Towards Automatic Generation of iStar Models Using ChatGPT

Yoshitake Hirabayashi[1,*,†], Motoshi Saeki[1,†]

[1]*Nanzan University, Nagoya 466-8673, Japan*

Abstract

The goal-oriented requirements analysis method is useful in requirements analysis; however, due to the lack of guidelines and methodologies, generating goal models can be challenging, especially for beginners. Therefore, this research aims to support the creation of iStar models, one of the goal-oriented requirements analysis methods, using ChatGPT. In this paper, our approach consists of two steps: generating iStar models without syntactic constraint violation and modifying iStar models. For the generation of iStar models without syntactic constraint violation, efforts were made in designing input prompts, describing syntactic rules, and refining the generation procedure. In the modification of iStar models, the focus was placed on the "Bad Smells" in iStar models, defined by us, and correction candidates were categorized into omissions, ambiguities, and inconsistencies, with modifications made by referencing specific examples. As a result, our technique successfully generated iStar models without syntactic constraint violation and improved the quality of iStar models through modifications. However, challenges remain, such as the overly generic nature of the generated iStar models, the inability to represent conflicts between elements, and insufficient detail.

Keywords

Goal Model Generation, Bad smell, ChatGPT, Self-Instruct

1. Introduction

One of the requirements analysis methods that aim at requirements elicitation is Goal-Oriented Requirements Analysis (GORA). This method takes customers' requirements as goals, iteratively decomposes and elaborates on them to derive system or software requirements. In GORA, elicited goals and their relationships are represented in the form of graphs, called a goal model. However, due to the lack of guidelines and methodologies for goal analysis and refinement, generating models is extremely challenging for beginners, and there is a possibility of producing low-quality goal models. In recent years, research has increased that leverages artificial intelligence technologies and generative AI to assist beginners in generating goal-oriented requirements analysis (GORA). Danilo et al. conducted research using a chatbot that employs natural language processing (NLP) as an assistant for KAOS modeling to support novice requirements engineers in eliciting requirements [1]. Marques et al. investigated how large language models (LLMs), particularly ChatGPT, can be utilized in software engineering, focusing on their role in requirements elicitation, documentation, and validation, based on several studies [2]. Chen et al. examined the extent of ChatGPT's knowledge of goal-oriented requirements analysis methods and its ability to generate GRL [3]. Nakagawa et al. proposed a semi-automatic goal generation process that utilizes LLMs and the MAPE-K loop in the goal model generation process. This approach uses LLMs as domain experts [4].

As far as we know, the quality of iStar models generated by AI remains low in all existing studies. Therefore, this research aims to automatically generate high-quality iStar models using ChatGPT-4o [5], focusing specifically on iStar within the context of GORA.

iStar'24: The 17th International i Workshop, October 28, 2024, Pittsburgh, Pennsylvania, USA*

*Corresponding author.

†These authors contributed equally.

EMAIL: m23se003@nanzan-u.ac.jp (Y. Hirabayashi); saekimot@gmail.com (M. Saeki)
ORCID: 0000-0001-7116-9338 (M. Saeki)

2. Approach

As highlighted in the research by Chen et al. [3], while generative AI possesses a certain level of knowledge related to goal-oriented requirements analysis (GORA), there is a risk that the AI may generate models with syntactic constraint violations and/or produce low-quality models that, while syntactically correct, are still suboptimal. To address these problems, first, we take an approach where ChatGPT generates an iStar model, and then we detect and correct low-quality or erroneous parts in the generated model. Therefore, in this paper, we adopt the following two steps when modeling iStar using ChatGPT.

1. The generated iStar model should avoid as many syntactic constraint violations as possible. To achieve this, we will develop prompt designs that include constraints in the prompt for generating an iStar model. We call these resulting prompts initial input prompts in this paper.
2. Although an iStar model generated using our initial input prompt does not contain syntactic constraint violations, it may have parts of lower quality such as ambiguity, omissions, inconsistency, etc. We use the concept of "Bad smells" [6] to detect and modify them.

2.1. Initial input prompt

Based on the research by Chen et al. [3], the basic input prompts are defined by dividing them into a single sentence, domain paragraph, and syntactic explanation. In this paper, in addition to these three prompts, we input a context, which is generally considered to improve the quality of outputs from ChatGPT, along with a text file containing the syntactic definitions of iStar 2.0. The syntactic definitions describe the constraints on which links between iStar elements are permissible, as well as the constraints defined in piStar[7]. The constraints defined in piStar were described with constraints, incorrect examples, and correct examples to make them easier for ChatGPT to understand. Therefore, the initial prompt included 1) context, 2) single sentence, 3) domain paragraph, 4) syntactic explanation, 5) syntactic constraints, 6) incorrect example, and 7) correct example. The actual prompt is explained in Section 3.1. The model generation process starts with entering the initial prompt, followed by sequentially providing instructions 1 through 4 to ChatGPT.

1. Create each actor and one goal for each actor.
2. Generate the necessary elements (goals, tasks, resources) to achieve the goals created in step 1, and connect them with appropriate links.
3. Identify the required "qualities" from the initial input prompt, create them for the appropriate actors, and link the elements related to those qualities.
4. For each element, if there are dependencies where other actors need to be involved, describe those dependencies.

The reason for adopting this procedure is that if the entire iStar model is generated directly from the input prompt, due to the constraints of the JSON format, elements need to be described first, and then dependencies and links are created based on those elements. This can lead to the violation of syntactic rules. Additionally, if the entire iStar model is generated first and then the syntactic rules are input to correct violations, the corrections often cause other parts of the model to violate the rules, leading to an endless loop of corrections. It is also worth noting that the quality of the iStar model remains consistent regardless of the method used, whether it follows the step-by-step procedure, generates the entire model at once, or corrects the syntax after generation.

2.2. Modification of the iStar model output by ChatGPT

The iStar models generated by ChatGPT, while not syntactically incorrect, may still produce low-quality models. In particular, the quality of the content within the elements is influenced by the input prompt, which can result in omissions, ambiguities, and inconsistencies. Therefore, this paper focuses on the Semantic Bad Smells, as defined by us, among the various Bad Smells [6]. Semantic Bad Smells refer to issues related to the content descriptions of iStar elements, and we have defined five types of such Bad Smells. Detection of each Semantic Bad Smell was attempted by focusing on the similarity of element descriptions, but the accuracy was not very high. Furthermore, in this paper, we revisited the concept of Semantic Bad Smells, categorizing those that cause omissions, ambiguities, and inconsistencies, and added new types of Semantic Bad Smells. Table 1 shows the Semantic Bad Smells used in this paper.

For the detection and modification of Bad Smells, we adopted a method where ChatGPT generates its own modification prompts, based on the Self-Instruct approach by Wang et al. [8]. This approach was chosen because ChatGPT may be able to identify areas that humans might find difficult to judge as Semantic Bad Smells. Additionally, it was expected that this method could detect additional Bad Smells beyond those defined by us. In the Self-Instruct method used in this paper, the input consists of a prompt for self-generation and a CSV file. In the CSV file, the input example is an iStar model (JSON) containing a Bad Smell, and the output example includes the locations of the Bad Smell, the reasons for it, and the corrected iStar model. The following explains the content of each description. The input prompt is: "The input CSV file contains instructions (prompts), inputs, and outputs for detecting bad smells in iStar 2.0. Refer to this CSV file to self-generate prompts for bad smell detection with iStar 2.0 and modify the model accordingly." The instructions are: sentences to modify input examples; input examples: iStar models with bad smells; output examples: the iStar model with identified bad smell areas, their reasons, and modifications. ChatGPT generates the instruction and output examples using Self-Instruct. These will be used to perform instruction tuning on ChatGPT.

3. An Example Applied to the Inventory Problem at a Liquor Store

An Example was conducted using the inventory problem of a liquor store to evaluate the effectiveness of the approach proposed in this paper. The inventory problem used here is an actual problem studied by university students learning requirements engineering. Next, within the same thread, a prompt was entered to detect and correct Bad Smells using the Self-Instruct approach on the output result. The outcome of this process was considered the example result.

3.1. The Example procedure for the input prompt and its Results

In this example, the initial step involved providing ChatGPT-4o with an input prompt that included all elements: context, a single sentence, a domain paragraph, and a syntactic explanation. Additionally, a text file containing the syntactic definitions of iStar 2.0 was provided. Table 2 shows the context, single sentence, and a part of domain paragraph. The syntactic explanation was derived from the iStar 2.0 Language Guide [9], where the iStar model was described using piStar, and unnecessary parts for generating the iStar model, such as positional information, were removed from the JSON file. Regarding the text file containing the syntactic definitions, experiments were conducted to evaluate how well ChatGPT adhered to the syntactic rules when provided with only the constraints, constraints with incorrect examples, and constraints with both correct and incorrect examples. The results showed that when only the constraints or constraints with incorrect examples were provided, no syntactic constraint violations occurred. However, when both correct and incorrect examples were provided alongside the constraints,

Table 1
Semantic Bad Smells

Category	Bad Smells	Grounds
Omissions	An element necessary to achieve a specific goal is missing.	The omission of elements necessary to achieve a goal hinders the fulfillment of the requirements.
Ambiguities	The descriptions between parent and child nodes as a graph in an iStar model are semantically the same.	The refinement between parent and child nodes is not done and it is unclear whether and how the parent nodes are refined.
Inconsistency	The descriptions of “dependum” and “dependerElmt” are not so semantically related to each other.	The reason why the actor having “ dependerElmt ” requires “ dependum ” is unclear.
	The descriptions of “dependum” and “dependeeElmt” are not so semantically related to each other.	It is unclear how the actor having “ dependeeElmt ” can achieve or realize “ dependum ” .
	The descriptions between parent and child nodes as a graph in an iStar model are not so semantically relatedto each other.	The refinement between parent and child nodes is inappropriate and semantically unclear.
	The links of “is-a” or “participates-in” relationships connect from abstract actors to concrete actors.	The direction of arrows, such as for inheritance, is reversed, altering the meaning of the iStar model.

Table 2
Input Prompt

Context	You are an excellent requirements analyst. Now you are going to model customer requirements using iStar 2.0. Model the following customer requirements in JSON format, ensuring that you follow the iStar2.0 description rules entered.
Single Sentence	Using the requirement modeling language for the iStar2.0, please provide a goal model for a business systems for liquor stores meant to provide online stores for customers, in stock, shipping, reservation, cancellation including different actors such as the liquor stores, customer, supplier and business systems.
Domain Paragraph	For example, consider the following requirements. Yakumo Sake Brewery sells wine and beer on reserve. Every day at 10:00 a.m., the manufacturer sends a predetermined number of each of these two types of liquor (e.g., 50 bottles of wine, 100 bottles of beer, etc.)....

syntactic constraint violations occurred. The text file used in this example contains constraints with incorrect examples.

Figure 1 shows the example results of the input prompt. ChatGPT generated an error-free iStar model according to the syntactic definitions, accurately describing the necessary elements for both the sale and purchase of alcohol. However, the content described in the iStar model appears to be somewhat generic. Although the domain paragraph specifies that Yakumo Liquor Store only sells wine and beer, this information is not reflected anywhere in the iStar model. Furthermore, all generated iStar models merely represent the content of the input prompt as is, without addressing any conflicts between elements or proposing new ideas.

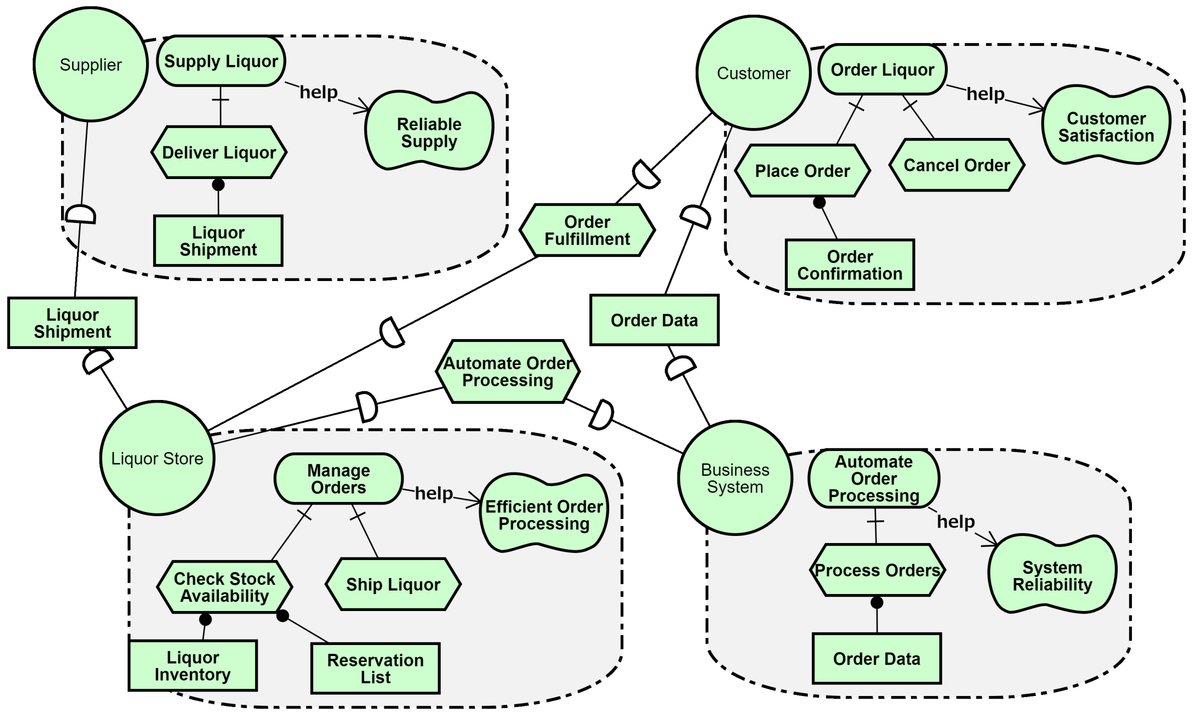

Figure 1: The Results of the Input Prompt

3.2. Modification of the iStar Model and its Results

Next, based on the results from Section 3.1, a prompt was entered within the same thread to detect and correct Bad Smells, and the output results were used as the example outcomes. After inputting the modification prompt, ChatGPT generated detection prompts for each type: inconsistency, ambiguity, and omission. It then generated the detection results and the modified iStar model. The following is a prompt for detecting omissions that was generated by ChatGPT using the CSV file and the prompt for generating prompts, as described in Seciton 2.2: ”Detect any omissions in the iStar 2.0 model, where actors have incomplete tasks or goals that don’t fully achieve their objectives.” Using this prompt, ChatGPT identified the missing element ”Track Order” within the Customer actor needed to achieve the ”Order Liquor” goal. The modified iStar model is shown in Figure 2. For other ambiguities and inconsistencies, ChatGPT performed checks and determined that they were not considered Bad Smells. In another example, redundancy was added as a new Bad Smell and was checked accordingly. Additionally, to resolve ambiguity, ”Automate Ordering” was revised to ”Automate Customer Order Processing,” clarifying the ambiguous expression.

Our findings indicate that we successfully generated iStar models without syntactic constraint violations and effectively detected and corrected Bad Smells. However, challenges remain regarding the quality of the generated iStar models as models; they tend to be too generic and fail to address conflicts between elements or propose new ideas.

4. Conclusion and Future Works

In this paper, it was highlighted that generating iStar models, one of the goal-oriented requirements analysis methods, is particularly challenging for beginners due to the lack of guidance on how to refine the models. This often leads to the creation of low-quality iStar models. To address this issue, a method for automatically generating iStar models using ChatGPT-4 was proposed. This method involved inputting syntactic definitions alongside the requirements defi-

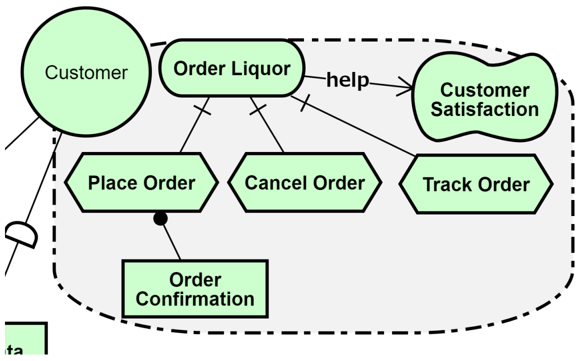

Figure 2: Results of the iStar Model Modification

nitions and generating the models sequentially as part of the input prompt design. Furthermore, to improve the quality of the iStar models generated by ChatGPT-4, a method was proposed to detect and correct Semantic Bad Smells in the iStar models using Self-Instruct. As a result, the method successfully generated iStar models without syntactic constraint violation and detected and corrected Semantic Bad Smells. However, the generated iStar models tended to be somewhat generic in content, leaving room for further refinement to support deeper analysis. Additionally, while the models captured the current state of the requirements definitions, they failed to propose conflicts between elements or new ideas. Future works include the following:

- Creation of prompts to further refine the generated iStar models
- Creation of prompts to propose new ideas, not just reflect the current state of requirements definition
- Creation of prompts to incorporate domain information from the requirements definition into the iStar models
- Development of a tool to assist in generating iStar models
- Examine the necessity of the domain paragraph in the initial input prompts

Acknowledgments

This work was partly supported by JSPS Grants-in-Aid for Scientific Research Nos. JP21K11823 and Nanzan University Pache Research Subsidy (I-A-2) for the 2024 academic year.

References

[1] D. Arruda, M. Marinho, E. Souza, F. Wanderley, A chatbot for goal-oriented requirements modeling, in: Computational Science and Its Applications–ICCSA 2019: 19th International Conference, Saint Petersburg, Russia, July 1–4, 2019, Proceedings, Part IV 19, Springer, 2019, pp. 506–519.

[2] N. Marques, R. R. Silva, J. Bernardino, Using chatgpt in software requirements engineering: A comprehensive review, Future Internet 16 (2024) 180.

[3] B. Chen, K. Chen, S. Hassani, Y. Yang, D. Amyot, L. Lessard, G. Mussbacher, M. Sabetzadeh, D. Varró, On the use of gpt-4 for creating goal models: an exploratory study, in: 2023 IEEE 31st International Requirements Engineering Conference Workshops (REW), IEEE, 2023, pp. 262–271.

[4] H. Nakagawa, S. Honiden, Mape-k loop-based goal model generation using generative ai, in: 2023 IEEE 31st International Requirements Engineering Conference Workshops (REW), IEEE, 2023, pp. 247–251.

[5] OpenAI, Chatgpt: GPT-4o model, 2024. URL: https://chat.openai.com/, accessed: 2024-08-18.

[6] Y. Hirabayashi, S. Ohota, S. Fujii, M. Saeki, Defining bad smells and automating their detection in goal-oriented requirement analysis method istar, in: 2023 30th Asia-Pacific Software Engineering Conference (APSEC), IEEE, 2023, pp. 349–358.

[7] J. Pimentel, J. Castro, pistar tool - A pluggable online tool for goal modeling, in: 26th IEEE International Requirements Engineering Conference, RE 2018, 2018, pp. 498–499. URL: https://github.com/jhcp/piStar.

[8] Y. Wang, Y. Kordi, S. Mishra, A. Liu, N. A. Smith, D. Khashabi, H. Hajishirzi, Self-instruct: Aligning language models with self-generated instructions, arXiv preprint arXiv:2212.10560 (2022).

[9] F. Dalpiaz, X. Franch, J. Horkoff, istar 2.0 language guide, 2016. URL: http://arxiv.org/abs/1605.07767v3.pdf.