MAPE-K Loop-based Goal Model Generation Using Generative AI

Hiroyuki Nakagawa
Osaka University
51-5 Yamadaoka, Suita, Osaka, 565-0871, Japan
nakagawa@ist.osaka-u.ac.jp

Shinichi Honiden
National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, 101-8430, Japan
honiden@nii.ac.jp

***Abstract*—Goal modeling constitutes a systematic modeling of representation, specifically crafted to capture and depict stakeholders' intentions, desires, and objectives. Notwithstanding its importance, describing the entire scope of goals to be achieved remains a complex task. To address this challenge, we propose a semi-automatic goal model generation process. The feature of the process lies in its use of a generative AI based on the MAPE-K loop mechanism. We conducted two case studies that built goal models using this proposed process. The results demonstrate that our process, grounded on the MAPE-K loop mechanism, efficiently aids goal model construction.**

***Index Terms*—Goal models, requirements analysis, large language models, generative AI, MAPE-K loop mechanism.**

I. Introduction

Goal-oriented requirements analysis is an effective process for eliciting, analyzing, and documenting the goals and requirements of a system. An artifact of the goal-oriented requirements analysis, i.e., a goal model, is an explicit analytical model that represents the interrelationships between goals. Although goal-oriented requirements analysis is a powerful analysis method, the construction of an accurate and sufficient goal model is still difficult. One of the most critical challenges is the incompleteness of the goal model, characterized by gaps or missing elements. This implies that the goal model does not fully capture or address all the relevant goals or requirements, a problem that arises from the lack of stakeholder input and inadequate analysis or requirements gathering.

To counter this problem, we utilize large language models (LLMs). LLMs and generative AI have revolutionized the field of natural language processing, demonstrating remarkable performance in various language-related tasks. However, as described by Cámara et al. [5], responses from generative AI can contain errors in the semantics of the domain being modeled. They also assert that the domain influences the outcome, meaning the more generative AI knows about a domain, the closer to the correct result it processes.

In this study, we propose a comprehensive goal model generation process based on a large language model. This process employs generative AI within the MAPE-K loop mechanism, known as an autonomic component for implementing self-adaptive systems. Activities within the MAPE-K loop mechanism systematically check for errors and variations in responses from a generative AI-based tool. Furthermore, the method aims to construct a goal model with no missing goal descriptions by assigning multiple expert roles to the generative AI tool and soliciting its opinions. The proposed process applies the MAPE-K loop activities incrementally to refine a goal model.

The main contributions of this study include:

- Proposal of a semi-automatic goal model generation process using a generative AI tool within the MAPE-K loop mechanism to continually refine the goal model.
- Provision of empirical evidence showing that the proposed process can generate a goal model across various domains, taking non-functional requirements into account.
- Summary of findings from evaluation results to illustrate how a goal model can be effectively generated using a generative AI tool.

The remainder of this paper is organized as follows: Section II describes the background of this study, particularly explaining the complexities of goal modeling, generative AIs, and the MAPE-K loop mechanism. Section III details the proposed goal model generation process using generative AI. Section IV evaluates the proposed process through two case studies, while Section V discusses the proposed process based on the results obtained from these case studies. Finally, Section VI concludes the paper and outlines potential directions for future work.

II. Background

A. Goal Modeling

Goal modeling is a requirements analysis technique that identifies goals from requirements and establishes relationships among them. It offers a systematic analysis method using AND/OR-refinement links. The outcome of goal modeling is a goal model, a diagrammatic representation of all the goals, tasks, and objectives that a system under development needs to achieve. The goal model is used to express system objectives and guide the process of deriving and refining system requirements. However, constructing goal models remains challenging for several reasons [2] [6]:

- The presence of *incomplete* or *inconsistent goals*. Stakeholders and requirements engineers may struggle to express their goals comprehensively or consistently.

- *Ambiguity* and *subjectivity* involve in defining and capturing goals. Stakeholders often possess diverse perspectives and varying levels of expertise, leading to discrepancies in goal understanding and representation.
- *Interdependencies* and potential *conflicts* among goals in a goal model. Achieving one goal may conflict with another [11]. Identifying and managing these interdependencies require careful analysis and prioritization of goals.
- Goal models increase in size when analyzing requirements concerning various aspects [13]. As the number of stakeholders, goals, and interactions grow, managing the *complexity* and *scalability* of the goal modeling process becomes daunting.

B. Large Language Models and Generative AI

Large language models (LLMs) and generative AI are sophisticated deep learning models designed to understand and generate natural language. They have significantly pushed the boundaries of natural language processing, enabling tasks such as language translation, text generation, sentiment analysis, question answering, and information retrieval. Representative LLMs include ChatGPT [14], Claude [1], Alpaca [15], LaMDA [8], Copilot [7], and Midjourney [12].

Prompt engineering, which offers explicit instructions to LLMs, is becoming a popular method to guide them towards generating more accurate and contextually relevant responses. It involves designing and formulating prompts or instructions to steer LLMs towards desired outputs, enhancing their capabilities and aligning them with specific tasks or objectives. Notable prompt design strategies include:

- Instruction-based Prompts: These prompts provide explicit instructions to LLMs, specifying the desired format, context, or style of the response. They are particularly useful for tasks requiring specific output structures, such as filling in the blanks or generating code snippets.
- Example-based Prompts: These prompts present LLMs with relevant examples that demonstrate the desired behavior. By providing both positive and negative examples, the model can learn to generalize and generate appropriate responses.
- Multi-turn Conversation Prompts: These prompts simulate conversational contexts by providing a sequence of dialogue turns, enabling LLMs to generate responses that maintain coherence and context continuity.

Various studies including [4] are exploring the potential advantages of LLMs and generative AI in diverse research domains, such as software engineering, where the primary goals is to generate code [3] [16]. Some studies are focusing on models used in the software development. For instance, Cámara et al. [5] conducted experiments to generate UML models using ChatGPT. While code and design models, including UML models generated by LLMs, may be syntactically correct, to construct a semantically correct software system, a correct requirements model is also needed.

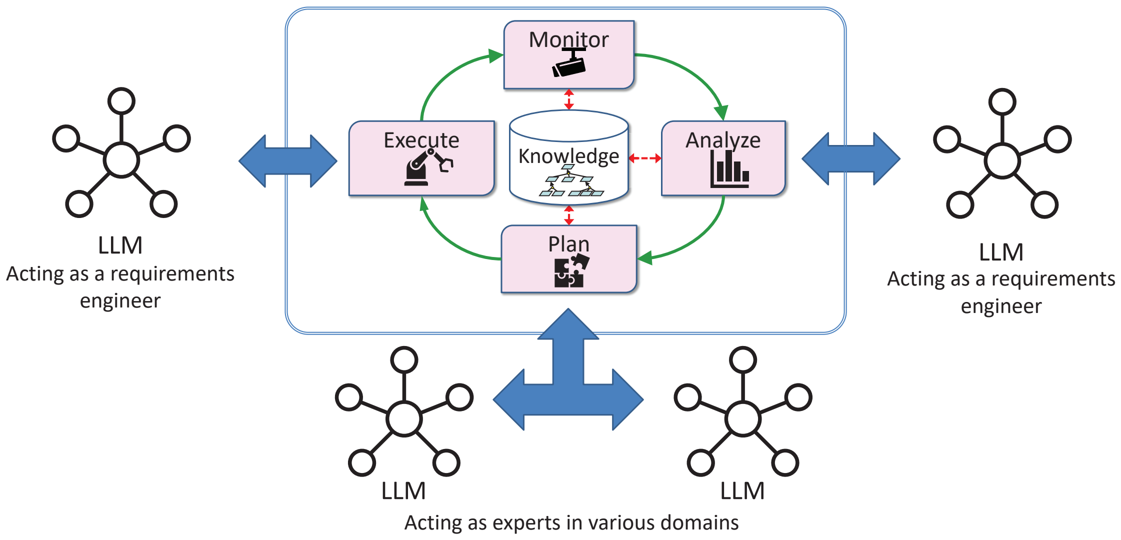
C. MAPE-K Loop Mechanism

The MAPE-K loop [9] was originally developed for autonomous software systems, such as self-adaptive systems and autonomic computing. The mechanism aims to control a software system by continuously executing four steps, known as *monitor, analyze, plan,* and *execute*, in a loop. The target system is monitored to determine whether problems have occurred using logs and sensors at the monitor step. If a problem is identified during the analyze step, the mechanism attempts to determine the cause of the problem. subsequently, in the plan step, actions are formulated to address the problem based on the analysis results. Finally, at the execute step, the planned actions are carried out. The MAPE-K loop mechanism then repeats these activities, beginning by monitoring the outcomes of the execute step. The MAPE-K loop also includes a complementary *knowledge* base that manages data shared among the four steps. The shared knowledge can contain information such as historical logs.

III. Goal modeling using LLMs

The objective of this study is to effectively leverage Large Language Models (LLMs) for the construction of goal models. More specifically, we consider LLMs as domain experts and utilize them within the MAPE-K loop-based process. Figure 1 depicts our proposed goal model generation process, consisting of four activities based on the MAPE-K loop mechanism: *monitor*, *analyze*, *plan*, and *execute*. This process also incorporates a storage component acting as the *knowledge* base. MAPE-K loop activities interact with the LLMs.

Algorithm 1 describes how the proposed process constructs a goal model. The initial goal model is generated by an LLM at the beginning of the process and stored to the knowledge base. The process then proceeds with the MAPE-K loop activities. *1) Monitor:* As the first activity of the MAPE-K loop, we examine the current goal model stored in the knowledge base. *2) Analyze:* This activity is responsible for validating the AND/OR-refinement relationships within the goal model. This validation is conducted by an LLM acting as a requirements engineer. *3) Plan:* This activity involves planning to add a viewpoint to the goal model. To identify goals that pertain to this viewpoint and should be included in the goal model, the activity seeks opinions from LLMs, acting as experts for the viewpoint. Possible viewpoints include non-functional requirements such as safety, security, reliability, performance, maintainability, and usability. The opinions obtained from the LLMs' response are stored in the knowledge base. *4) Execute:* Based on the opinions obtained during the planning activity, this activity incorporates goals into the goal model. This addition is performed by an LLM acting as a requirements engineer. *5) Knowledge:* The knowledge base stores shared data that all the MAPE activities can access. In the proposed process, this base stores the current goal model and opinions obtained during the planning activity. The goal model is continuously updated through the MAPE activities. Each activity references the goal model in the knowledge base when it seeks information from LLMs.



Monitor
Execute
Knowledge
Analyze
Plan
LLM
Acting as a requirements engineer
LLM
Acting as a requirements engineer
LLM
LLM
Acting as experts in various domains

Fig. 1. An overview of the proposed goal model generation process. The goal model stored in the Knowledge base is incrementally refined by the MAPE-K loop activities.

Algorithm 1 Goal model generation.

```
1: Create an initial goal model and store it as gModel in the Knowledge base.
2: repeat
3:     Observe the current gModel in Knowledge. //Monitor
4:     Validate the AND/OR-refinement relationships described in gModel by consulting an LLM acting as a requirements engineer. //Analyze
5:     Collect opinions related to non-functional requirements from experts in various domains, all of whom are acted by LLMs. //Plan
6:     Incorporate opinions obtained during the planning activity into gModel by consulting an LLM acting as a requirements engineer. //Execute
7: until All of viewpoints are incorporated into gModel.
```

To ensure a systematic interaction with LLMs, the generation process uses a template prompt, shown in Listing 1. This template first defines a role of the LLM, such as a requirements engineer or security specialist, to ascertain the appropriate source for obtaining opinions (*Role definition* in Listing 1). The template then outlines instruction corresponding to individual MAPE activities (*Instruction*). The *constraints description* field is utilized to explain the current goal model and constraints on anticipated responses.

Listing 1. A template prompt used by the proposed process for interacting with LLMs.

```
<Role definition>
<Instruction>

#Consraints
<Constraints description>
```

Listing 2. A query for generating the initial goal model for a library management system.

```
I want you to act as a requirements
    engineer.
Please construct a goal model in accordance
    with the following constraints.

# Constraints
- The goal model should contain general
    requirements for a library management
    system.
- Goals in the goal model should be
    described in a numbered list.
- AND/OR-refinement links should be used to
    refine requirements into subgoals.
```

Listing 3. A query for requesting maintenance opinions of the cleaning robot from a cleaning robot engineer role-played by chatGPT. The query corresponds to the planning activity of the MAKE-K loop mechanism.

```
I want you to act as a cleaning robot
    engineer.
Please provide opinions that meet the
    following constraints.

# Constraints
- The opinions should be described in a
    numbered list.
- The opinions should concern the
    maintenance of the cleaning robot.
```

IV. Case Studies

We explore the following research questions related to the goal model generation:

- **RQ1:** *Can generative AI produce accurate goal models?*
- **RQ2:** *Can the MAPE-K loop efficiently construct a goal model using a generative AI-based tool?*
- **RQ3:** *Can various experts played by generative AI contribute to expand a goal model?*

TABLE I
OPINIONS ACQUIRED AT THE PLAN ACTIVITY. IN THE EXPERTS FIELD, THE LABEL **L** REPRESENTS A *librarian*. OTHER LABELS REPRESENT EXPERTS AS FOLLOWS: **S** FOR *system engineer*, **G** FOR *garbage collector*, **C** FOR*cleaning robot developer*, **Sec** FOR *security specialist*, AND **H** FOR *human-centered design specialist*.

		Library system		**Cleaning robot**	
Performance	Expert	**L**	**S**	**G**	**C**
	Total	8	10	10	10
	Correct	5	3	8	7
	Precision	0.63	0.30	0.80	0.70
Maintenance	Expert	**L**	**S**	**G**	**C**
	Total	15	15	10	10
	Correct	9	13	9	10
	Precision	0.73	0.87	0.90	1.00
Security	Expert	**L**	**Sec**	**G**	**Sec**
	Total	10	10	10	10
	Correct	9	10	8	4
	Precision	0.90	1.00	0.80	0.40
Human centered design	Expert	**L**	**H**	**G**	**H**
	Total	10	10	10	10
	Correct	10	9	8	8
	Precision	1.00	0.90	0.80	0.80

TABLE II
CHANGES IN THE NUMBER OF GOALS IN THE TWO CASE STUDIES.

		Library system	**Cleaning robot**
Initial	Total goals	30	39
Performance	Goals added	32	25
	Goals deleted	0	0
	Total goals	62	64
Maintenance	Goals added	10	21
	Goals deleted	0	0
	Total goals	72	85
Security	Goals added	23	28
	Goals deleted	5	0
	Total goals	90	113
Human centered design	Goals added	33	51
	Goals deleted	0	7
	Total goals	123	157

To answer the research questions, we applied the proposed process to two sample systems, using them as case studies.
Case 1: Library management system. The first system is a library management system, frequently used as a sample system in the goal modeling domain, such as exemplified in [10]. The potential experts in this case include *librarians* and *system engineers*.
Case 2: Cleaning robot. The second case study focuses on the development of a cleaning robot. Compared to Case 1, goal model development in this context is less prevalent on the web, implying that LLMs may not find enough knowledge to construct the goal model. The likely experts in this case are *garbage collectors* and *cleaning robot developers*.

In these case studies, we employed ChatGPT-4 as a generative AI tool (or LLM). We initially generated the goal models by posing questions like the one illustrated in Listing 2. We applied the proposed process to the two systems, sequentially adding each non-functional requirements in this order: *performance, maintenance, security,* and *human-centered design*. Listing 3 presents an example prompt used during the planning activity in Case 2.

The results of the case studies are provided in Tables I and II. Table I presents the accuracy of the opinions obtained during the planning activity, indicating that most opinions can serve as valuable contributions to the expansion of the goal model. We assessed some opinions as incorrect because they were overly general and crossed the boundaries. For example, when we sought opinions on performance, the LLMs returned opinions that, while related to performance, also pertained to security or maintenance. Table II displays the changes in the number of goals. In both case studies, new goals were integrated based on the opinions related to non-functional requirements during each MAPE-K loop.

V. DISCUSSION

In light of our case studies, we respond to our research questions and discuss the applicability and limitation of the proposed process.

A. Response to Research Questions

RQ1 (*Can generative AI produce accurate goal models?*): The Results of the case studies indicate that generative AI is capable of creating a goal model for a given target domain. In both case studies, ChatGPT produced satisfactory initial goal models that effectively utilized AND/OR-refinement links. However, goal models generated by LLMs tend to express more generalized concerns, with the majority of goals within the models representing functional requirements and little mention of nonfunctional requirements. Another concern pertains to the maintenance of the goal model. When refining a goal model using a generative AI tool, it is crucial to confirm consistency and detect any missing goals before and after conversion.

RQ2 (*Can the MAPE-K loop efficiently construct a goal model using a generative AI-based tool?*): The outcomes from two case studies suggest that the MAPE-K loop effectively aids in the construction of the goal model. Our proposed process handles goal models within the four activities of the MAPE-K loop mechanism, using a loop to incorporate goals about non-functional requirements. The goal model generated by the proposed process appears to contain adequate goals. However, as the process incrementally adds goals based on feedback obtained in the plan activity, the structure of the goal model tends to become flattened. To enhance the depth of refinement for the goal model, refactoring of the goal model is recommended.

RQ3 (*Can various experts played by generative AI contribute to expand a goal model?*): The results of two case studies indicate that the MAPE-K loop effectively employs LLMs, which act as various experts for goal model generation. The analyze and execute activities utilize LLMs as requirements engineers to review the current goal model and incorporate goals. The plan activity solicits opinions from LLMs as experts in various domains. These opinions contribute significantly to goal addition. Experimental results show that most feedback obtained in the plan activity is correct, but some opinions are generalized and cross the scope boundary. At present, we verify the opinions, but in the future, LLMs should check the validity of feedback.

B. Applicability and Limitation

For the construction of an adequate goal model, LLMs must possess extensive knowledge about a target system and its domain. If the target system is pioneering and highly advanced, the process may not generate an adequate goal model. Similarly, the goal model obtained by the proposed process is a generic goal model, meaning the process can acquire goals generally known. We need to add case-specific goals into the goal model after the generation.

VI. Conclusions

This paper presented a goal model generation process using generative AI. The process progressively integrates goals into the goal model based on the MAPE-K loop mechanism. The process also employs LLMs as domain experts for soliciting opinions and constructing an accurate goal model. Results from two case studies showed that the process successfully incorporate goals concerning multiple non-functional requirements.

We identified the following significant directions for future studies. Firstly, as the current process is semi-automated — meaning it contains human decision-making steps — we aim to enhance the process to achieve full automation. Our aim will be to delegate decision-making to LLMs. The next step will be to work on developing a process for integrating case-specific goals after the initial goal model generation. To interact with a generative AI tool, the current process manages a text list as a goal model. Future work will also include goal model refactoring and visualization.

Appendix

A portion of the goal model generated in Case 1 (Library management system) is illustrated in Listing 4.

Listing 4. A Portion of the goal model for a library management system, which was generated during the first case study.

```
Goal 1: Develop a user-friendly Library
     Management System (LMS)
1.1: AND-refinement
1.1.1: Ensure system is easy to navigate
1.1.2: Provide user manuals and help
     resources
1.1.3: Implement a simple and clean UI
Goal 2: Efficiently manage library
     resources
2.1: AND-refinement
2.1.1: Track and manage books inventory
2.1.1.1: AND-refinement
2.1.1.1.1: Catalogue books
2.1.1.1.2: Track book locations
2.1.1.1.3: Manage book checkout and return
2.1.2: Handle reservation requests
2.1.2.1: OR-refinement
2.1.2.1.1: Handle reservations online
2.1.2.1.2: Handle reservations at library
     counter
2.1.3: Maintain records for periodicals,
     magazines and newspapers
2.1.4: Manage digital resources
2.1.4.1: AND-refinement
2.1.4.1.1: Manage e-books
2.1.4.1.2: Manage online databases
Goal 3: Provide advanced search
     capabilities
3.1: AND-refinement
3.1.1: Allow search by various parameters (
     title, author, subject, ISBN, etc.)
3.1.2: Implement search filters
3.1.3: Allow saving and reusing previous
     searches
Goal 4: Enhance user services
4.1: AND-refinement
4.1.1: Enable users to check account status
4.1.2: Allow users to renew and reserve
     books online
4.1.3: Provide notifications for due dates,
      overdue books, reservations, etc.
4.1.3.1: OR-refinement
4.1.3.1.1: Send notifications by email
4.1.3.1.2: Send notifications by SMS
Goal 5: Ensure system security
5.1: AND-refinement
5.1.1: Protect user data privacy
5.1.2: Implement secure login and access
     control
...
```

References

[1] Anthropic: Claude. https://www.anthropic.com/index/introducing-claude (2023)

[2] ao, S.A., Insfran, E., de Guevara, F.G.L., Fernández-Diego, M., Cano-Genoves, C., Pereira de Oliveira, R.: Assessing the effectiveness of goal-oriented modeling languages: A family of experiments. Information and Software Technology **116**, 106171 (2019)

[3] Barke, S., James, M.B., Polikarpova, N.: Grounded copilot: How programmers interact with code-generating models. arXiv:2206.15000 (2022)

[4] Borji, A.: A categorical archive of ChatGPT failures. arXiv:2302.03494 (2023)

[5] Cámara, J., Troya, J., Burgueo, L., Vallecillo, A.: On the assessment of generative ai in modeling tasks: an experience report with chatgpt and uml. Software and Systems Modeling (2023)

[6] Cheng, B.H.C., Atlee, J.M.: Research directions in requirements engineering. In: 2007 Future of Software Engineering. pp. 285–303. FOSE '07, IEEE CS (2007)

[7] GitHub, I.: Copilot. https://github.com/features/copilot/ (2023)

[8] Google: Lamda: our breakthrough conversation technology. https://blog.google/technology/ai/lamda/ (2021)

[9] IBM: An architectural blueprint for autonomic computing third edition

[10] van Lamsweerde, A.: Requirements Engineering — From System Goals to UML Models to Software Specifications. Wiley (2009)

[11] van Lamsweerde, A., Letier, E.: Handling obstacles in goal-oriented requirements engineering. IEEE Transactions on Software Engineering (TSE) **26**(10), 978–1005 (2000)

[12] Midjourney: Midjourney. https://www.midjourney.com/home/ (2023)

[13] Nakagawa, H., Ohsuga, A., Honiden, S.: A goal model elaboration for localizing changes in software evolution. In: Proc. of RE'13. pp. 155 – 164. IEEE CS (2013)

[14] OpenAI: Chatgpt. https://openai.com/blog/chatgpt (2023)

[15] Stanford Alpaca project: Stanford alpaca: An instruction-following LLaMA model. https://github.com/tatsu-lab/stanford_alpaca (2023)

[16] Vaithilingam, P., Zhang, T., Glassman, E.L.: Expectation vs. experience: Evaluating the usability of code generation tools powered by large language models. In: CHI EA '22. ACM (2022)