ISSN 2093-0542




KOREAN SOCIETY FOR INTERNET INFORMATION

The 20th Asia Pacific International Conference on Information Science and Technology (APIC-IST 2025)

July 06-09, 2025, SAii Laguna Resorts, Phuket, Thailand

http://www.apicist.org

Proceedings of APIC-IST 2025

| Organized by |

Korean Society for Internet Information (KSII)

Contents

Generating C3Tree Model with Non-Conditional Korean Requirements Specification for Cause-Effect Graph

Woosung Jang and R. Young Chul Kim*
Software Engineering Laboratory, Hongik University
Sejong, South Korea
[e-mail: uriel200@hongik.ac.kr, bob@hongik.ac.kr]
*Corresponding author: R. Young Chul Kim

Abstract

As the need for software validation of AI and software may be increasing. The automatic generation of test cases is a software validation method. To date, research on test case generation methods based on Korean requirements has been insufficient. In our previous study, we worked to automatically generate a Cause-Effect Graph via a C3Tree Model with informal Korean requirements. This approach is limited to non-conditional statements (clauses) of requirement specifications. That is, we cannot generate test cases from sentences that do not contain conditional clauses. To solve this, we propose a conversion mechanism that generates conditional clauses from non-conditional statements (clauses) in requirement specifications. For this, we use generative AI to classify two types of lists and sequential requirements, and automatically convert the classified sentences into conditional sentences. Then, we automatically generate a C3Tree Model from the conditional sentences. As a result, the accuracy of automatic test case generation is increased, and more test cases can be generated.

Keywords: Korean requirements, Cause-effect graph, Test case generation, Natural language classification

1. Introduction

As software becomes more advanced, software quality issues become more critical, and the importance of software testing also increases [1]. In particular, strict software tests are conducted in places where severe casualties may occur due to software malfunctions, such as in the military or heavy industry [2]. However, testing time is always insufficient. Everyone wants test automation, but the method of generating test cases from Korean natural language requirements remains inaccurate [3].

We study the automatic generation of test cases from Korean natural language requirements [4]. Previous studies have automatically generated a Conditional and Conjunction Clause Tree (C3Tree) Model, Cause-Effect Graph, Decision Table, and Test Case sequentially from Korean natural language requirements. Test cases generated from the cause-effect graph can achieve 100% test coverage with a minimum number of test cases [5]. However, the existing method cannot automatically generate sentences without condition clauses into the C3Tree Model.

To address the existing problem, we categorize requirement sentences that do not include condition clauses into a list and sequential types, and automatically classify sentences using

This research was conducted with the support of the Korea Creative Content Agency (Project Name: Artificial Intelligence-Based Interactive Multimodal Interactive Storytelling 3D Scene Authoring Technology Development, Project Number: RS-2023-00227917, Contribution Rate: 100%) and the Korea Research Foundation's four, Brain Korea 21 (Project Name: Ultra-Distributed Autonomous Computing Service Technology Research Team, Project Number: 202003520005).

generative AI with prompt engineering. The classified sentences are converted into conditional sentences, and a C3Tree Model is generated from the conditional sentences. Then, a cause-effect graph is generated from the C3Tree Model. This method provides higher accuracy than general prompt-based sentence classification methods. It is also possible to address exceptions that algorithmic classification methods cannot handle.

Section 2 describes related research. Section 3 describes our research. Section 4 describes a case study. Section 5 describes the discussion. Finally, we describe the conclusions.

2. Related Research

2.1 Cause-Effect Graph Testing

A cause-effect graph [5] identifies the smallest unit of function in a requirements sentence and connects functions with cause-effect relationships. Each node has an Identity, as well as an AND, OR, and NOT relationship. The Identity is the result of False if the cause is True. NOT is the result is True if the cause is False. AND is the result is True if all causes are True. OR is the result is True if any one of several causes is True. **Fig. 1** shows the structure of a cause-effect graph.

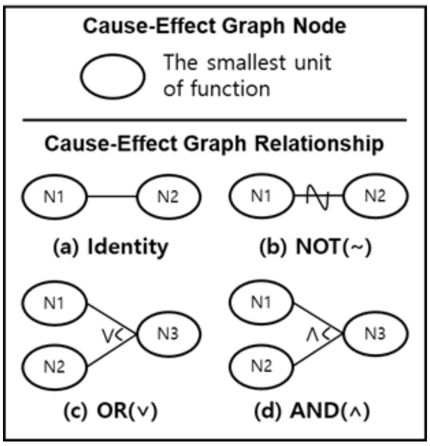

Fig. 1. Components of a cause-effect graph

2.2 C3Tree Model

The C3Tree Model is a model that shows the process of simplifying a natural language sentence [4]. This model can trace the process of sentence simplification, and the simplified sentence nodes can be easily mapped to the nodes of the Cause-Effect Graph. The node types of the C3Tree Model include the original text node (Sentence), complex-sentence node (Complex-Sentence), and simple-sentence node (Clause). The types of associations include positive condition (COND-P), negative condition (COND-N), AND (CONJ-AND), and OR (CONJ-OR) relationships. **Fig. 2** shows an example of the C3Tree Model and an example of generating a Cause-Effect Graph from the C3Tree Model.

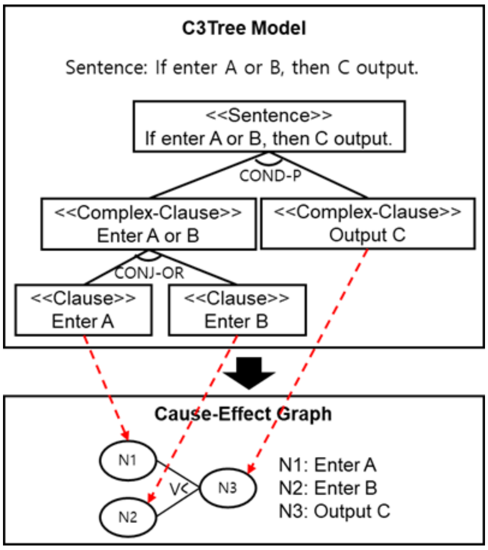

Fig. 2. Example of generating a cause-effect graph from the C3Tree Model

2.3 Prompt Engineering

Prompt engineering is a technique for designing and optimizing prompts for generative AI. We use prompting techniques to distinguish between sentence types and transform the sentence structure. For this purpose, we consider Chain of Thought (CoT) prompting [6] and Fukatsu prompting [7].

CoT presents each step when solving a complex problem and solves the problem in a logical order.

Fukatsu-style divides the prompt into Command, Constraints, Input, and Output. The Command and Constraints list outlines explicitly how to process the input statement. Input refers to the query that the user enters into the generative AI.

The output is the answer of the generative AI.

3. C3Tree Model Generation from Non-Conditional Requirement

We propose a method to automatically generate cause-effect graphs from non-conditional sentences that do not contain causes and effects. **Fig. 3** shows a method for identifying cause/effect relationships from non-conditional requirement sentences.

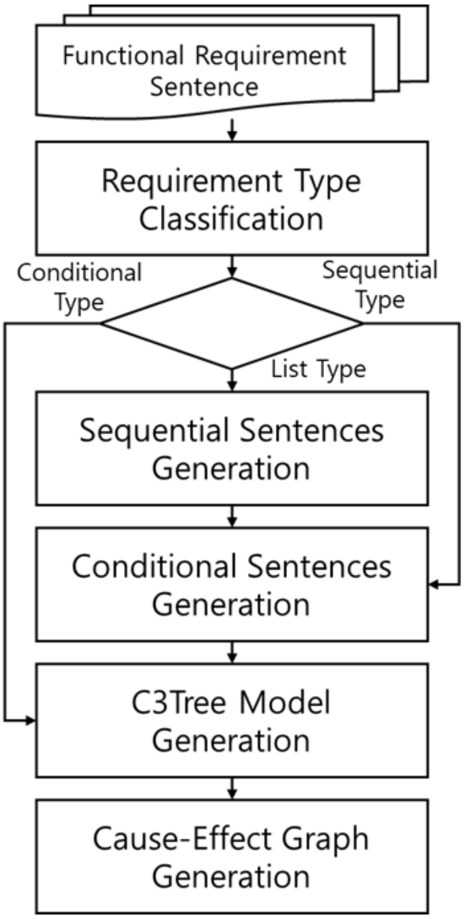

Fig. 3. Method for identifying cause/effect relationships from non-conditional requirement sentences

3.1 Requirement type classification

We analyze functional requirement specifications among many requirement specifications. We classify requirement sentences into three types.

The first type is a conditional sentence. This is our previous research. It is a sentence that contains a cause clause, a result clause, and a conjunction clause.

The second type is a listed sentence. This lists functional requirement sentences in a non-sequential manner. **Table 1** shows examples of listed sentences.

The third type is a sequential sentence. This sequentially lists functional requirement sentences. **Table 2** shows examples of sequential sentences.

Table 1. Part of the listed sentences

1. Memos written in the memo system are automatically and continuously saved to the local system at regular intervals (1 second). 2. Memos should be saved automatically without a separate save command from the user.

Table 2. Part of sequential sentences

1. The memo system requests the user's ID and password, connects to the server, and verifies the validity of the provided ID and password. 2. If the obtained ID/password is invalid, it re-requests the login information.

We classify sentences using CoT prompts and Fukatsu prompts on ChatGPT, Claude, and Gemini. **Tables 3** and **4** show examples of sentences for each prompt.

Table 3. Part of the CoT prompt

Question: " The memo system asks for the user's ID/password, connects to the server, and …(requirement sentence)" Distinguish this sentence into a list type or a sequential type. These three sentences describe the login function of the memo system. The second sentence is executed after the first sentence is executed...(description of classification method)

Table 4. Part of Fukatsu's prompt

#Command: - Enter the functional specification of the software. - …(command description) #Constraints: - The correlation between the input sentences is analyzed and divided into two types. - The first form is the list type. List sentences are simply elements that make up a function. - …(constraints description)

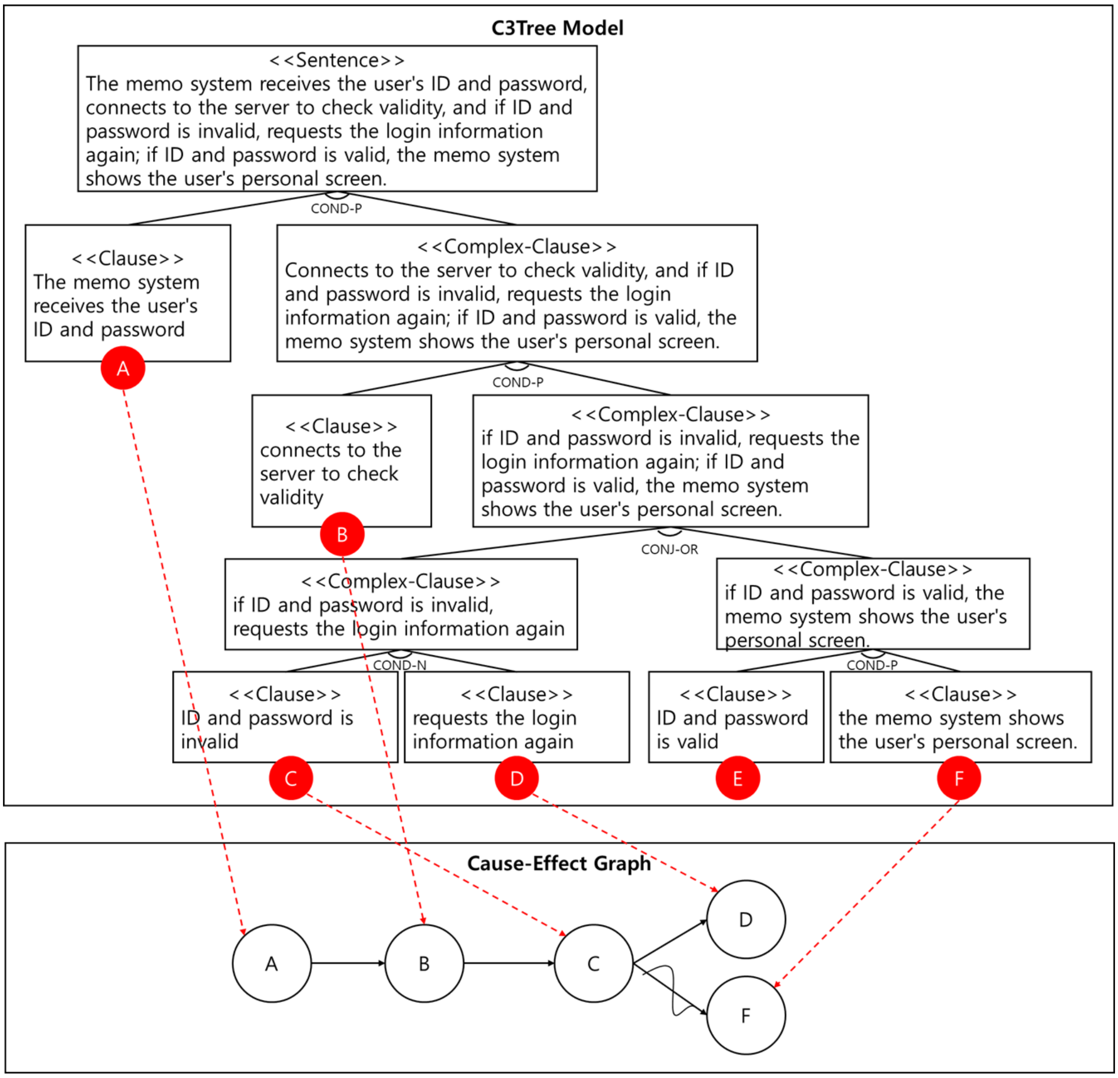

Fig. 4. Example of generating C3Tree Model and Cause-Effect Graph from conditional sentences

Tables 5 and 6 display natural language sentences and responses entered into the generative AI.

Table 5. Requirement Sentences (list type)

- Memos written in the memo system are automatically and continuously saved to the local system at regular intervals (1 second). Memos should be saved automatically without a separate save command from the user. - There should be no noticeable delay in the user's use of notes due to the saving of notes. Saved notes are saved to the local file system or local database.

Table 6. Sentences output by generative AI

List type

3.2 Generating sequential sentences from listed sentences

We use generative AI to transform listed sentences into sequential sentences. We integrate multiple sentences contained in a single function into a single sentence. We exclude non-functional requirement sentences. Finally, we re-divide the integrated sentences in order of their tasks. Table 7 presents the results of integrating the sentences listed in Table 5. Table 8 shows the sentences re-divided in order of their functions, excluding non-functional elements from the integrated sentences.

Table 7. Integrated sentences

Memos written in the memo system are automatically and continuously saved to the local system at regular intervals (1 second). The memos must be saved automatically without a separate save command from the user. There must be no noticeable delay in the user's use of the memos due to the saving process, and the saved memos are stored in the local file system or local database.

Table 8. Re-divided sentences (Sequential type)

1. Memos written in the memo system are automatically and continuously saved to the local system at regular intervals (1 second). 2. Saved memos are saved to the local file system or local database.

3.3 Generating conditional sentences from sequential sentences

We use generative AI to convert sequential sentences into conditional sentences. Conditional sentences are sentences that contain cause-and-effect relationships. All sequential sentences are sequentially combined into one sentence. Clauses with redundant meanings are combined into one clause. Clauses without subjects restore their subjects. **Table 9** shows examples of ordinal sentences. **Table 10** shows sentences in which sequential sentences are converted into conditional sentences.

Table 9. Sequential sentences

1. The memo system requests the user's ID and password, connects to the server, and verifies the validity of the provided ID and password. 2. If the obtained ID/password is invalid, it re-requests the login information. 3. If the ID and password are valid, the user's screen is displayed.

Table 10. Conditional sentence

The memo system receives the user's ID and password, connects to the server to verify their validity, and if the ID and password are invalid, it requests the login information again. If the ID and password are valid, it displays the user's screen.

3.4 Generating C3Tree Model and Cause-Effect Graph from Conditional Sentences

Fig. 4 shows an example of generating a C3Tree Model from a conditional sentence and generating a cause-effect graph from the C3Tree Model. The top node of the C3Tree Model is the original conditional sentence. The bottom node is a simplified sentence. The simplified sentence is converted into a node of the cause-effect graph, and the converted nodes are connected by identity, AND, OR, and NOT combinations.

4. Case Study

This chapter develops a test case generation environment from unconditional requirement sentences and generates related models. **Fig. 5** shows the components of the proposed environment.

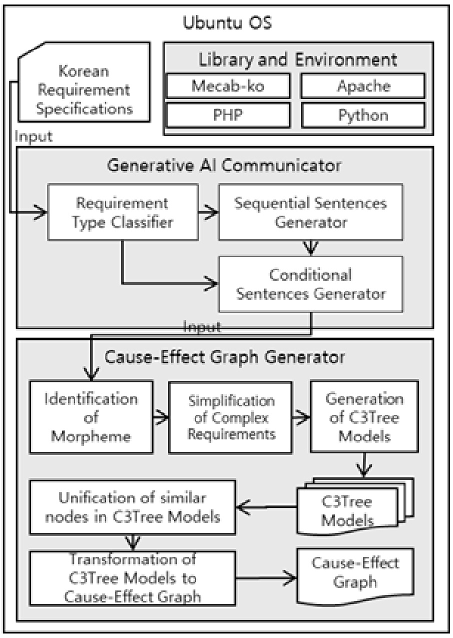

Fig. 5. Design of the experimental environment

Fig. 6 shows the result of generating a C3Tree model by inputting a conditional requirement sentence.

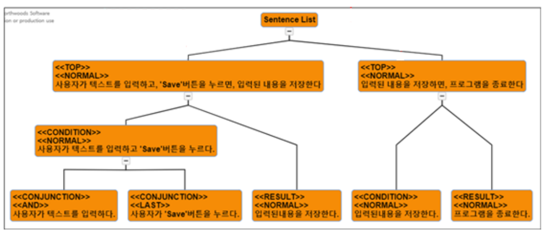

Fig. 6. Results of generating the C3Tree model from the requirement sentence

5. Discussion

We apply CoT and Fukatsu techniques to ChatGPT, Claude, and Gemini to classify 50 features and measure the generative AI with the highest sentence classification accuracy. **Table 11** shows the sentence classification accuracy of generative AI. As a result, we can confirm that the combination of ChatGPT and CoT has the highest accuracy.

Table 11. Sentence classification accuracy of generative AI

	CoT	Fukatsu
ChatGPT	94%	88%
Claude	76%	76%
Gemini	76%	64%

Table 12 shows the classification accuracy of ChatGPT for list and sequential types. It can be seen that the sentence classification accuracy differs depending on the prompting technique.

Table 12. Classification-specific generation accuracy of ChatGPT

	CoT	Fukatsu
List type	100%	71%
Sequential type	90%	100%

6. Conclusions

As part of the automatic test case generation method from Korean natural language requirements, we propose a method to automatically convert natural language sentences that do not include causes and effects into conditional sentences, automatically convert conditional sentences into C3Tree Models, and automatically generate cause-effect graphs from C3Tree Models.

We applied prompt engineering techniques to generative AI to classify sentences into listed types and sequential types, and compared the classification accuracies. As a result, the method of combining ChatGPT and CoT to classify sentences and generate cause-effect graphs showed the highest accuracy.

As a future study, we plan to investigate how to incorporate functions that are commonly understood but not explicitly mentioned in requirement sentences into cause-effect graphs.

References

[1] W. C. Jung, “Study on Software Security Testing: Focused on Methodology,” *Convergence Science and Technology Society Research*, vol.3, no.2, pp.21-26, 2024.

[2] J. K. Ahn and S. Kim, “A Study on Software Supply Chain Risk Management Policies and Techniques Suitable for Korean Military,” *Journal of The Korea Institute of Information Security & Cryptology*, vol.35, no.1, pp.135-155, 2025.

[3] C. Jeon and S. Ryu, “An Analysis of Trends in Natural Language Processing Research in the Field of Science Education,” *Journal of the Korean Association for Science Education*, vol.44, no.1, pp.39-55, 2024.

[4] W. S. Jang and R. Y. C. Kim, “Automatic Cause–Effect Graph Tool with Informal Korean Requirement Specifications,” *Applied Sciences*, vol.12, no.8, pp.1-16, 2022.

[5] G. E. Mogyorodi, “Requirements-Based Testing-Cause-Effect Graphing,” *Software Testing Services*, 2005.

[6] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le and D. Zhou, “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models,” in *Proc. of 36th Conference on Neural Information Processing Systems*, 2022.

[7] Japanese AI Prompting, [Online]. Available: https://macrolingo.com/japanese-fukatsu-shunsuke-ai-prompting-methods/