	0	7	5	23	0	0	0	0
Project 04	13	13	0	2	15	13	1	11	1
Project 05	11	4	7	16	78	4	0	11	2
Project 06	14	14	0	9	50	13	2	12	0
Project 07	2	0	2	13	64	0	0	0	0
Project 08	13	13	0	4	29	13	5	8	0
Project 09	23	22	1	17	102	21	5	17	3
Project 10	28	28	0	5	24	28	2	26	0
Project 11	94	0	94	14	98	0	0	0	0
Project 12	51	51	0	10	53	51	8	43	0
Project 13	5	5	0	3	16	5	0	5	0
Project 14	4	4	0	3	14	4	2	2	0
Project 15	8	8	0	7	37	8	4	4	0
TOTAL:	**318**	**205**	**113**	**140**	**779**	**203**	**54**	**159**	**07**

The findings should be interpreted as exploratory evidence regarding the feasibility and potential of multimodal semantic traceability approaches. Additional evaluations in industrial environments remain necessary to further investigate the applicability and robustness of VIA-aware traceability and V&V workflows.

7 Conclusion and Future Work

This study investigated multimodal semantic traceability by combining textual requirements analysis with interface-aware validation through Visual Interface Artifacts (VIA). The results suggest that textual and visual artifacts provide complementary semantic perspectives for Requirements Engineering (RE) and Verification and Validation (V&V) activities. While textual requirements preserve business-oriented intent, VIA frequently expose interaction constraints, validation behaviors, navigation flows, and operational semantics that are absent or only partially represented in written specifications.

The findings indicate that approaches relying exclusively on textual artifacts may overlook interface-level validation scenarios embedded in implemented systems. In several evaluated projects, VIA-oriented analysis generated validation artifacts even when User Stories (US) could not support automated test derivation due to structural inconsistencies or insufficient semantic information. This behavior provides evidence that software interfaces should not be interpreted solely as operational GUI elements, but also as semantic artifacts capable of complementing textual specifications in traceability and V&V activities.

The multimodal traceability analysis also revealed semantic gaps between business-oriented requirements and implemented interfaces, including undocumented functionalities, orphan interface elements, and vocabulary divergence between requirements and operational behaviors. Examples observed during the evaluation indicate that VIA-oriented analysis may expose interface elements and operational actions without explicit correspondence in textual requirements, reinforcing the existence of semantic gaps between documented specifications and implemented systems. These findings suggest that integrating textual and visual analysis may contribute to expanding validation coverage and improving semantic consistency assessment in software projects.

Although the study was designed as an exploratory feasibility investigation rather than a controlled effectiveness experiment, the results provide initial evidence regarding the potential of VIA-aware traceability workflows in realistic software engineering scenarios. More broadly, the study suggests that functional semantics in software projects are frequently distributed across heterogeneous artifacts, requiring multimodal approaches capable of integrating textual and interface-level perspectives during RE and V&V activities.

The exploratory evaluation also revealed opportunities to improve the workflow, particularly regarding prompt refinement, the integration of textual and visual reasoning, and the investigation of semantic mismatches between requirements and implemented interfaces.

As future work, the proposed workflow should be evaluated in industrial environments involving larger-scale software ecosystems and heterogeneous interface paradigms. Future investigations may also explore multimodal embedding models, unified reasoning strategies integrating textual and visual artifacts into a single inference process, and interface-aware approaches for automated inconsistency detection, requirements evolution analysis, and semantic coverage assessment.

Artifact Availability

Research artifacts are available at the following repositories: Figshare and GitHub.

Acknowledgments

We thank all the participants in the empirical study and the SPHERE Research Group members for their support. We thank the financial support granted by CNPq - 313137/2025-0; CNPq 406506/2025-6; CAPES- Financing Code 001; Amazonas State Research Support Foundation - FAPEAM - through POSGRAD 2026-2027. We also acknowledge the use of Generative AI tools for text revision and language support. All content was reviewed and validated by the authors.

References

[1] Abdulhady Abas Abdullah, Srwa Hasan Abdulla, Dalia Mohammad Toufiq, Halgurd S. Maghdid, Tarik A. Rashid, Pakshan F. Farho, Shadan Sh. Sabr, Akar H. Taher, Darya S. Hamad, Hadi Veisi, and Aras T. Asaad. 2024. NER-RoBERTa: Fine-Tuning RoBERTa for Named Entity Recognition (NER) within Low-Resource Languages. arXiv:2412.15252 [cs.CL] https://arxiv.org/abs/2412.15252

[2] Giulio Antoniol, Gerardo Canfora, Gerardo Casazza, Andrea De Lucia, and Ettore Merlo. 2025. Recovering Traceability Links Between Code and Documentation: A Retrospective. *IEEE Transactions on Software Engineering* 51, 3 (2025), 825–832. doi:10.1109/TSE.2025.3534027

[3] Woubshet Behutiye, Pilar Rodríguez, Markku Oivo, Sanja Aaramaa, Jari Partanen, and Antonin Abhervé. 2022. Towards optimal quality requirement documentation in agile software development: A multiple case study. *Journal of Systems and Software* 183 (2022), 111112. doi:10.1016/j.jss.2021.111112

[4] Markus Borg, Per Runeson, and Anders Ardö. 2014. Recovering from a Decade: A Systematic Mapping of Information Retrieval Approaches to Software Traceability. *Empirical Software Engineering* 19, 6 (2014), 1565–1616. doi:10.1007/s10664-013-9255-y

[5] Wontae Choi, Koushik Sen, George Necula, and Wenyu Wang. 2018. DetReduce: minimizing Android GUI test suites for regression testing. In *Proceedings of the 40th International Conference on Software Engineering*. Association for Computing Machinery, New York, NY, USA, 445–455. doi:10.1145/3180155.3180173

[6] Souvick Das, Novarun Deb, Agostino Cortesi, and Nabendu Chaki. 2024. Extracting goal models from natural language requirement specifications. *Journal of Systems and Software* 211 (2024), 111981. doi:10.1016/j.jss.2024.111981

[7] Shaik Arsh Hussain, Ravishta Kohli, Saniya Zahoor, and Shabir A. Sofi. 2025. Transforming the GUI Landscape: Harnessing the Power of MPNet base v2 Sentence Transformers. *Procedia Computer Science* 259 (2025), 1809–1816. doi:10.1016/j.procs.2025.04.136 Sixth International Conference on Futuristic Trends in Networks and Computing Technologies (FTNCT06), held in Uttarakhand, India.

[8] Aftab Alam Janisar, Khairul Shafee bin Kalid, Aliza Bt Sarlan, and Umar Danjuma Maiwada. 2024. Software Development Teams Knowledge and Awareness of Security Requirement Engineering and Security Requirement Elicitation and Analysis. *Procedia Computer Science* 234 (2024), 1348–1355. doi:10.1016/j.procs.2024.03.133 Seventh Information Systems International Conference (ISICO 2023).

[9] Zubair Khaliq, Sheikh Umar Farooq, and Dawood Ashraf Khan. 2022. A deep learning-based automated framework for functional User Interface testing. *Information and Software Technology* 150 (2022), 106969. doi:10.1016/j.infsof.2022.106969

[10] Wahab Khan, Ali Daud, Khairullah Khan, Shakoor Muhammad, and Rafiul Haq. 2023. Exploring the frontiers of deep learning and natural language processing: A comprehensive overview of key challenges and emerging trends. *Natural Language Processing Journal* 4 (2023), 100026. doi:10.1016/j.nlp.2023.100026

[11] Yves R. Kirschner, Moritz Gstür, Timur Sağlam, Sebastian Weber, and Anne Koziolek. 2025. Retriever: A view-ba sed approach to reverse engineering software architecture models. *Journal of Systems and Software* 220 (2025), 112277. doi:10.1016/j.jss.2024.112277

[12] Kristian Kolthoff, Felix Kretzer, Christian Bartelt, Alexander Maedche, and Simone Paolo Ponzetto. 2024. Interlinking User Stories and GUI Prototyping: A Semi-Automatic LLM-Based Approach. In *2024 IEEE 32nd International Requirements Engineering Conference (RE)*. 380–388. doi:10.1109/RE59067.2024.00045

[13] Janosch Luttmer, Vitalijs Prihodko, Dominik Ehring, and Arun Nagarajah. 2023. Requirements extraction from engineering standards – systematic evaluation of extraction techniques. *Procedia CIRP* 119 (2023), 794–799. doi:10.1016/j.procir.2023.03.125 The 33rd CIRP Design Conference.

[14] Ahmad Mustafa, Wan M. N. Wan-Kadir, Noraini Ibrahim, Muhammad Arif Shah, Muhammad Younas, Atif Khan, Mahdi Zareei, and Faisal Alanazi. 2021. Automated Test Case Generation from Requirements: A Systematic Literature Review. *Computers, Materials and Continua* 67, 2 (2021), 1819–1833. doi:10.32604/cmc.2021.014391

[15] Liming Nie, Kabir Sulaiman Said, Lingfei Ma, Yaowen Zheng, and Yangyang Zhao. 2023. A systematic mapping study for graphical user interface testing on mobile apps. *IET Software* 17, 3 (2023), 249–267. doi:10.1049/sfw2.12123

[16] Ana Carolina Oran, Gleison Santos, Bruno Gadelha, and Tayana Conte. 2021. A Framework for Evaluating and Improving Requirements Specifications Based on the Developers and Testers Perspective. *Requirements Engineering* 26, 4 (2021), 481–508. doi:10.1007/s00766-021-00352-6

[17] Jainik Sudhanshubhai Patel. 2025. AI-driven test automation: Transforming software quality engineering. *Journal of Computer Science and Technology Studies* 7, 2 (2025), 339–347. doi:10.32996/jcsts.2025.7.2.35

[18] Claes Wohlin, Per Runeson, Martin Höst, Magnus C. Ohlsson, Björn Regnell, and Anders Wesslén. 2012. *Experimentation in Software Engineering* (1 ed.). Springer, Berlin, Heidelberg. doi:10.1007/978-3-642-29044-2

[19] Elise Wolf, Adam Trendowicz, and Julien Siebert. 2026. Quality assessment of software requirements using artificial intelligence methods: A systematic literature review. *Information and Software Technology* 191 (2026), 107979. doi:10.1016/j.infsof.2025.107979

[20] Xiangqian Xu, Yajie Dou, Liwei Qian, Jiang Jiang, Kewei Yang, and Yuejin Tan. 2023. Quality improvement method for high-end equipment's functional requirements based on user stories. *Advanced Engineering Informatics* 56 (2023), 102017. doi:10.1016/j.aei.2023.102017

[21] Liping Zhao, Waad Alhoshan, Alessio Ferrari, and Keletso J. Letsholo. 2022. Classification of Natural Language Processing Techniques for Requirements Engineering. arXiv:2204.04282 [cs.CL] https://arxiv.org/abs/2204.04282