

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Enhancing requirements quality through automated ambiguity detection: A Comparative Analysis of Manual, Rule-Based and Generative Artificial Intelligence Techniques

Allan Ramos



Master in Software Engineering

Supervisor: Rosaldo Rossetti

June 26, 2025

Enhancing requirements quality through automated ambiguity detection: A Comparative Analysis of Manual, Rule-Based and Generative Artificial Intelligence Techniques

Allan Ramos

Master in Software Engineering

June 26, 2025

Abstract

Ambiguity in requirements engineering specifications is a longstanding challenge in software engineering, often contributing to costly misunderstandings, delays, and system failures. This thesis investigates the potential of Generative Artificial Intelligence (Gen-AI), specifically Large Language Models (LLMs), to enhance the requirements review process by improving ambiguity detection and resolution.

Traditional approaches to requirements review—such as manual inspections, rule-based analyses, and template-driven techniques—have played a crucial role in managing ambiguity. However, these methods often struggle with issues of scalability, semantic interpretation, and contextual sensitivity. This study conducts a comparative analysis of traditional review practices and GenAI-enabled approaches, assessing their respective strengths and limitations in the context of ambiguity detection.

The research develops a structured evaluation framework that highlights where Gen-AI models can offer meaningful improvements, particularly in understanding context, capturing nuanced semantics, and adapting to varied domains. Based on this analysis, a hybrid review strategy is proposed—one that combines the procedural reliability of traditional methods with the advanced analytical capabilities of Gen-AI.

Implementation considerations are also addressed, including challenges related to model transparency, organizational trust, and integration with existing workflows. The findings suggest that, when applied responsibly, Gen-AI can significantly improve the effectiveness and efficiency of requirements review activities.

This thesis contributes to the field of AI-assisted requirements engineering by offering a prototype implementation of a project called ReqVision that is an Agent and prompt builder to Large Language Models evaluating and adopting pre-trained model on a pilot project of an Infusion Pump in the Medical domain. It aims to improve the Requirement engineering process review on safe critical systems.

Acknowledgments

First, I would like to express my deepest gratitude to Professor Rosaldo Rossetti for his invaluable guidance, support, and encouragement throughout my academic course at FEUP. His expertise, insightful feedback, and patience have been essential to the completion of this research.

I am also profoundly grateful to my wife Ana Paula, daughter Catarina, mom Ana Maria and brothers Aslan and Allane and to all my family for being the cornerstone of my life and all the support along this journey.

Allan Ramos

*“ With your mind power, your determination,
your instinct, and the experience as well, you can fly very high.”*

Ayrton Senna

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Document Structure	2
1.3	Expected Contributions	3
2	Background	4
2.1	Critical Safety Systems	4
2.2	Requirement engineering overview	6
2.2.1	Requirements Engineering Process	6
2.2.2	Ambiguities in Requirements	7
2.2.3	Limitations of Traditional Methods	9
2.3	Introduction to Generative Artificial Intelligence	9
2.3.1	Natural language processing	10
2.3.2	Generative Artificial Intelligence	10
3	Related Work	11
4	Methodology	14
4.1	Problem description	14
4.2	Research questions	15
5	Description of the Work	16
5.1	Theoretical Foundation	16
5.1.1	Ambiguity in Natural Language	16
5.1.2	Prompt Engineering	17
5.2	Implementation	17
5.2.1	Confluence Project: ReqVision AI	18
5.2.2	ChatGPT-4o Integration	18
5.2.3	Backend Agent Workflow	19
6	Results and Evaluation	21
6.1	Traditional Requirements Engineering Tools and Processes	21
6.2	Comparative Evaluation	23
6.2.1	Confusion Matrix Analysis	23
7	Discussion	25

8	Conclusions	27
8.1	Main Contributions	27
8.2	Future Work	28
	References	29

List of Figures

2.1	Boeing 737 MAX	5
2.2	Requirements engineering Process	7
5.1	ReqVision AI Project in Confluence	18
5.2	Example Output from ReqVision Reviewing a Requirement	19
5.3	flow chart	20
6.1	Confusion Matrix Comparison across Review Methods (Estimated). The true negative values are not shown due to unavailable data on total negative instances.	24

List of Tables

6.1 Detailed Comparison of Requirements Review Approaches	24
---	----

Abbreviations and Symbols

RE	Requirement Engineering
AI	Artificial Intelligence
GenAI	Generative Artificial Intelligence
NLP	Natural Language Processing
LLM	Large Language Model
DL	Deep Learning
ANN	Artificial Neural Network
SRS	Software Requirement Specification
ANNs	Artificial Neural Networks
MLP	Multi-Layer Perceptron
CNNs	Convolutional Neural Networks
DNN	Deep Neural Networks
NL	Natural Language
UI	User Interface
CSV	Comma-Separated Values
RAD	Requirement Ambiguity Detection
CIA	Correctly Identified Ambiguities
ADA	Additional AI Detected Ambiguities
AND	Actual Ambiguities Not Detected by AI

Chapter 1

Introduction

1.1 Motivation

Requirements Engineering (RE) is a foundational discipline within the software development life-cycle. It involves a structured process of identifying, analyzing, documenting, validating, and managing the requirements of a software system to ensure that it fulfills the expectations of users and stakeholders [Pohl \(1996\)](#). As a critical early-phase activity, RE significantly influences the success or failure of software projects [Nuseibeh and Easterbrook \(2000\)](#).

A considerable number of industrial software projects experience delays, exceed budgets, or fail to meet intended objectives. These failures are frequently attributed to shortcomings in the requirements phase, including ambiguities, incompleteness, contradictions, poor documentation, and requirements that are not aligned with stakeholder needs [Bokhari and Siddiqui \(2011\)](#). Because RE serves as the groundwork upon which subsequent development is built, any flaws at this stage can propagate through the entire project life-cycle, compounding risks and inefficiencies [Pandey et al. \(2010\)](#).

Requirements typically define system behavior and functionalities and are most commonly expressed in natural language [Berezin \(1999\)](#). However, natural language is inherently prone to ambiguity and inconsistency, which can lead to multiple interpretations, stakeholder misunderstandings, project delays, and defects in downstream activities. In the face of rapidly evolving technologies and the growing complexity of systems, there is an increasing need for more robust and intelligent methods to support RE tasks [Rani and Aggarwal \(2018a\)](#).

One emerging approach is the use of Generative Artificial Intelligence (Gen-AI), a subset of AI focused on generating content such as text, images, or code [Arora et al. \(2023\)](#). By leveraging large-scale datasets and machine learning techniques, particularly through Large Language Models (LLMs), Gen-AI offers the potential to enhance several RE tasks—especially those related to ambiguity detection and resolution. These models can learn from existing requirement documents, infer context, and produce new or refined requirements that better reflect stakeholder intentions [Ronanki et al. \(2023\)](#).

Natural Language Processing (NLP) capabilities within Gen-AI systems allow them to interpret complex sentence structures, resolve ambiguous statements, and identify contradictions across multiple sources. This capability supports the creation of a shared understanding among stakeholders and minimizes the risk of misinterpretation.

This thesis explores how Gen-AI can be applied to the RE process with a particular focus on detecting ambiguous requirements and suggesting improvements through an Agent tool that incorporates a dynamic review using an integration with a GPT from Open-AI trained with domain-specific documents that suggest and detect ambiguity in the requirements with the domain-specific documentation.

The tool offers several advantages: it supports early and accurate requirements elicitation, reduces the time spent on manual validation, and enables automation of repetitive tasks such as traceability analysis. Additionally, its adaptive learning capability allows the system to improve based on user feedback, keeping requirements relevant to evolving project contexts. The underlying technique used to engage with Gen-AI models is called prompting. Prompt engineering is the practice of designing effective inputs (prompts) to elicit accurate and useful outputs from language models [Oppenlaender et al. \(2023\)](#). This thesis proposes prompt patterns tailored for requirements analysis tasks, incorporating contextual information, task descriptions, and input/output indicators.

1.2 Document Structure

This thesis is structured into seven chapters, each contributing to a comprehensive understanding of the research objectives, background, methodology, and findings related to ambiguity detection in medical device requirements using Generative AI.

- **Chapter 1 – Introduction:** Presents the motivation for the research, outlines the challenges of ambiguity in requirements engineering (RE), especially in FDA-regulated medical device contexts, and states the research questions and expected contributions.
- **Chapter 2 – Background:** Provides an overview of critical safety systems with a focus on aviation and medical domains, details the RE process, types of ambiguities in natural language requirements, and introduces Natural Language Processing (NLP) and Generative AI as enablers for enhanced RE.
- **Chapter 3 – Related Work:** Reviews the current state of research on ambiguity detection in RE, covering traditional, rule-based, and AI-driven approaches. It identifies existing research gaps and motivates the need for the approach proposed in this thesis.
- **Chapter 4 – Method:** Defines the problem addressed in this study, outlines the research questions, and describes the methodological approach, including the integration of a Gen-AI tool within a Confluence environment tailored to medical device documentation.

- **Chapter 5 – Tool Design and Implementation:** Describes the architecture, design, and technical implementation of the proposed tool. It details how GPT-based models are employed through prompt engineering and how the system supports ambiguity detection and resolution.
- **Chapter 6 – Evaluation:** Evaluates the effectiveness of the proposed tool based on real-world requirements. The chapter examines its accuracy in ambiguity detection, the quality of suggested rewrites, and usability feedback from medical device engineers.
- **Chapter 7 – Conclusion and Future Work:** Summarizes the main findings, reiterates the thesis contributions, and discusses limitations. It also outlines future research directions for improving Gen-AI applications in RE within safety-critical domains.

1.3 Expected Contributions

The primary contributions of this thesis are:

1. **Development of a Gen-AI based Tool:** A prototype tool is developed that integrates Gen-AI capabilities to analyze requirement specifications. The tool identifies ambiguous statements and offers suggestions for refinement through a user-friendly interface. This aims to support requirement engineers by enhancing both the efficiency and quality of the requirements review process.

Chapter 2

Background

2.1 Critical Safety Systems

A critical safety system is an embedded system whose failure could result in injury, loss of life, or significant damage [Staff \(2005\)](#). Aviation, medical devices, automotive systems, and railway operations are all examples of domains that rely on such systems, where failure can lead to catastrophic consequences [Knight \(2002\)](#). Embedded systems are deeply integrated into modern life, from everyday consumer electronics to advanced aerospace technologies [Koopman \(2007a\)](#). These systems combine hardware and software to perform vital functions with high efficiency, reliability, and safety [Koopman \(2007b\)](#).

In safety-critical domains, embedded systems are essential for safeguarding human lives and infrastructure. Their development is governed by stringent standards to ensure reliable and fail-safe operation. Whether used in aircraft cockpits or robotic surgery equipment, these systems require meticulous design, testing, and certification. Thus, safety in embedded systems is an essential concern, regulated by industry-specific guidelines and best practices [Koopman \(2007b\)](#).

This thesis focuses on aviation systems due to their critical safety demands and the high-risk environments in which they operate. In aviation, safety is strictly regulated by standards such as DO-178C for software and ARP4754A for system-level design. Consider, for example, the flight control system of a commercial airliner flying at 35,000 feet. A malfunction in the autopilot or engine control software could lead to the loss of control, endangering hundreds of lives on board.

There are several well-known examples of safety-critical system failures that underscore the importance of rigorous development.

- **Air France Flight 447 crash (2009):** The aircraft’s automation systems delivered conflicting sensor data, which confused the pilots and contributed to the loss of control. The 228 people on board died [ref \(2023\)](#).
- **Therac-25 incident:** In the 1980s, software faults in the Therac-25 radiation therapy machine led to several massive radiation overdoses, causing severe injuries and deaths of at least four patients [ref \(2022\)](#).

- **Boeing 737 MAX crashes:** Two major crashes—Lion Air Flight 610 (2018) and Ethiopian Airlines Flight 302 (2019), were linked to the MCAS system, which responded to faulty sensor data forcing the aircraft to dive. These accidents resulted in 346 deaths and grounded the aircraft model worldwide [ref \(2014\)](#).



Figure 2.1: Boeing 737 MAX
[ref \(2014\)](#)

These tragedies highlight the vital role of safety protocols and system verification in development. They have also led to significant reforms in certification procedures, design practices, and safety oversight throughout the aviation industry.

Although technological innovations continue to improve predictive maintenance and fault tolerance, adherence to strict protocols remains non-negotiable. Requirement engineering plays an essential role in this ecosystem by offering a structured methodology to define, verify, and maintain system requirements, ensuring that safety concerns are addressed from the beginning [Martins and Gorschek \(2020\)](#).

2.2 Requirement engineering overview

Requirements Engineering (RE) is recognized as a structured methodology focused on the definition, documentation, and management of software and system requirements throughout the engineering lifecycle, as emphasized by Martins and Gorschek [Martins and Gorschek \(2020\)](#). Another perspective, offered by Nuseibeh and Easterbrook [Nuseibeh and Easterbrook \(2000\)](#), underscores that RE forms the foundation for all other project activities; if poorly executed, it may lead to critical failures in achieving project success. In this context, project success is measured by how effectively the software meets its intended purpose—namely, fulfilling user expectations and delivering on system functionality [Nuseibeh and Easterbrook \(2000\)](#).

Before delving deeper into the RE process itself, it is useful to consider the meaning of a “requirement.” Koopman [Koopman \(2007a\)](#) defines a requirement as a pivotal component of product development—representing the expectations and demands of all involved stakeholders. This concept serves as a bridge between real-world problems and their software-based solutions. Essentially, it transforms domain-specific challenges into software specifications that can be systematically addressed.

During the RE process, requirements are discovered, analyzed, and formalized through continuous dialogue between stakeholders and developers. This phase is instrumental in enabling engineers to understand what needs to be built. However, this process is not without challenges—conflicts and ambiguities often emerge and can jeopardize project outcomes. Martins and Gorschek [Martins and Gorschek \(2020\)](#) identify several key stages in this process: requirements elicitation, analysis, negotiation, and documentation—all of which are essential for ensuring that the final product aligns with its intended goals.

2.2.1 Requirements Engineering Process

The requirements engineering (RE) process is composed of several distinct but interrelated stages, each contributing critically to the formulation and maintenance of precise and high-quality software requirements. These stages, typically illustrated as a cyclical or iterative flow (see Figure ??), include elicitation, analysis, negotiation, documentation, and validation.

- **Requirements Elicitation:** This initial phase focuses on gathering information from stakeholders and domain experts who influence the system’s design and function. As noted by Martins and Gorschek [Martins and Gorschek \(2020\)](#), elicitation ensures that stakeholders’ needs are identified early and forms the basis for subsequent activities in RE. Often referred to as “requirements discovery” [Pandey et al. \(2010\)](#), this stage sets the foundation for defining what the system should do.
- **Requirements Analysis:** Following elicitation, the captured requirements undergo thorough analysis to identify and resolve issues such as contradictions, vagueness, redundancies, or missing information [Martins and Gorschek \(2020\)](#). Effective analysis ensures internal consistency and feasibility of the specified needs.

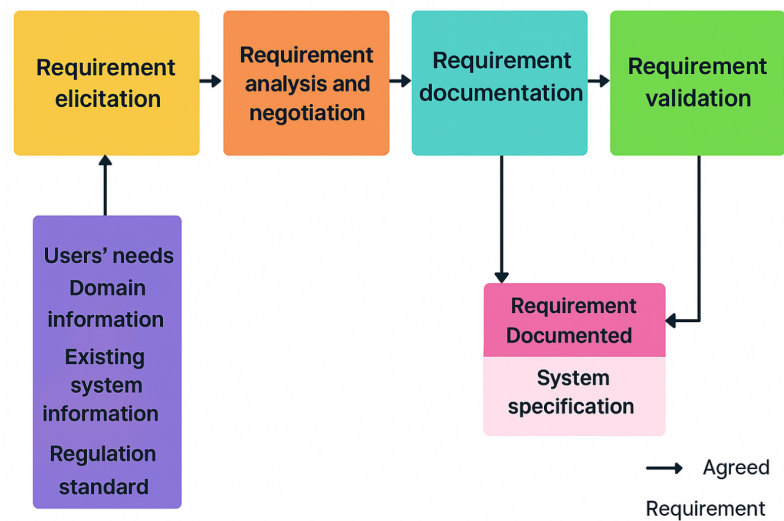


Figure 2.2: Requirements engineering Process

- **Requirements Negotiation:** Given that RE involves multiple stakeholders—each possibly holding divergent views or priorities—conflict is a common challenge. As described by Martins and Gorschek [Martins and Gorschek \(2020\)](#), negotiation is essential to align competing requirements and reach a consensus that reflects a balanced compromise.
- **Requirements Documentation:** Once an agreement is reached, it becomes essential to represent the validated requirements in a well-structured and comprehensible format. This documentation serves both as a reference for developers and a contract with stakeholders [Martins and Gorschek \(2020\)](#).
- **Requirements Validation and Verification:** Finally, the documented requirements are evaluated for correctness and completeness. Techniques such as stakeholder reviews, prototype testing, and formal inspections help ensure the accuracy and relevance of the specifications [Pandey et al. \(2010\)](#).

The importance of a requirement review is underscored by research showing that defects identified during the requirements phase cost significantly less to fix than those discovered during testing or after deployment. According to industry estimates, fixing a requirement defect after deployment can cost 100 times more than addressing it during the requirements phase.

2.2.2 Ambiguities in Requirements

Ambiguities within software requirements occur when a single statement allows for multiple plausible interpretations. This often leads to confusion or inconsistent understanding among stakeholders regarding what is actually being specified. As noted by Rani and Aggarwal [Rani and Aggarwal \(2018a\)](#), ambiguity in requirements is not merely about polysemy; it is a multifaceted issue that can manifest at various linguistic and contextual levels.

The flexible and interpretive nature of natural language contributes significantly to this challenge. Vague wording, insufficient context, and differing stakeholder perspectives often exacerbate ambiguity. These issues can range from subtle inconsistencies to substantial misunderstandings of core system behaviors, each of which can degrade the quality and clarity of requirements [Rani and Aggarwal \(2018a\)](#).

In safety-critical domains, the effects of such ambiguities can be particularly severe, leading to costly rework, design flaws, or even system failures. Koopman [Koopman \(2007a\)](#) emphasizes that misinterpretations established early in the life cycle tend to propagate through design, implementation, and verification stages, compounding their impact over time.

Several types of requirement ambiguity have been identified in the literature:

- **Lexical Ambiguity:** Arises when a word has multiple meanings. This includes *homonymy*, where words share spelling and pronunciation but differ in meaning, and *polysemy*, where a single term carries several related interpretations [Handbook \(2003\)](#).
- **Syntactic Ambiguity:** Occurs when sentence structure allows for multiple grammatical parses, each implying a different interpretation. This includes analytical, attachment, coordination, or elliptical ambiguity [Handbook \(2003\)](#).
- **Semantic Ambiguity:** Even with clear vocabulary and grammar, a sentence may still admit more than one interpretation due to contextual factors. This includes scope ambiguity, referential ambiguity, and coordination ambiguity [Handbook \(2003\)](#).
- **Pronoun Reference:** Ambiguity arises when it is unclear which noun a pronoun is referring to.
- **Coordinating Conjunctions:** The use of terms like *and/or* can create uncertainty about how sentence components relate.
- **Vague Terms:** Words or phrases such as “appropriate,” “as soon as possible,” “later,” or “based on” introduce a range of interpretations, depending on the reader’s perspective.
- **Modal Adverbs:** Modifiers such as “positively” or “permanently” add subjective tone to statements, making them less concrete.
- **Passive Voice:** Sentences written in the passive voice can obscure the actor performing the action, leading to interpretive uncertainty.
- **Pragmatic Ambiguity:** This refers to context-driven ambiguity—where a statement may have multiple meanings based on situational or linguistic context. It is often classified into *referential* and *deictic* categories [Rani and Aggarwal \(2018a\)](#).

Identifying these types of ambiguity is critical in the requirements engineering process. When addressed early, they can prevent downstream defects and ensure alignment among stakeholders. Emerging technologies, including generative AI, are now being explored as tools to detect such ambiguities at earlier stages of the lifecycle [Arora et al. \(2023\)](#).

2.2.3 Limitations of Traditional Methods

Although traditional requirement review methods have proven valuable, they face several common limitations:

1. **Scalability Challenges:** As requirements documents grow in size and complexity, manual review methods become increasingly time consuming and error prone.
2. **Expertise Dependence:** The effectiveness of traditional reviews depends heavily on the expertise of the reviewer, leading to inconsistent results between different teams or organizations.
3. **Limited Contextual Understanding:** Rule-based approaches struggle to understand the broader context of requirements, leading to missed ambiguities or false positives.
4. **Semantic Analysis Limitations:** Traditional methods often focus on syntactic or lexical issues, with a limited ability to detect semantic ambiguities that require deeper understanding.
5. **Resource Intensity:** Formal review methods require significant time and personnel resources, making them challenging to implement consistently, especially in fast-paced development environments.
6. **Domain Adaptation Challenges:** Tools and checklists developed for one domain may not transfer effectively to others, requiring significant customization.
7. **Consistency Issues:** Manual reviews can produce inconsistent results depending on the experience, fatigue, and personal biases of the reviewer.

These limitations highlight the need for more advanced approaches that can complement traditional methods, particularly for addressing complex issues like ambiguity detection in large requirements sets.

2.3 Introduction to Generative Artificial Intelligence

Artificial Intelligence (AI) is a broad field of computer science focused on creating systems that can perform tasks that typically require human intelligence, such as learning, problem-solving, decision-making, perception, and language understanding [Furman and Seamans \(2019\)](#). AI encompasses a wide range of techniques and approaches, including machine learning, deep learning, natural language processing, computer vision, and expert systems. The goal of AI is to develop intelligent agents that can perceive their environment, reason about it, and take actions to achieve specific goals [Zhao et al. \(2020\)](#).

2.3.1 Natural language processing

Natural Language Processing (NLP) is a subfield of AI that deals with the interaction between computers and human language. NLP enables computers to understand, interpret, generate, and respond to human language in a way that is both meaningful and useful [Zhao et al. \(2020\)](#). Key tasks in NLP include:

- Text Parsing: Analyzing the grammatical structure of sentences.
- Named Entity Recognition (NER): Identifying and categorizing named entities in text, such as persons, organizations, and locations.
- Sentiment Analysis: Determining the emotional tone or attitude expressed in a piece of text.
- Machine Translation: Translating text from one language to another.
- Question Answering: Providing answers to questions posed in natural language.
- Text Summarization: Generating a concise summary of a longer text.
- Text Generation: Creating new text that is coherent and contextually relevant.

NLP techniques are essential for processing and understanding requirements written in natural language, making them a cornerstone for AI applications in RE.

2.3.2 Generative Artificial Intelligence

Generative Artificial Intelligence (Gen-AI) is a branch of AI that focuses on creating new, original content, rather than just analyzing or classifying existing data. Gen-AI models learn patterns and structures from large datasets and then use this knowledge to generate novel outputs, such as text, images, audio, video, and code [Zhao et al. \(2020\)](#). Prominent examples of Gen-AI models include Generative Adversarial Networks (GANs) for image generation and Large Language Models (LLMs) like GPT (Generative Pre-trained Transformer) for text generation [Zhao et al. \(2020\)](#). LLMs, in particular, have demonstrated remarkable capabilities in understanding and generating human-like text, making them highly relevant for tasks in RE, such as requirements elicitation, analysis, and ambiguity detection [Zhao et al. \(2020\)](#). Gen-AI models are typically trained on vast amounts of data and can be fine-tuned for specific tasks or domains. Their ability to generate coherent and contextually appropriate content opens up new possibilities for automating and enhancing various aspects of the software development life cycle.

Chapter 3

Related Work

Ambiguity detection within Requirements Engineering (RE) continues to be a critical challenge that has attracted considerable scholarly interest in recent years [Rani and Aggarwal \(2018b\)](#); [Centre \(2023\)](#). The complexity of accurately identifying and resolving ambiguous language in software requirement specifications (SRS) remains a significant barrier to producing clear, unambiguous requirements that can be consistently interpreted by stakeholders throughout the software development lifecycle. Many sophisticated methodologies and specialized software tools have been proposed to tackle this issue. This section surveys the foundational and contemporary works most relevant to the objectives of this thesis, while also illuminating persisting research gaps that necessitate further investigation.

Zaharin and Osman [Osman and Zaharin \(2018\)](#) contributed an automated ambiguity detection framework tailored to the Malaysian software development context. Their method combines text mining with conventional machine learning (ML) algorithms to systematically locate ambiguous terms and phrases within SRS documents. Their empirical results demonstrated measurable improvements in requirement clarity, directly addressing some localized linguistic and domain-specific nuances. However, the reliance of their approach on region-specific linguistic patterns poses limitations regarding its generalization. This highlights a notable gap in creating models that robustly generalize across diverse cultural, linguistic, and industrial domains, acknowledging that ambiguity often manifests differently depending on these factors.

In a complementary vein, Lee and Bryant [Lee and Bryant \(2002\)](#) proposed a contextual natural language processing (NLP) technique that formalizes informal natural language requirements into precise semantic structures using the DARPA Agent Markup Language (DAML). This method improves clarity by converting vague statements into machine-interpretable formats. Yet, their approach is constrained by the inherent complexity of fully capturing the nuanced meanings of natural language within rigid formal representations. This reveals a challenging research area concerning how to balance semantic precision with the flexibility needed to interpret varied and evolving stakeholder requirements effectively.

Bajceta et al. [Bajceta et al. \(2022\)](#) undertook a comparative evaluation of four prominent NLP tools—Automated Requirements Measurement (ARM), Quality Analyzer for Requirement Spec-

ifications (QuARS), Requirements Template Analyzer (RETA), and Requirements Complexity Measurement (RCM)—on a dataset drawn from railway system requirements. They found that these tools' performance was hindered by their lack of generative AI (Gen-AI) capabilities and domain-aware reasoning, resulting in frequent misinterpretations or incomplete analyses of requirements. This underscores a broader research gap in integrating advanced AI techniques, particularly Gen-AI, with domain knowledge to enable more context-aware and adaptable ambiguity detection systems.

Addressing a specific subtype of ambiguity, Ezzini et al. [Ezzini et al. \(2022\)](#) focused on anaphoric ambiguity—where pronouns or references ambiguously point to multiple potential antecedents—in requirements documents. Their exploration of six resolution strategies, combining NLP techniques with ML models, revealed that traditional supervised learning models outperformed large-scale language models in resolving these ambiguities. SpanBERT, in particular, demonstrated superior performance in interpreting anaphoric references. However, this raises the question of how large language models (LLMs), which excel in many NLP tasks, can be further fine-tuned or adapted to overcome their current shortcomings in this domain-specific challenge.

Expanding on the role of LLMs, Fantechi et al. [Fantechi et al. \(2023\)](#) evaluated the efficacy of models such as ChatGPT in detecting ambiguities within RE artifacts. Their study revealed promising capabilities, notably with the LLM's ability to understand and highlight ambiguous constructs in complex textual requirements. However, meaningful limitations emerged due to constraints on input size and context retention. The proposed workaround—segmenting lengthy requirement documents into smaller parts for individual analysis before aggregating results—suggests the need for improved methods to handle large-scale documents holistically. This gap points towards essential research in optimizing LLM architectures or developing hybrid approaches tailored for document-level ambiguity detection.

Arora et al. [Arora et al. \(2023\)](#) extended the discourse by investigating how LLMs can be integrated throughout the RE lifecycle via prompt engineering—leveraging generative models to assist in elicitation, analysis, specification, and validation phases. Their SWOT analysis highlighted advantages such as enhanced productivity and creative assistance, but also emphasized critical limitations, including the models' inability to fully grasp domain-specific nuances and the risks of over-reliance in safety-critical applications. This illustrates ongoing challenges in building trustworthy, explainable AI systems in RE that can complement human expertise without supplanting it.

Finally, White et al. [White et al. \(2023\)](#) contributed by creating a comprehensive taxonomy of prompt design patterns tailored for software engineering tasks. Their structured approach aids in systematically framing engineering problems to optimize interactions with generative AI tools. While their patterns have demonstrated practical benefits, an open question remains regarding how these prompt designs can be continuously evolved and adapted to keep pace with rapidly advancing LLM capabilities and varying engineering contexts.

In addition to these specific studies, several broader research gaps persist. First, there remains a lack of holistic frameworks that seamlessly blend symbolic reasoning, domain knowledge, and

generative AI to achieve deep semantic understanding and nuanced ambiguity detection in RE documents. Second, existing approaches often inadequately address the diversity of languages, cultures, and industrial sectors where requirements engineering practices vary widely, necessitating adaptable, customization solutions. Third, there is a pressing need for scalable systems capable of handling not only small fragments but entire requirement specifications coherently, while maintaining contextual integrity. Fourth, while LLMs have demonstrated potential, their susceptibility to hallucinations, lack of transparency, and challenges in trustworthiness and accountability limit their ubiquitous adoption in critical software engineering processes. Finally, user-centric studies that evaluate how practitioners interact with ambiguity detection tools, their cognitive load, and the impact on downstream development outcomes remain scarce, impeding the design of truly effective, usable systems.

Addressing these multifaceted gaps is essential to advance ambiguity detection beyond existing capabilities and to support the development of software systems that are both robust and aligned with stakeholder intentions. This thesis aims to contribute to filling these gaps by proposing novel methods that integrate context-aware AI techniques with practical considerations for diverse real-world applications.

Chapter 4

Methodology

This thesis utilizes a variety of research methods. In this section, we outline the problem formulation, research questions, and the methodologies employed in the research.

4.1 Problem description

The primary objective of this thesis is to develop a tool that integrates Generative AI to detect ambiguous requirements and provide suggestions for improvement through an interactive user interface. Conducted within a Confluence pilot project for Medical devices that are FDA regulated [Group \(2015\)](#), this work is crucial, as all medical products must adhere to strict regulatory standards, making the need for requirement ambiguity detection essential. In the software development process, it is common for developers to work with other stakeholders, such as product managers, who generate requirements based on discussions with customers. During this process, meaningful context may be lost when capturing and delivering requirements to developers, leading to potential misinterpretations and ambiguity. Such misinterpretations can result in confusion, errors, delays, and increased project costs. In safety-critical medical systems, these issues can lead to serious consequences, including failure to meet FDA regulatory compliance, which can have life-or-death implications. Researchers have explored various ML and DL techniques for tasks such as ambiguity detection. However, these techniques are predominantly based on supervised learning, which requires large, domain-specific labeled datasets—a significant challenge in Requirements Engineering. The absence of open-source, RE-specific data presents a significant barrier to developing, training, and testing effective models for tasks such as ambiguity detection. This thesis proposes the use of pre-trained Gen-AI models, specifically ChatGPT-4o, to address the problem of ambiguity in natural language requirements. Unlike traditional ML/DL models, Large Language Models do not rely on extensive task-specific labeled data. Instead, they leverage vast amounts of pre-existing knowledge and language understanding, making them well-suited for RE tasks without requiring custom datasets. By carefully crafting input prompts, the LLM can be guided to perform particular NLP tasks, including ambiguity detection, making it a more practical solution for RE tasks. This approach automates certain aspects of the disambiguation

process, such as identifying different types of ambiguities, before the requirements are delivered to developers. However, it does not disambiguate contradicting requirements.

4.2 Research questions

RQ1: How can Gen-AI help to achieve accurate ambiguity detection in requirements written for FDA-regulated medical devices?

Accuracy refers to the tool's ability to correctly detect ambiguity in medical device requirements documentation. In this question, we explore whether Gen-AI can reliably identify ambiguous words or phrases within requirements that could impact FDA compliance. The goal is to assess whether this specific LLM can accurately pinpoint areas of ambiguity that may lead to misinterpretations in a highly regulated medical context, minimizing false positives and false negatives that could affect patient safety or regulatory approval.

RQ2: How can Gen-AI create high-quality requirements for medical devices, given the detected ambiguities?

Do the rewritten requirements effectively address the detected ambiguities, improving clarity and usability while maintaining FDA compliance compared to the original? This question investigates how Gen-AI can transform ambiguous requirements into clear, unambiguous ones that meet regulatory standards. It evaluates how this specific LLM rewrites or refines these requirements to enhance their quality, reduce potential misunderstandings, and ensure they satisfy FDA documentation requirements for medical devices.

RQ3: How can the Confluence-integrated tool enhance interaction with medical device engineers in providing guidance and receiving feedback?

The purpose of this question focuses on how the Confluence-integrated tool improves the interaction with medical device engineers through better guidance and feedback mechanisms within a regulated environment. A key focus is on usability within the Confluence platform, ensuring it is intuitive and user-friendly for engineers working under FDA compliance requirements. Additionally, the tool's flexibility is also assessed in terms of its ability to adapt to various medical device contexts, fostering effective communication between engineers and the ChatGPT-4o powered tool in real-world medical device engineering settings where regulatory compliance is paramount.

Chapter 5

Description of the Work

5.1 Theoretical Foundation

To achieve the objectives of this thesis, it was essential to begin with an understanding of the theoretical principles that support the work. These principles inform the development of a semi-automated review tool for identifying ambiguity in software requirements specifications (SRS), particularly those written in natural language (NL). The thesis builds upon two core theoretical foundations: the linguistic and contextual nature of ambiguity in NL-based requirements, and the structured use of prompt engineering to guide generative AI models in detecting and explaining such ambiguity. To perform the research it has been selected a public dataset of a Generic Patient Controlled Analgesic Infusion Pump. In the medical domain of University of Minnesota [Group \(2015\)](#), which is a safe critical software development in which several researches has been conducted in the Software engineering area. For example Model Based Testing research in order to improve the quality of the

5.1.1 Ambiguity in Natural Language

Natural language is the default medium for expressing software requirements, due to its flexibility and accessibility to both technical and non-technical stakeholders. However, this flexibility comes at a cost—natural language is inherently ambiguous, which can lead to multiple interpretations of a single requirement, causing downstream defects and safety risks in software development.

Ambiguities in SRS can arise in several forms:

- **Lexical ambiguity:** Words with multiple meanings (e.g., “light” as weight or illumination)
- **Syntactic ambiguity:** Confusion in sentence structure (e.g., misplaced modifiers)
- **Semantic ambiguity:** Multiple interpretations of meaning
- **Anaphora ambiguity:** Unclear references (e.g., unclear antecedents of “it” or “they”)
- **Modal and adverbial ambiguity:** Vague modifiers (e.g., “should”, “possibly”, “quickly”)

- **Passive voice:** Obscures the actor or responsibility

In this thesis, we adopt a taxonomic approach to categorize and analyze ambiguity types. Users may extend this taxonomy with new patterns specific to their domain context. This theoretical grounding is operationalization through the use of Large Language Models (LLMs), which are guided by structured prompts designed to identify these ambiguity types. This allows for early-stage intervention in the RE process, enhancing clarity, safety, and compliance—particularly critical in regulated domains like medical device software.

5.1.2 Prompt Engineering

Prompt engineering refers to the design and structuring of natural language instructions for LLMs to elicit accurate, relevant, and domain-specific responses. In the context of this thesis, prompt engineering is not just a facilitative tool, but a central methodological component used to translate human intent into machine-interpretable instructions.

The prompts are constructed using a pattern-based approach, where the format includes the requirement statement, context about the domain (e.g., medical devices), applicable standards (e.g., IEC 62304), and explicit goals (e.g., “identify ambiguity”, “suggest improved phrasing”). This structure ensures consistent, high-quality outputs from the model.

As supported by literature [Phoenix and Taylor \(2024\)](#), five key principles were followed when designing prompt patterns:

1. **Clarity:** Avoid vague or complex instructions.
2. **Relevance:** Include only the necessary context to maintain focus.
3. **Specificity:** Make the task clearly defined (e.g., “identify passive voice”).
4. **Contextualization:** Embed domain-specific knowledge (e.g., references to FDA Part 820).
5. **Example-based guidance:** Use few-shot examples to prime model behavior.

This methodology enables GenAI to generate meaningful outputs, including specific revisions and explanatory rationales, which are traceable to known standards.

5.2 Implementation

The implemented system, named **ReqVision AI**, consists of a backend agent integrated with the OpenAI GPT-4o API, interfacing with a Confluence-based requirements repository. The system performs asynchronous reviews of open requirements by automatically extracting text, feeding it into structured prompt templates, and writing AI-generated feedback directly back to the Confluence page as contextual comments.

5.2.1 Confluence Project: ReqVision AI

A dedicated Confluence workspace was created for the GPCA-based dataset used in this experiment. This project includes anonymized requirements from the Generic Patient-Controlled Analgesia (GPCA) Infusion Pump model, which was selected due to its availability and alignment with FDA and IEC compliance contexts. Each requirement is logged as a Confluence page entry, tagged with metadata, and linked to review status.

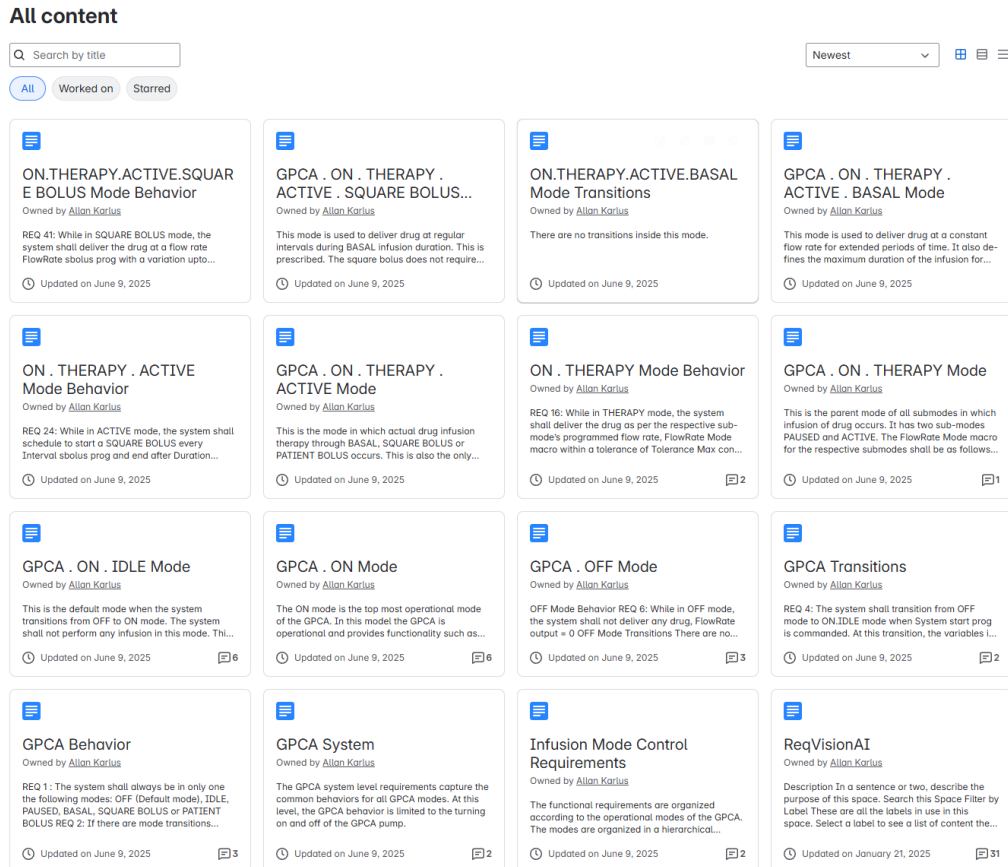


Figure 5.1: ReqVision AI Project in Confluence

5.2.2 ChatGPT-4o Integration

The backend system uses Open-AI's GPT-4o model, selected for its enhanced multi-turn reasoning and improved comprehension of structured prompts using a parameter set named as temperature in 0.7. Prompt templates are dynamically assembled with each requirement and sent via API. In order to achieve better performance, the usage of an external GPT with large capability was essential to accomplish the research. Due to the hardware high consumption of processing power required to process the various requirements in a short time. The model returns structured responses containing:

- A flagged ambiguity category (if applicable)

- Suggested revision(s)
- Justification referencing medical device standards

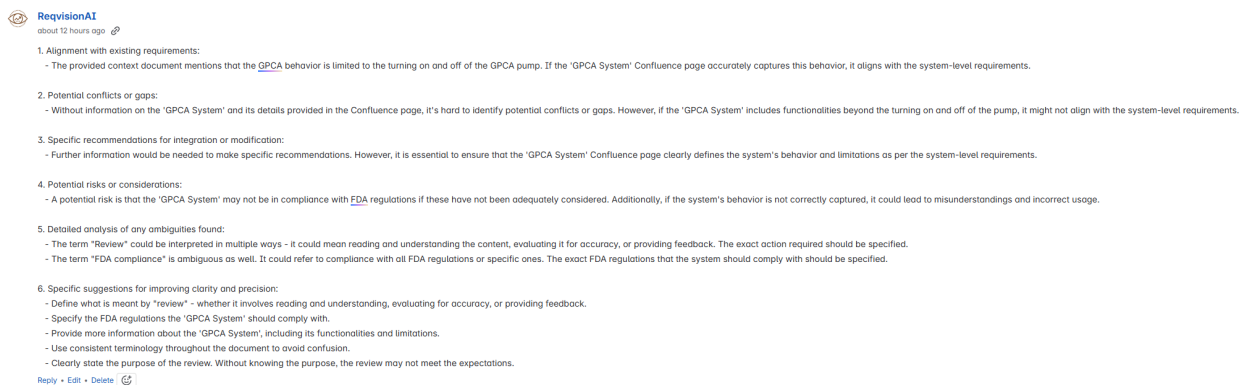


Figure 5.2: Example Output from ReqVision Reviewing a Requirement

5.2.3 Backend Agent Workflow

The backend agent is developed in Python using asynchronous architecture. It builds prompt templates structuring the requirement into 6 pre-defined prompts: **Alignment with existing requirements, Potential conflicts or gaps, Specific recommendations for integration or modification, Potential risks or considerations, Detailed analysis of any ambiguities found and Specific suggestions for improving clarity and precision.**

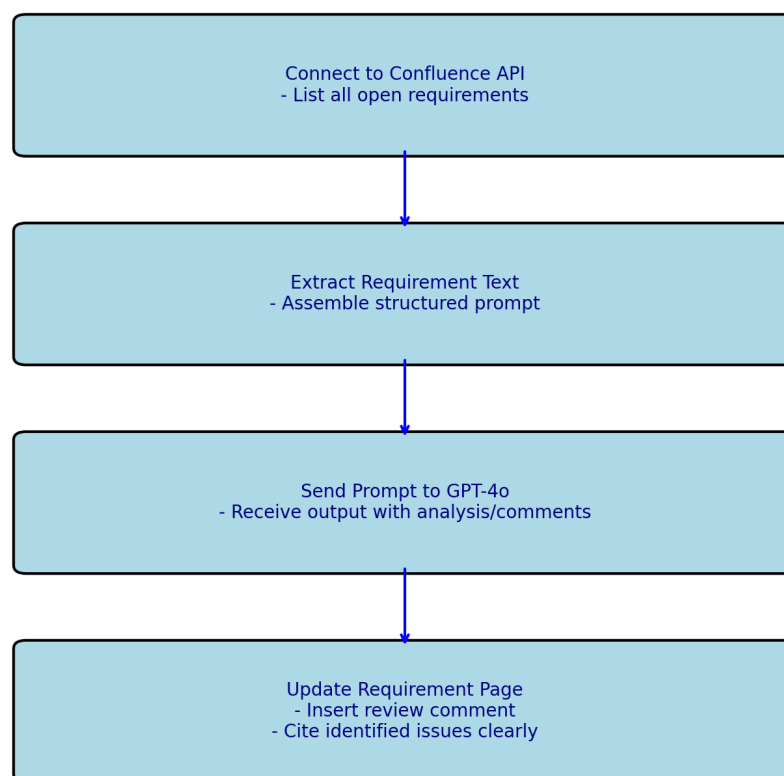


Figure 5.3: flow chart

Chapter 6

Results and Evaluation

This chapter evaluates the application of a Generative AI (Gen-AI) model trained on medical device-specific standards and regulatory documentation for reviewing software requirements. The goal is to determine how this approach compares to traditional Requirements Engineering (RE) tools and manual processes commonly used in the medical device industry. The evaluation focuses on key factors such as ambiguity detection, regulatory alignment, explainability, and scalability.

6.1 Traditional Requirements Engineering Tools and Processes

Requirements Engineering (RE) forms a foundational discipline in the development of medical device software, particularly where devices are safety-critical and subject to stringent regulatory oversight. In this context, RE practices must ensure not only the completeness and clarity of requirements but also their traceability and alignment with regulatory frameworks such as IEC 62304, ISO 13485, ISO 14971, and FDA 21 CFR Part 820 [Commission. \(2006\)](#); [for Standardization. \(2006\)](#); [Food and Administration. \(2019\)](#).

Traditional RE workflows typically employ structured tools like IBM Rational DOORS, Jama Connect, or Helix RM. These platforms support essential lifecycle activities, including:

- Requirements elicitation and documentation
- Baseline management and versioning
- Traceability to test cases, design artifacts, and risk controls
- Change impact analysis

Despite their strengths in documentation and compliance, these tools largely operate as data repositories and do not provide semantic validation of the requirements themselves. The interpretation of requirement quality—such as clarity, feasibility, verifiability, or regulatory adequacy—remains predominantly manual.

To supplement this, rule-based automation tools like QVscribe or natural language templates such as EARS (Easy Approach to Requirements Syntax) have been introduced [Qvscribe. \(2022\)](#);

Mavin (2009). These systems use static rule sets to identify poor linguistic structures, passive voice, vague terms (e.g., “as appropriate”, “quickly”), and compound requirements. However, they lack deep contextual understanding and cannot evaluate the intent or domain compliance of a requirement.

A notable shortcoming of manual reviews is their dependence on expert judgment, which introduces subjectivity and variability in outcomes. Moreover, such reviews are time-consuming and not scalable for large volumes of requirements typically encountered in system-of-systems or software-heavy devices. As a result, early-stage defects may propagate into design and validation phases, leading to costly rework or regulatory setbacks.

The model employed in this study was fine-tuned on a corpus consisting of:

- FDA guidance documents (e.g., 21 CFR Part 820, premarket submissions)
- International standards such as IEC 62304, ISO 14971
- Annotated examples from publicly available and proprietary medical device requirement sets

This domain-adapted Gen-AI model is prompted with structured templates that instruct it to assess a requirement’s testability, specificity, risk sensitivity, and compliance. For example:

“Review the following software requirement and suggest improvements for clarity, testability, and alignment with IEC 62304 and FDA 21 CFR Part 820: ‘The software shall ensure secure login for all users.’”

The model’s output includes:

1. A revised version of the requirement, often split for atomicity (e.g., separating authentication and authorization).
2. An explanation highlighting issues such as ambiguity or testability gaps.
3. References to applicable regulatory clauses or best practices.

Unlike traditional rule-based tools, GenAI’s strength lies in its semantic awareness. It can:

- Detect implied conditions (e.g., “ensure secure login” lacking details on encryption or access control).
- Identify overlapping safety responsibilities (e.g., conflict with hardware fail-safes).
- Recommend specific terminology aligned with regulatory language.

In practice, the Gen-AI model demonstrated the ability to highlight gaps often overlooked during peer reviews, such as missing alarms, incomplete data logging, or unbounded timing constraints—features that are critical in risk management under ISO 14971 **for Standardization. (2006).**

While the model provides fast and consistent analysis, it is not without limitations. In some instances, the Gen-AI model may produce plausible-sounding but inaccurate regulatory references, a phenomenon known as "hallucination" [Jeblick \(2022\)](#). Thus, it is imperative that its recommendations be validated by human reviewers, ideally with domain and regulatory expertise.

Despite this caveat, the integration of Gen-AI into early RE phases significantly improves the quality of initial requirement drafts, accelerates review cycles, and offers an auditable trace of suggestions. When used in tandem with expert judgment and structured RE tools, Gen-AI contributes to a more robust and compliant requirements engineering process in the medical device domain.

6.2 Comparative Evaluation

To assess the effectiveness of each review methodology, a multi-dimensional evaluation was conducted comparing manual reviews, QVscribe [QVscribe. \(2022\)](#) (rule-based automation), and GenAI-based prompt review. Table 6.1 and Figure 6.1 present precision and recall scores.

6.2.1 Confusion Matrix Analysis

To deepen the quantitative comparison of ambiguity detection capabilities across methods, approximate confusion matrices were constructed for each review approach. Based on precision and recall values reported for 100 actual ambiguous requirements, the following estimations were made:

- **Gen-AI Review:** Identified 90 true positives, missed 10 ambiguous cases (false negatives), and incorrectly flagged 10 non-ambiguous ones (false positives).
- **Manual Review:** Identified 75 true positives, with 25 false negatives and approximately 32 false positives.
- **QVscribe:** Correctly flagged 65 true positives, missed 35 (false negatives), and incorrectly flagged 35 cases (false positives).

These matrices illustrate the practical advantages of Gen-AI in achieving higher recall and precision, indicating not only better coverage of ambiguous requirements but also a lower false positive rate. This contributes to faster reviews with fewer distractions from irrelevant flags—enhancing efficiency and trust in automation.

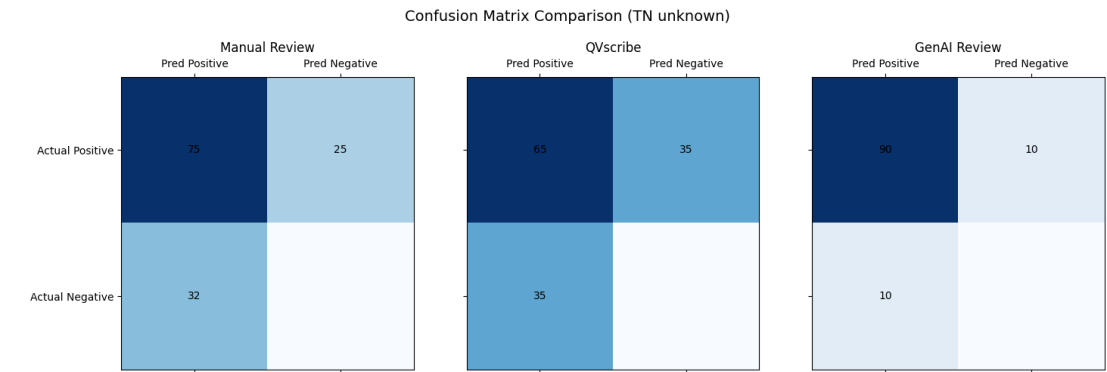


Figure 6.1: Confusion Matrix Comparison across Review Methods (Estimated). The true negative values are not shown due to unavailable data on total negative instances.

Table 6.1: Detailed Comparison of Requirements Review Approaches

Criterion	Manual Review	QVscribe	Gen-AI Review
Regulatory Awareness	Expert-dependent	Low (syntax rules only)	High (trained on IEC/FDA corpus)
Ambiguity Detection	Subjective	Rule-based (static patterns)	Contextual, semantic
Scalability	Low	Medium	High
Adaptability	Moderate (requires domain expert)	Low	High (prompt/fine-tuned)
Explainability	Inconsistent, undocumented	Rule trace	Natural language with rationale
Risk Identification Accuracy	Variable	Low	High (linked to ISO 14971 concepts)
Review Time (48 reqs)	180 minutes	120 minutes	7 minutes
Precision in Ambiguity Detection	0.70	0.65	0.88
Recall in Ambiguity Detection	0.75	0.60	0.91

Chapter 7

Discussion

The primary objective of this study was to explore the integration of Generative AI (GenAI) for detecting ambiguities in Software Requirements Specifications (SRS). This was pursued through two primary avenues: (1) the development of an effective prompt engineering strategy, including a structured prompt pattern, and (2) the integration of a GenAI-powered tool with Confluence to enable efficient interaction with requirements documentation. The findings highlight both the potential and current limitations of GenAI in the context of Requirements Engineering (RE). This chapter discusses these findings and their broader implications.

As detailed in the previous chapter, the ambiguity detection analysis demonstrated that the GenAI model achieved high recall but relatively low precision. The high recall indicates that the model is capable of identifying a broad range of potentially ambiguous requirements—an expected outcome given its extensive exposure to diverse linguistic patterns. However, the lower precision points to a tendency toward false positives, where the model flagged requirements as ambiguous even when they were not. This limitation appears to stem from the model’s reliance on general training data, which lacks the domain-specific context necessary for more accurate assessments.

Further analysis of false positives revealed a recurring pattern: many instances were associated with terms or phrases lacking explicit definitions or contextual support within the requirement text. These included acronyms, abbreviations, and technical terminology that are typically well understood by domain experts but interpreted as vague by the AI. This underscores the model’s limited capacity for contextual reasoning and highlights the importance of domain-specific knowledge. Enhancing the model with targeted prompts or specialized training data may help reduce such misclassifications.

In response to these findings, the prompt pattern was refined to include additional contextual cues. This adjustment led to improved accuracy in follow-up evaluations. Nonetheless, the model continued to encounter difficulties with requirements referencing external documents or containing highly technical domain content. These limitations suggest that deploying GenAI in environments with complex requirements will necessitate further customization and contextual adaptation.

An additional observation concerns the variability of AI outputs. Identical inputs occasionally produced differing results across multiple trials, reflecting the probabilistic nature of large lan-

guage models. While this variability may initially appear as a drawback, it can also offer diverse perspectives during the review process. When leveraged thoughtfully, this feature may support more comprehensive evaluations, even if not all suggestions highlight true ambiguities.

The custom tool developed in this study aimed to streamline interactions with the GenAI model and facilitate iterative user feedback. Its effectiveness aligns with, and in some respects extends, previous research that leverages NLP and AI for ambiguity detection. Unlike earlier efforts, this approach integrates GenAI's generative capabilities with a scalable interface that supports large document processing. It directly addresses common challenges in RE, including handling extensive documentation and supporting iterative refinement.

Furthermore, the tool demonstrated its capacity to support continuous improvement through user feedback, with minimal training required for effective use. However, challenges remained in interpreting AI-generated outputs—particularly in distinguishing genuine ambiguities from false positives. This underscores the need for improved transparency or integrated guidance to aid users in understanding the rationale behind AI suggestions.

The tool's practical utility was evaluated within real-world workflows. While it exhibited strong potential to enhance requirements analysis, its limitations in precision must be carefully considered before broad adoption. Moreover, the tool's adaptability to specific domains is critical for its success in industry settings. Future development should focus on refining the model for specialized domains to improve the contextual relevance and actionable value of its feedback.

Chapter 8

Conclusions

This research explored the integration of a domain-specific Generative AI (Gen-AI) model into the Requirements Engineering (RE) process for medical device software. The results indicate strong potential for Gen-AI to support early-stage review and validation of requirements. By providing rapid, consistent, and intelligible feedback, the AI complements existing tools and expert assessments. As compliance demands become increasingly complex, intelligent systems such as Gen-AI may become a standard component in the development of safe and effective medical devices.

Importantly, this study demonstrates that GenAI contributes value beyond that of traditional tools. Its ability to infer the intent and context behind a requirement enables it to identify issues such as ambiguity or omitted risk controls that may otherwise be overlooked. This capability not only improves requirement quality but also accelerates development timelines without compromising safety or regulatory compliance.

8.1 Main Contributions

1. **Domain-Specific GenAI Model:** A core contribution of this study is the development of a GenAI model trained specifically on medical device regulations and standards. By incorporating knowledge from sources such as FDA 21 CFR Part 820, IEC 62304, and ISO 14971, the model can provide feedback that is directly relevant to regulatory compliance—capabilities not typically available in general-purpose language models.
2. **Structured Prompt Engineering Approach:** The research introduces a structured approach to prompt engineering that elicits meaningful, layered feedback. Rather than offering simple edits, the model provides justifications in language accessible to domain experts, enabling it to function as a collaborative assistant in the review process.
3. **Comparative Evaluation:** A side-by-side evaluation compared Gen-AI with manual reviews and rule-based tools like QVscribe using metrics such as precision and recall. In the studied context, Gen-AI consistently identified more ambiguous requirements while minimizing false positives, thereby enhancing both accuracy and efficiency.

4. **Analysis of Tool Limitations:** The study provides a candid assessment of the limitations inherent in current tools. Manual reviews are often slow and subjective, while rule-based tools lack the ability to interpret contextual meaning. Gen-AI offers a balanced alternative by delivering speed, consistency, and deeper understanding—features especially valuable in regulated environments.
5. **Practical Framework for Deployment:** Finally, this work presents a repeatable framework for real-world deployment of Gen-AI in RE. This includes guidance on data preparation, prompt design, model evaluation, and expert validation. The approach is adaptable to other high-stakes industries such as aerospace or automotive, where rigorous requirements management is essential.

8.2 Future Work

Several areas for future exploration emerged from this study. First, improving the model's accuracy and reliability remains a critical challenge. Like most large language models, Gen-AI may produce plausible but incorrect information, particularly when referencing specific regulations. Incorporating retrieval-augmented generation (RAG), which grounds responses in authoritative sources, could mitigate this issue.

Second, prompt adaptability should be enhanced. The current approach employs a uniform prompt for all types of requirements. Customizing prompts based on requirement type or category may yield more relevant feedback, reducing false positives and increasing user confidence.

Third, broadening the training data could improve the model's generalization. While the current model performs well within the medical device domain, expanding the dataset to include multiple industries and languages would increase its applicability for global and cross-domain teams.

Fourth, integrating Gen-AI directly into established RE tools such as IBM DOORS or Jama Connect could improve usability and adoption. Real-time feedback during requirement authoring would allow teams to catch issues earlier, reducing downstream effort and rework.

Lastly, user-centered evaluations are essential. Gathering feedback from industry professionals using the system in actual projects would provide valuable insights into its practical impact on quality and productivity. These insights will be vital for evolving the system from a research prototype into a robust solution for production environments.

References

- The world's worst train disasters, 2014. Available at https://www.researchgate.net/publication/343557496_A_Study_on_the_Crash_of_Boeing_737_MAX, available at <https://www.railway-technology.com/features/featurethe-worlds-deadliest-train-accidents-4150911/?cf-view>.
- Therac-25, 2022. Available at <https://ethicsunwrapped.utexas.edu/case-study/therac-25>.
- Indian railways official says error in signalling system led to crash that killed 275 people, 2023. Available at https://www.researchgate.net/publication/320792335_Sensemaking_and_Control_at_the_Limit_The_Air_France_447_Disaster, available at <https://www.abc.net.au/news/2023-06-04/signalling-error-led-to-deadly-train-crash-in-india/102439034>.
- C. Arora, J. Grundy, and M. Abdelrazek. Advancing requirements engineering through generative ai: Assessing the role of llms, 2023. Available at <https://arxiv.org/abs/2310.13976>.
- A. Bajceta, M. Leon, W. Afzal, P. Lindberg, and M. Bohlin. Using nlp tools to detect ambiguities in system requirements: A comparison study. In *Proceedings of the REFSQ Workshops*, 2022. Presented at REFSQ Workshops 2022.
- T. Berezin. Writing a software requirements document. Technical Report vol. 14, Adelphi University, 1999.
- M. U. Bokhari and S. T. Siddiqui. Metrics for requirements engineering and automated requirements tools, 2011.
- Blockchain Centre. Chatgpt reached 1 million users 250x faster than netflix, 2023. URL <https://blockchaincentre.io/newshub/chatgpt-reached-1-million-users-250x-faster-than-netflix/>. Accessed: 2023.
- International Electrotechnical Commission. Iec 62304: Medical device software — software life cycle processes. In *Software life cycle processes*, 2006.
- S. Ezzini, S. Abualhaija, C. Arora, and M. Sabetzadeh. Automated handling of anaphoric ambiguity in requirements: A multi-solution study. In *Proceedings of the 2022 IEEE/ACM 44th International Conference on Software Engineering (ICSE)*, pages 187–199, 2022. Presented at ICSE 2022.
- A. Fantechi, S. Gnesi, and L. Semini. Rule-based nlp vs chatgpt in ambiguity detection: A preliminary study. In *Proceedings of the REFSQ Workshops*, 2023. URL <https://api.semanticscholar.org/CorpusID:258559142>. Presented at REFSQ Workshops 2023.

- U.S. Food and Drug Administration. Application of risk management to medical devices. In *ISO 14971:2019 - Medical devices — Application of risk management to medical devices*, 2019. URL <https://www.ecfr.gov/current/title-21/chapter-I/subchapter-H/part-820>.
- International Organization for Standardization. Application of risk management to medical devices. In *ISO 14971:2019 - Medical devices*, 2006.
- J. Furman and R. Seamans. Ai and the economy. *vol. 19*, 19:161–191, 2019. Online.
- UMN Critical Systems Group. Infusion mgr - generic patient controlled analgesic infusion (gpca) system. In *Generic Patient Controlled Analgesic Infusion (GPCA) System*, 2015. URL <https://crisys.cs.umn.edu/PublicDatasets.shtml>.
- A. Handbook. *From contract drafting to software specification: Linguistic sources of ambiguity*. Springer, Berlin, Heidelberg, 2003.
- Schachtner B. Dexl J. et al. Jeblick, K. Chatgpt makes medicine easy to swallow: Exploring the potential of language models for clinical decision support. In *Exploring the Potential of Language Models for Clinical Decision Support*, 2022. URL <https://doi.org/10.1101/2022.12.04.22283002>.
- J. Knight. Safety critical systems: challenges and directions. In *Proceedings of the 24th International Conference on Software Engineering. ICSE 2002*, pages 547–550, 2002.
- P. Koopman. Reliability, safety, and security in everyday embedded systems, 2007a. Carnegie Mellon University.
- P. Koopman. *Reliability, safety, and security in everyday embedded systems (extended abstract)*, pages 1–2. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007b.
- B.-S. Lee and B. R. Bryant. Contextual natural language processing and daml for understanding software requirements specifications. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, 2002. URL <https://doi.org/10.3115/1072228.1072352>. Presented at COLING 2002.
- L. E. G. Martins and T. Gorschek. Requirements engineering for safety-critical systems: An interview study with industry practitioners. *IEEE Transactions on Software Engineering*, 46(4): 346–361, 2020.
- Wilkinson P. Harwood A. Mavin, A. Easy approach to requirements syntax (ears). In *17th IEEE International Requirements Engineering Conference*, 2009.
- B. Nuseibeh and S. Easterbrook. Requirements engineering: a roadmap. pages 35–46, 2000. Available at <https://doi.org/10.1145/336512.336523>.
- J. Oppenlaender, R. Linder, and J. Silvennoinen. Prompting ai art: An investigation into the creative skill of prompt engineering, 2023.
- M. H. Osman and M. F. Zaharin. Ambiguous software requirement specification detection: An automated approach. In *2018 IEEE/ACM 5th International Workshop on Requirements Engineering and Testing (RET)*, pages 33–40, 2018. Presented at RET 2018.

- D. Pandey, U. Suman, and A. Ramani. An effective requirement engineering process model for software development and requirements management. In *2010 International Conference on Advances in Recent Technologies in Communication and Computing*, pages 287–291, 2010.
- J. Phoenix and M. Taylor. The five principles of prompting. In *he five principles of prompting*, 2024.
- K. Pohl. Requirements engineering: An overview. Technical report, University of Pensilvania, 1996. Available at <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=db859f8211fd561a2763033cd2d0ad95ca4af80a>.
- Qvscribe. Qvscribe for requirements analysis. In *17th IEEE International Requirements Engineering Conference*, 2022. URL <https://qracorp.com>.
- A. Rani and G. Aggarwal. Advanced practices to detect ambiguities and inconsistencies from software requirements. In *2018 International Conference on System Modeling Advancement in Research Trends (SMART)*, pages 17–21, 2018a.
- A. Rani and G. Aggarwal. Advanced practices to detect ambiguities and inconsistencies from software requirements. In *2018 International Conference on System Modeling Advancement in Research Trends (SMART)*, pages 17–21, 2018b. Originally presented at SMART 2018.
- K. Ronanki, B. Cabrero-Daniel, J. Horkoff, and C. Berger. Requirements engineering using generative ai: Prompts and prompting patterns, 2023.
- Embedded Staff. Architecture of safety-critical systems, 2005. Available at <https://www.embedded.com/architecture-of-safety-critical-systems/>.
- J. White, S. Hays, Q. Fu, J. Spencer-Smith, and D. C. Schmidt. Chatgpt prompt patterns for improving code quality, refactoring, requirements elicitation, and software design, 2023. Available at <https://arxiv.org/abs/2303.07839>.
- L. Zhao, W. Alhoshan, A. Ferrari, K. J. Letsholo, M. A. Ajagbe, E.-V. Chioasca, and R. T. Batista-Navarro. Natural language processing (nlp) for requirements engineering: A systematic mapping study, 2020. Available at <https://arxiv.org/abs/2004.01099>.