SMU SINGAPORE MANAGEMENT UNIVERSITY
School of Computing and Information Systems

31 Jan 2026

Trustworthy Holistic Intelligent Agent Framework for Software Engineering

Research Statement

Background

I am Nhat-Minh Nguyen, a graduate of the Computer Science and Engineering department at Ho Chi Minh City University of Technology, Vietnam. I have been working as a Research Engineer at the SMU School of Computing and Information Systems under the supervision of Professor Lingxiao JIANG. During this time, I have developed a strong foundation in software engineering (SE) and Machine Learning/Deep Learning (ML/DL), specifically applying Graph Learning and Large Language Models (LLMs) to programming languages. Our team has published four papers on Vulnerability Detection for smart contracts using Graph Neural Networks (GNNs) and LLMs. Based on these research achievements, we are processing a patent filing and have launched a security startup, MANDO. Following my positive experience in SMU's research environment, I am eager to pursue a PhD to advance the field of AI4SE further.

Research Statement

1. **Motivation**: From Secure Smart Contracts to Trustworthy Holistic Software Intelligence

   To follow up on our achievements at SMU, I would like to go further in intelligent SE fields. While Generative AI has shown promise in coding, it suffers from a "reliability gap"—often generating code that is syntactically correct but functionally insecure, especially in the complex SE challenges. I aim to build a trustworthy holistic AI agent framework that functions across the entire software lifecycle by deeply integrating security compliance and indexing the code context under graph formats (GraphRAG). I believe the approach can bridge the gap between the SE challenges and the current generative AI capabilities.

2. Methodology

a. **Self-Orchestrating Multi-Agent Ecosystem** with **Adversarial Security**

To address the complexity of the full SE lifecycle, I propose moving beyond single-model generation to a **self-orchestrating multi-agent ecosystem**. Utilizing **Hierarchical Task Networks (HTN)**, a high-level "Architect Agent" will autonomously decompose abstract requirements into executable sub-tasks, delegating them to specialized worker agents (e.g., Backend Developer, QA Engineer, DevOps).

Crucially, to ensure secure code generation, I plan to implement an **Adversarial Critic Loop**. A dedicated "Security Auditor Agent"—governed by **Neuro-Symbolic** constraints—will rigorously evaluate generated artifacts against retrieved threat models and formal security policies (e.g., OPA). This creates a feedback loop where the system iteratively refines code until it satisfies both functional and security invariants. By integrating analysis tools directly into the agent's decision process, we ensure the output is not just statistically probable, but mathematically robust against known attack vectors.

b. Graph-based Retrieval-Augmented Generation (**GraphRAG**)

Standard Retrieval-Augmented Generation (RAG) fetches textual code snippets, which often fails to capture the complex logic and file structure necessary to detect vulnerabilities. Moreover, unlike natural language, code exhibits **high syntactic variance**—functionally identical code can look completely different due to variable naming and algorithmic choices. This limitation is critical when attempting to identify security flaws that rely on structural patterns rather than textual code.

Structural Retrieval: Instead of retrieving similar text, the agent retrieves relevant subgraphs (e.g., Control Flow Graphs, Program Dependence Graphs, or Call Graphs). This allows the agent to access verified "logic patterns" rather than just surface-level syntax for code assistant tasks. In the context of security, this enables the system to identify complex vulnerabilities such as reentrancy or improper access controls by analyzing the underlying graph structure of the code.

Application: In general SE, it could involve retrieving the correct API usage graphs for complex internal/external libraries, ensuring the agent generates code that is not only syntactically correct but also logically sound and secure. By following the retrieved standard project structure and security constraints, the application can proactively avoid security issues. This approach moves beyond simple code generation to a "correct-by-construction" paradigm.

3. Expected Contributions

My research aims to deliver a unified AI framework adaptable to multiple SE tasks, including **bug detection**, **repair,** and **secure code generation**. By combining the structural rigor of GraphRAG with the generative capabilities of LLMs, I aim to create AI systems that are safe and reliable for industrial applications. This work will directly support the technological growth of the MANDO ecosystem and contribute robust tools to the wider software engineering community.

4. Future Directions and Applications

In the future, I want to develop a trustworthy holistic AI agent framework that can cover the full reliable software lifecycle, not just **coding**, but also **CI/CD pipelines** and **operational monitoring**. By embedding industry standards, specifically the OWASP Top 10 and the NIST Secure Software Development Framework (SSDF), into the agent's graph-retrieval process, I can ensure that AI-generated software is secure by design and ready for the real world.