GraphRAG-Driven Compliance Validation of Software Requirements: Semantic, Content, and Data Compliance Metrics

Matas Čenys, Asta Slotkienė

Vilnius University, Faculty of Mathematics and Informatics,
Universiteto g. 3, Vilnius, Lithuania
matas.cenys@mif.stud.vu.lt

Abstract. This paper proposes a graphRAG-driven method for validating the semantic, content, and data compliance of software requirements against organisational and technical reference documentation, using a graph-based retrieval-augmented generation system paired with a large language model. The experimental evaluation demonstrated that the proposed method achieved high semantic compliance accuracy in its baseline graphRAG configuration, while adjusting graph database parameters and prompting techniques yielded improvements in requirements compliance accuracy.

Keywords: graphRAG, requirement compliance, knowledge retrieval.

1 Introduction

Requirements engineering and the quality of these requirements are an important part of the project development process. Incorrectly defined functional and non-functional requirements can be the reason the product delivered to project stakeholders does not meet the time and quality expectations set for them, as indicated by software quality problems, wasted effort, and delivery delays that affect such projects [4]. Research shows that mistakes caused by incorrect requirements in the software code base can account for 70% to 85% of the additional costs incurred to correct them throughout the project [9]. One reason for incorrect requirements is a lack of compliance with higher-level organisational or national requirements and documents, particularly in highly regulated environments such as finance or aerospace, where maintaining consistency, adhering to regulatory frameworks, minimising errors, and meeting critical expectations are essential to a reliable system [2]. With the growing pressure for software projects to comply with laws and regulations, research proposes methods

DOI: https://doi.org/10.15388/LMITT.2026.5

involving large language models (LLMs) and retrieval-augmented generation (RAG) usage to aid in the validation task – Masoudifard et al. [2] has proposed an automated framework for validating SRS documents against higher-level regulatory requirements, with graphRAG method used for retrieval, paired with an LLM model, using advanced prompting techniques, such as Chain of Thought, while Arora, Chetan, et al. proposed a framework called CompliAT, which uses LLMs with a RAG model that handles three compliance tasks – terminology consistency, classifying products to international standards and tracing product specifications to standard requirements with the aim of significantly reducing manual compliance effort [4]. The paper proposes a method for validating requirement quality and compliance with semantic, content, and data requirements against organisational or technical documentation using a graphRAG solution paired with an LLM model.

The remainder of this paper is structured as follows. Section 2 presents the proposed graphRAG compliance validation method. The experiment results are presented in Section 3. Conclusions are drawn in Section 4.

2 GraphRAG-Driven Compliance Validation Method

To provide enhanced, project-specific context for LLMs, the Retrieval Augmented Generation (RAG) technique, which connects LLMs to an external knowledge base at inference time, has come into prominence. One such technique is graphRAG, which extends the RAG system by converting supporting documents into a graph [3]. One type of graphRAG is the lexical graph with extracted entities, which splits documents into text chunks, extracts entities from those chunks, and connects them into a graph [9] (Figure 1). Each chunk is connected to a document through PART_OF_DOCUMENT relationship and with NEXT_CHUNK relationship to chunks that go after it. If an entity was extracted from the chunk, this entity is connected to it using the HAS_ENTITY relationship.

The method proposed in this paper utilises a graphRAG system to aid the LLM model by validating requirement compliance. The process can be divided into two parts: database construction and requirement compliance assessment (Figure 2, steps 1 and 2).

Database construction is performed by inspecting the reference documentation (Figure 2, step 1.1), which supports the system whose requirements should be validated. These documents are then divided into text chunks (Figure 2, step 1.2), forming a graph structure. Once that is

performed, entities are extracted from the text chunks and connected in the RAG database as relationships between the entity and all text chunks that reference it (Figure 2, step 1.3).

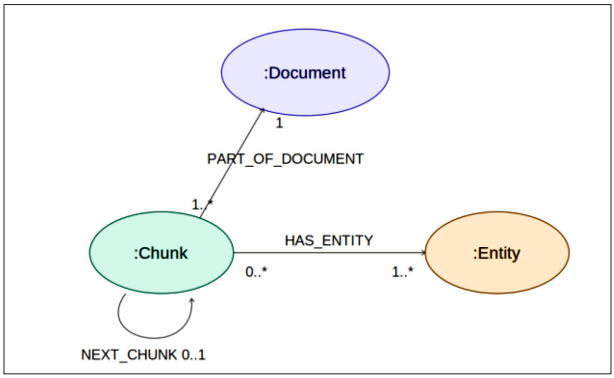

Figure 1. Graph schema of a lexical graph with extracted entities.

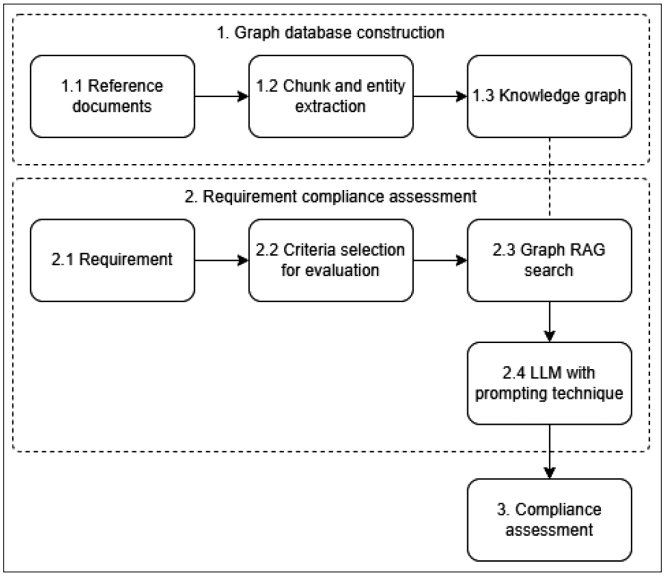

Figure 2. Principal schema of the method.

Table 1. The configurable parameter of the database construction step.

Configurable parameter	Description	Base-line value	Available values
Chunk size	The maximum number of characters per text chunk. If adding a sentence would exceed this limit, it is excluded from the chunk.	1000	300, 400, 500, 600, 700, 800, 900, 1000 characters
Overlap	The number of characters taken from the end of the previous chunk and added to the next one.	200	100, 150, 200 characters

For determining the optimal configuration of the database (Table 1) for the compliance task, the method allows for modifying two metrics:

1. Chunk size determines the size of a single text chunk in the database.
2. Overlap size, which determines how many characters overlap in two text chunks that go one after another.

During the requirement compliance assessment step (Figure 2, step 2), a requirement is first selected for assessment (Figure 2, step 2.1). Afterwards, the compliance criteria that the LLM will be validating the requirement against are defined (Figure 2, step 2.2). The method provides the ability to validate a requirement against one of three compliance criteria:

1. Semantic compliance validates whether a requirement describes a single need by inspecting its semantic structure. This criterion is more heavily dependent on the LLM part of the proposed method, due to its ability to assess semantic-level requirement properties, such as consistency, contradiction and unambiguousness [1].
2. Content compliance validates whether the requirement is supported by the context documentation by extracting documentation from the GraphRAG, which is connected to the requirement [6]. This criterion ties to completeness and traceability of the requirement.
3. Data compliance validates that the requirement clearly states all data constraints, as documented in the supporting documentation. For example, a statement about permissions that must be set on a generated file also specifies the exact permissions that should be set on the file, as documented in the context. Such validation ensures the correctness and verifiability of requirements by strictly enforcing the data values described in the context documentation [8].

With the requirement and compliance criteria selected, the validation process (Figure 2) begins by passing the requirement text as search criteria for the GraphRAG database (Figure 2, step 2.3), which begins a hybrid search:

1. Firstly, performing a vector search in the database for all chunks and entities that would be the closest match to the provided requirement.
2. Secondly, performing a direct graph traversal to enhance the selected context.

Once context is collected from the database, it is passed to the LLM model, along with a prompting technique (Figure 2, step 2.4), to generate a compliance assessment score for the requirement in question (Figure 2, step 3). During this process, additional requirements can be configured that have a direct impact on the graph database search strategy, such as search depth and top k context chunks, or LLM performance in the validation task via the configured prompting technique (Table 2).

Table 2. The configurable parameters for the requirement compliance assessment.

Configurable parameter	Description	Base-line value	Available values
top_k	Indicates the amount of context chunks that will be returned by the tool. If top_k parameter exceeds the amount of context found, only the relevant number of found context will be returned.	5	3, 4, 5, 6, 7, 8, 10
Depth	Defines the depth of relationships that the tool will explore when traversing graph database to collect relevant context.	1	1, 2, 3, 4
Prompt technique	Defines the instructions provided to the LLM model that control how LLM will decide on the final answer and how the answer will be structured.	Tree of Thought	Few shot, zero shot, Chain of Thought, Tree of Thought

To validate the requirement against all three compliance criteria, the process is repeated once per criterion, with database construction skipped when context documents and parameters remain unchanged.

Once the LLM generates an assessment score, additional quality metrics are calculated alongside the results – including total database construction and evaluation times in seconds as an efficiency indicator.

To determine the quality of the answers provided by the method, a set of metrics has been selected that focuses primarily on the graphRAG database performance:

1. Similarity Score is a number computed by, for each reasoning sentence (r_i) extracted from the LLM's response, finding the maximum cosine similarity with all retrieved context chunks (c_j), then averaging these maximum values across all N reasoning sentences. It reflects how well the method's reasoning is grounded in the retrieved context (Equation 1).

$$Similarity\ score = \frac{1}{N} \times \sum_{i=1}^{N} \max_{j} \frac{r_i \times c\_j}{|r_i| \times |c_j|} \quad (1)$$

2. Faithfulness is a number gained by dividing the number of atomic statements extracted from the answer that are verified as supported by the context (V) from the total number of such statements (S) [10] (Equation 2).

$$Faithfulness = \frac{V}{S} \quad (2)$$

3. Retrieval Quality is a number gained by computing the cosine similarity between the requirement embedding (q) and each retrieved context chunk embedding (c_j), then averaging across all retrieved chunks (M). It reflects how relevant the retrieved context is to the requirement being evaluated, independently of the LLM's response. [10] (Equation 3).

$$Retrieval\ Quality = \frac{1}{M} \times \sum_{j=1}^{M} \frac{q \cdot c_j}{|q| \cdot |c_j|} \quad (3)$$

The quality metrics can then be used to determine the optimal setup of the graphRAG database configuration and the requirement compliance assessment process to efficiently generate accurate requirement validation results.

3 Results

Each experiment iteration evaluated a set of 31 requirements, together with 5 supporting documents, against all three compliance criteria. In each iteration, the validation process was repeated twice to ensure consistent results. For each of the previously mentioned quality and efficiency metrics, calculations and collections have been completed, together with the method

accuracy metric. Method accuracy (Equation 4) is the ratio between correctly scored requirement criteria and scores provided by a human specialist, who had access to the same information as the method:

$$Assessment\ accuracy = \frac{model\ requirement\ scores\ for\ criteria}{human\ specialist\ requirement\ scores\ for\ criteria} \times 100\% \quad (4)$$

To better understand the accuracy metric, false-positive and false-negative evaluations have also been calculated and reported after every iteration.

Firstly, the baseline iteration was performed, demonstrating that the method correctly validated the semantic compliance of requirements, achieving 93.5% accuracy; however, it required improvement in data and content compliance, achieving only 61.3% and 41.9% accuracy, respectively (Table 3). Furthermore, database setup in this case took almost 94 seconds, while evaluation took even longer – 246 seconds.

Table 3. Baseline iteration results summary.

Criterion	Correct assessments	Setup time (s)	Evaluation time (s)
Semantic compliance	93.5%	93.46	246.2
Data compliance	61.3%		
Content compliance	41.9%		

Afterwards, all database setup affecting iterations were carried out. While neither of the iterations managed showcase better than baseline accuracy results, and in some case even reduced it, the similarity score, faithfulness and retrieval quality metrics were visibly affected by the change (Figure 3). The chunk size iterations had the greatest positive effect – smaller chunks improve both retrieval quality and faithfulness, suggesting more focused context and better answer grounding. However, this comes at a cost: each reduction in chunk size increases database setup time, peaking at 427 seconds with a chunk size of 300 characters, while evaluation time remains comparable to baseline.

The similarity score improved most when overlap was reduced, indicating reduced graph noise; however, retrieval quality and accuracy metrics suggest this comes at the cost of overall method performance.

Although the top-k and depth parameters do not affect the database setup, they were analysed alongside other parameters due to their

relationship with the graphRAG database (Figure 4). Retrieval quality remains largely unchanged across values, except at increased search depth, where it drops – indicating the model retrieved irrelevant context rather than enhancing it. A depth of 2 and a top-k of 6 yield the best results, improving accuracy, similarity score, and faithfulness while maintaining retrieval quality.

		Sim.	Faith.	Retr.	Acc. (%)
Chunk Size	300	0.370	0.180	0.313	64.5%
	400	0.373	0.185	0.316	63.4%
	500	0.367	0.159	0.315	64.5%
	600	0.366	0.180	0.308	65.6%
	700	0.367	0.170	0.311	63.4%
	800	0.372	0.167	0.290	62.4%
	900	0.362	0.146	0.293	62.4%
	1000*	0.367 *	0.152 *	0.277 *	65.6% *
Overlap	100	0.379	0.174	0.274	65.6%
	150	0.378	0.170	0.276	65.6%
	200*	0.367 *	0.152 *	0.277 *	65.6% *

* = baseline | colour scale is independent per metric column

Figure 3. Chunk and overlap size parameters results.

		Sim.	Faith.	Retr.	Acc. (%)
Top-K	3	0.355	0.134	0.288	65.6%
	4	0.366	0.153	0.276	64.5%
	5*	0.367 *	0.152 *	0.277 *	65.6% *
	6	0.379	0.185	0.278	67.7%
	7	0.377	0.171	0.277	66.7%
Depth	1*	0.367 *	0.152 *	0.277 *	65.6% *
	2	0.385	0.176	0.268	68.8%
	3	0.383	0.175	0.262	62.4%
	4	0.389	0.186	0.258	65.6%

* = baseline | colour scale is independent per metric column

Figure 4. Top-k and search depth experiment results.

The usage of different prompting techniques does not show any impact to the semantic compliance accuracy due to already well defined rules for detecting unambiguousness in a requirement through connecting words. Data accuracy does not show improvements either, with all prompting techniques leaning the model towards false negative answers, apart from the

few shot algorithm, that showcases a slight improvement when compared to other methods. Content compliance shows varying results, with zero shot algorithm reaching best results by reducing method strictness and converting part of the false negative answers into true positives (Table 4).

Table 4. Prompting technique experiment iteration results.

Value	Evaluation time (s)	Semantic accuracy	Data accuracy	Content accuracy
Tree-of-Thought (baseline)	246.2	93.5%	61.3%	41.9%
Zero shot	275.4	93.5%	61.3%	58.1%
Chain-of-Thought	250.7	93.5%	61.3%	41.9%
Few shot	293.5	93.5%	64.5%	54.8%
Chain-of-Thought few shot	343.9	93.5%	61.3%	38.7%

4 Conclusions

The paper proposed a method for validating the semantic, data, and content compliance of the provided requirements against their supporting documents.

The method can achieve 93.5% semantic, 61.3% data, and 41.9% content compliance assessment accuracy in its baseline configuration with graphRAG, with additional accuracy improvements possible by adjusting the top-k and search depth parameters. While chunk and overlap sizes did not improve the overall accuracy of the model, but gave model more focused context leading better answer grounding and improvement to overall quality metrics, like faithfulness and retrieval quality. It is also important to note that while reduction in chunk size did not increase the method performance, it did allow method to get more relevant context as indicative by the improvements to retrieval quality metric. Lastly, the top-k and search depth parameters showed noticeable improvements in the similarity score and faithfulness metrics. It is important to note that excessive improvements to the search depth metric led to worse accuracy and retrieval quality, due to missing relevant context when going too deep into the graphRAG database.

Lastly, the use of prompting techniques can also affect the method's results; however, in the performed experiment iterations, improvements came only from using less advanced prompting techniques, such as few-

shot and zero-shot, which reduced the method's strictness, thereby lowering the false negative rate and allowing it to correctly classify the requirement.

References

[1] Alessandro Fantechi, Stefania Gnesi, Lucia Passaro, Laura Semini. *Inconsistency Detection in Natural Language Requirements using ChatGPT: A Preliminary Evaluation.* IEEE, 2023.

[2] Arsalan Masoudifard, Mohammad Mowlavi Sorond, Moein Madadi, Mohammad Sabokrou, Elahe Habibi. „Leveraging Graph-RAG and Prompt Engineering to Enhance LLM-Based Automated Requirement Traceability and Compliance Checks." 2024.

[3] Boci Peng, Yun Zhu, Yongchao Liu, Xiaohe Bo, Haizhou Shi, Chuntao Hong, Yan Zhang, Siliang Tang. „Graph Retrieval-Augmented Generation: A Survey." 2024.

[4] Chetan Arora, John Grundy, Louise Puli, Natasha Layton. *Towards Standards-Compliant Assistive Technology Product Specifications via LLMs.* Cornell University, 2024.

[5] Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, Jonathan Larson. *From Local to Global: A Graph RAG Approach to Query-Focused Summarization.* Cornell University, 2024.

[6] Elizabeth Bjarnason, Per Runeson, Markus Borg, Michael Unterkalmsteiner, Emelie Engström, Björn Regnell, Giedre Sabaliauskaite, Annabella Loconsole, Tony Gorschek, Robert Feldt. „Challenges and Practices in Aligning Requirements with Verification and Validation: A Case Study of Six Companies." 2023.

[7] Group, The Standish. *CHAOS Report Beyond Infinity (digital version).* Boston, 2020.

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

[9] Orlando Amaral Cejas, Muhammad Ilyas Azeem, Sallam Abualhaija, Lionel C. Briand. *NLP-Based Automated Compliance Checking of Data Processing Agreements Against GDPR.* IEEE, 2023.

[10] Shahul Es, Jithin James, Luis Espinosa-Anke, Steven Schockaert. „Ragas: Automated Evaluation of Retrieval Augmented Generation." 2025.