KAOS Modeling Editor: A tool for semi-automated goal modeling

Keitaro Watanabe[1], Hiroyuki Nakagawa[1] and Tatsuhiro Tsuchiya[1]

[1]*Graduate School of Information Science and Technology, Osaka University, 1-5 Yamadaoka, Suita, Osaka, Japan*

Abstract

KAOS modeling editor is a tool that enables semi-automatic goal modeling via browsers. This tool draws/derives a goal model from a requirements description of a software system. This tool allows software engineers to discover and organize goals that the envisioned software has to accomplish. Engineers can also edit the diagram manually using a graph interface or DOT source file. Moreover, the goal description can be stated in natural languages. The tool is currently available in Github[1] and can be installed in a PC with *Node.js* and *Python*. The demo video[2] is also available on Youtube.

Keywords

goal modeling, requirements description, natural language processing

1. Introduction

In recent times, the requirements that software must meet have grown increasingly complex, mirroring the expanding scale of software development. Consequently, software developers often grapple with the challenge of discerning precisely what their clients expect from their products, striving to avoid either excessive features or deficiencies. To tackle this issue, a goal-oriented requirement analysis model, referred to as the goal model, is employed. In our work, we propose an open-source editing tool that automatically generates an initial KAOS goal model based on the provided requirements description, detailing the states or objectives of a software system.

Numerous studies have explored goal modeling, including efforts to extract and visualize goals through data mining techniques [1], deriving domain models from requirements using natural language processing [2], web-based tools for goal model construction [3], assistance tools for i* goal models via machine learning and algorithms [4], tools for generating i* models from narratives [5], and automatic i* model generation through natural language processing [6]. Additionally, this study is based on Nakagawa's interactive goal modeling theory [7]. While recent research has heavily focused on automatic goal modeling, none have produced a fully

[1]https://github.com/OrdinaryHyphen/KAOS-modeling-editor

[2]https://youtu.be/qZtHIdtfSuk

ER2023: Companion Proceedings of the 42nd International Conference on Conceptual Modeling: ER Forum, 7th SCME, Project Exhibitions, Posters and Demos, and Doctoral Consortium, November 06-09, 2023, Lisbon, Portugal

k-watanabe@ist.osaka-u.ac.jp (K. Watanabe); nakagawa@ist.osaka-u.ac.jp (H. Nakagawa); t-tutiya@ist.osaka-u.ac.jp (T. Tsuchiya)

https://github.com/OrdinaryHyphen/ (K. Watanabe); http://www-ise4.ist.osaka-u.ac.jp/h-nakagawa/index-j.html (H. Nakagawa); https://tatsuhirotsuchiya.github.io/jp/ (T. Tsuchiya)

comprehensive goal model due to incomplete methods or models. In contrast, our modeling editor can automatically generate preliminary KAOS goal models, allowing users to refine and modify them, offering a head start in goal model construction.

2. Overview

Figure 1 showcases our proposed editor. This tool functions in environments with Node.js and Python and is compatible with web browsers like Google Chrome and Firefox. On the left side, there's a text field for entering requirement descriptions. After the user input and activation by pressing the button below, the editor autonomously generates a goal model on the right side after a set time. You can manipulate the goal model's structure directly using the right-side interface. Our KAOS modeling editor is based on GVE[8], designed for GraphViz graphs and accessible via web browsers like Google Chrome and Firefox. Consequently, it includes most of GVE's user interfaces (UIs) and functions, except for automatic goal modeling.

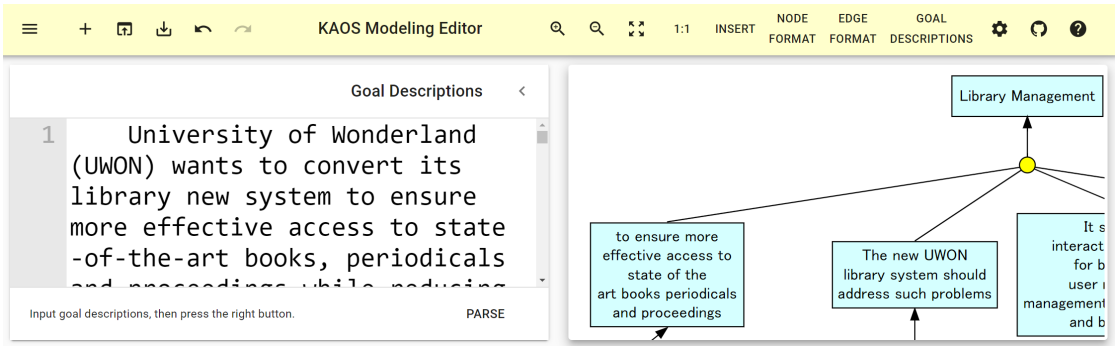

Figure 1: Snapshot of the proposed editor.

3. Implementation

This section describes the automatic construction of a goal model using the proposed tool.

Parsing sentences into words Initially, the editor parses the sentences into words. Each word has information about its part of speech, and its dependencies on other words.

Extraction of important sentences Following this, crucial sentences are identified based on specific keywords. These critical sentences pertain to the primary requirements outlined in the requirement description. To illustrate, in the context of describing an automotive control software system, sentences containing words like "car," "driving," and "speed," as well as sentences following them, are considered to contain essential requirements.

To extract these vital sentences, our proposed tool leverages a concept known as an "epoch phrase" as a guiding principle for identification. An epoch phrase is a specific phrase that not only includes words occurring a predetermined number of times within the requirement description but also incorporates particular terms like "envisioned" and "to be developed."

For instance, phrases such as “The envisioned auto-driving software” or “The speed control system to be developed” fall within the epoch phrases category when terms like “auto-driving,” “software,” “speed,” and “system” appear a predefined number of times in the descriptions. In the case of a specification for an automotive control system, these terms undoubtedly refer to the automotive control system under development and typically precede related requirements. Consequently, sentences within the same paragraph following these epoch phrases are singled out as pivotal sentences.

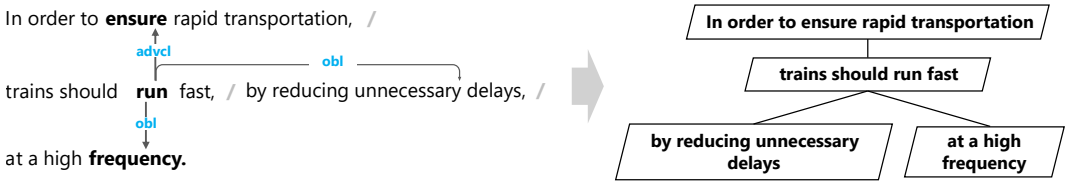

Figure 2: Construction of a preliminary goal model based on dependency tags.

Extraction and linking goals Figure 2 illustrates the creation of a KAOS goal model based on word-to-word dependencies. In general, one word can be considered “dependent” on another. As demonstrated in Figure 2, the word “ensure” in the first line displays an “advcl” dependency originating from the word “run” in the second line. The arrows in the figure visually represent this connection. An “advcl” dependency arises when an “adverbial clause modifier” is present in the text. An adverbial clause modifier is a clause that modifies another word, like adverbs such as “very,” “usually,” “pleasantly,” and so on. Our tool creates a goal model by breaking down sentences into phrases based on identified word dependencies. It forms parent-child connections between these phrases according to the dependencies that link them. For instance, the “advcl” dependency links the phrase in the first line of Figure 2 with the phrase “trains should run fast” in the second line, resulting in an automatic parent-child relationship between them. However, connecting goals from different sentences through dependencies is challenging since dependencies are limited to words within a single sentence. In such cases, our tool compares nouns in one goal with those in another and establishes a parent-child relationship between goals with the most similar nouns. This is based on noun co-occurrence, indicating that the goals may share the same topic, describing a common concept.

4. Evaluation

Figure 3 shows a part of a preliminary KAOS goal model that the proposed editor automatically generates. The other part of the model was omitted due to space limitations. To assess the accuracy of the tool, we conducted an experiment comparing the generated goal model with the modified goal model. We used two metrics to evaluate the applicability of the editor. First, we used NDCG, which measures the accuracy of the ranking prediction, and the number of editor operations. To begin with, it was necessary for us to measure how closely the automatically generated model resembled the ideal model’s structure and to assess the differences. For this purpose, we prepared rankings for both the automatically generated goal model and the manually adjusted goal model, arranging the nodes in each by their depth. We then compared

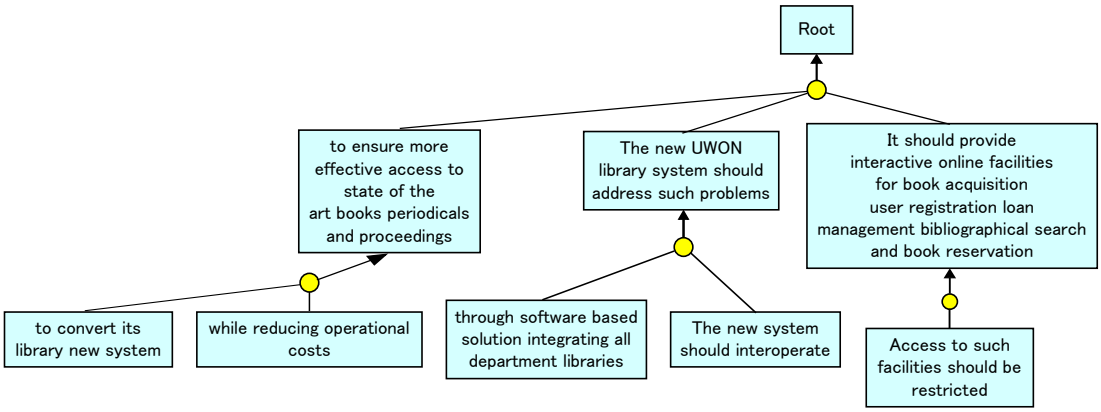

Figure 3: A half portion of auto-generated goal model for the library management system.

these rankings using the NDCG metric to evaluate the accuracy of the model's automatic generation. Next, we manually used the editor to measure which operations and how many times it took to modify the preliminary goal model to make a valid goal model. This is a direct indicator regarding the usefulness of the tool.

Table 1
Evaluation for KAOS modeling editor

Requirements Description	NDCG Score	Goals Added	Goals Edited	Relation Edited
Library Management	0.940	1	59	3
Train Control	0.710	4	668	13
Meeting Scheduling	0.529	38	1554	52

Table 1 shows the evaluation results for goal models generated from three different requirements in a requirements engineering book[9] using our developed tools. These requirements are related to Library Management, Train Control, and Meeting Scheduling systems, respectively. Each requirement is stored as a text file with 477 to 1080 words.

The accuracy of the automatically generated models depended on how precisely goals were extracted. For instance, in modeling Library Management requirements automatically, most necessary goals were successfully extracted with minimal manual intervention. In contrast, for Meeting Scheduling, the Normalized Discounted Cumulative Gain (NDCG) score was lower than the other two systems, and manual adjustments significantly deviated from the automatic model, requiring substantial effort. This was due to many goals not being extracted automatically, necessitating manual input during adjustment. It's important to note that the figures mentioned represent character counts. In terms of word count, it took around 200 words for input. The maximum time for manual model adjustment ranged from 20 to 30 minutes.

The challenge in achieving perfect automatic extraction accuracy lies in the mechanistic nature of goal extraction algorithms, which lack context consideration in requirement descriptions.

In essence, it's like having a rule such as "extract phrases containing infinitives as goals." The algorithm doesn't distinguish contextually relevant infinitives from irrelevant ones; it extracts them all. Solely relying on algorithmic judgments for context is a significant hurdle. To address this fundamentally, it's advisable to explore the integration of large-scale language models like GPT-4 in the future, capable of understanding and interpreting context.

5. Conclusion

In this study, we introduced our tool, the KAOS Modeling Editor, designed to autonomously create initial KAOS goal models from requirement descriptions. It identifies crucial segments using epoch phrases and then extracts and connects goals based on dependencies and noun co-occurrence. The highest accuracy achieved for initial models is 0.940 by NDCG, with a maximum generation time of 30 minutes. Our future goals include integrating LLMs like GPT-4 into the goal model implementation process to improve accuracy and processing speed.

References

[1] M. Rahimi, M. Mirakhorli, J. Cleland-Huang, Automated extraction and visualization of quality concerns from requirements specifications, in: 2014 IEEE 22nd International Requirements Engineering Conference (RE), 2014, pp. 253–262. doi:10.1109/RE.2014.6912267.

[2] C. Arora, M. Sabetzadeh, L. Briand, F. Zimmer, Extracting domain models from natural-language requirements: Approach and industrial evaluation, in: Proceedings of the ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems, MODELS '16, Association for Computing Machinery, New York, NY, USA, 2016, p. 250–260. URL: https://doi.org/10.1145/2976767.2976769. doi:10.1145/2976767.2976769.

[3] J. Pimentel, J. Vilela, J. Castro, Web tool for goal modelling and statechart derivation, in: 2015 IEEE 23rd International Requirements Engineering Conference (RE), 2015, pp. 292–293. doi:10.1109/RE.2015.7320444.

[4] Q. Zhou, T. Li, Y. Wang, Assisting in requirements goal modeling: A hybrid approach based on machine learning and logical reasoning, in: Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems, MODELS '22, Association for Computing Machinery, New York, NY, USA, 2022, p. 199–209. URL: https://doi.org/10.1145/3550355.3552415. doi:10.1145/3550355.3552415.

[5] R. Mesquita, A. Jaqueira, M. Lucena, C. S. Filho, F. M. R. Alencar, Us2startool: Generating i* models from user stories, in: International i* Workshop, 2015.

[6] T. Güneş, F. Aydemir, Automated goal model extraction from user stories using nlp, 2020.

[7] H. Nakagawa, H. Shimada, T. Tsuchiya, Interactive goal model construction based on a flow of questions, IEICE Transactions on Information and Systems E103.D (2020) 1309–1318. doi:10.1587/transinf.2019KBP0015.

[8] magjac, Graphviz visual editor, 2022. magjac.com/graphviz-visual-editor.

[9] A. van Lamsweerde, Requirements Engineering, Wiley, 2011. URL: https://lead.to/amazon/jp/?op=bt&la=en&key=B00DWHU40E.