Jeong-Jin Kang
Edward J. Rothwell
Yang Hao
Sang-Hyun Lee

Advanced and Applied Convergence Letters AACL 22

Advanced and Applied Convergence & Adavanced Culture Technology

11th International Symposium, ISAAC 2023
in Conjunction with ICACT 2023, ICKAI 2023

November 16 - 18, 2023, Jeju, Korea
Revised Selected Papers

IIBC The Institute of Internet, Broadcasting and Communication
IPACT The International Promotion Agency of Culture Technology
지식의 숲 FOREST OF KNOWLEDGE

UML State Diagram Extraction from Natural Language based Requirements with Linguistic Analysis Mechanism

Ye Jin Jin*, Jang Hwan Kim**, Ji Hoon Kong***, R. Young Chul Kim****

*,**,****Department of Software and Communications Engineering, Hongik University,
***Toonsquare,

{yejin_jin, janghwan.kim}@g.hongik.ac.kr, john.tooning@toonsquare.co, bob@hongik.ac.kr

Abstract

The current software projects perform development from natural language-based requirement definition. The requirement analysis phase, where most errors occur, necessitates precise analysis. It is crucial to have a systematic requirement definition and analysis process to address software errors as a whole[1]. As software projects grow in scale and complexity, considerable effort is required to develop high-quality software through requirement analysis. To address these challenges, this paper proposes the analysis of natural language-based requirements and extracting object state designs. In other words, it identifies objects and states to extract state diagrams from natural language requirements using linguistic principles from linguists Chomsky and Fillmore. Through the proposed mechanism, the aim is to automate the analysis of natural language-based requirements. Additionally, these state diagrams are intended to be applied to the state transitions of objects within the Cartoon.

Keywords: *Requirement, State Diagram, Fillmore's Semantic Role, Chomsky's Syntactic Structure*

1. Introduction

The primary cause of software failures lies in the requirement phase of the software development process. 66% of actual software errors stem from system specifications[1]. This is due to the differences in interpretation between customers and developers, as requirements are composed in natural language. Moreover, accommodating ongoing informal customer requirements is challenging and comes with associated costs. To address these issues, previous research has analyzed requirements using linguistics-based natural language processing[2]. Building on this research, our study proposes a mechanism to analyze requirements according to the software lifecycle and extract state diagrams. Through this, we anticipate an improvement in customer and developer understanding of requirements, as well as the ability to identify and analyze requirements systematically.

2. Related Research

2.1 Mapping Design Thinking Mechanism with Software Development Process

This study combines software development process techniques and design thinking mechanisms to propose a code template[3]. It defines requirements specifications to extract use cases and, in turn, derive the design. Through this process, systematic software development becomes possible.

In this research, we skip the mapping process between requirements and use cases and directly extract state diagrams from requirements.

2.2 Fillmore's Case Grammar

Fillmore's Case Grammar is a linguistic theory that focuses on the relationship between nouns and verbs in

J.J. Kang et al. (Eds.): ISAAC 2023, AACL 22, 2023

sentences[4]. This theory examines the combination of semantic roles based on verbs to analyze syntax. Table 1 defines the roles of noun phrases related to verbs.

Table 1. Fillmore's Case Grammar

Case	Definition
Agent	A person or entity causing a verb's action to be performed.
Counter agent	The force or resistance against which a verb's action is carried.
Object	An entity affected directly by a transitive verb's action.
Experiencer	A person or thing affected by a verb's action, replacing the dative.
Source	The place from which something moves.
Goal	The place to which something moves.
Locative	Location or spatial orientation of the state or action.
Instrument	The inanimate entity is causally involved in a verb's action.
Time	The time of the event.

In studies that apply such linguistic theories to software engineering, Case is redefined, and requirements sentences are analyzed to extract use cases[2]. Building upon this, our research applies Fillmore's theory to analyze natural language-based requirements.

3. The method proposed in this study

3.1 Requirements analysis and status extraction process

In this study, we analyze natural language requirements and extract state diagrams for objects.

The process is as follows:

1. Using the Stanford Parser, natural language requirements were analyzed to identify part-of-speech tags at the word level following Chomsky's Syntactic Structure approach[5].

2. With the identified part-of-speech tags, we extract the relationships between nouns and verbs through Fillmore's linguistics. The conventional use of Fillmore's Case Grammar has limitations when analyzing object states. Therefore, we augment it by incorporating Fillmore's Semantic Roles and redefining the elements necessary for state diagrams, as shown in Table 2.

Table 2. Fillmore's Case Grammar Redefinition

Role [Notation]	***Definition***
Agent [A]	The entity that performs an action.
Patient [P]	The entity that undergoes an action or change of state.
Theme [T]	The entity that is focused upon or affected by an action.
Experiencer [E]	The entity that perceives or feels.
Cause [C]	The entity that causes an action or event.
Recipient [R]	The entity that receives something.
Instrument [I]	The entity used to perform an action.
Location [L]	The place where an action occurs.
Goal [G]	The endpoint of a movement.
Source [S]	The starting point of a movement.

3. Using the extracted relationship information, we identify objects by recognizing the subject of the sentence as the object and analyzing it as the Agent.

4. We extract state diagrams by considering the action information of the identified objects, the states before and after the action, and any associated conditions. Table 3 presents the necessary object information for extracting state diagrams.

Table 3. Main parameters

Category	Description
State	All states for an object
Event	Events that change the state of an object
Transition	Transitioning the state of an object for a specific event
Condition	Conditions that must be met for transition
Action	Action performed by the object after transition

3.2 Application Example

A computer power system was applied to the mechanism proposed in this study. The entire process of extracting state diagrams is depicted in Figure 1.

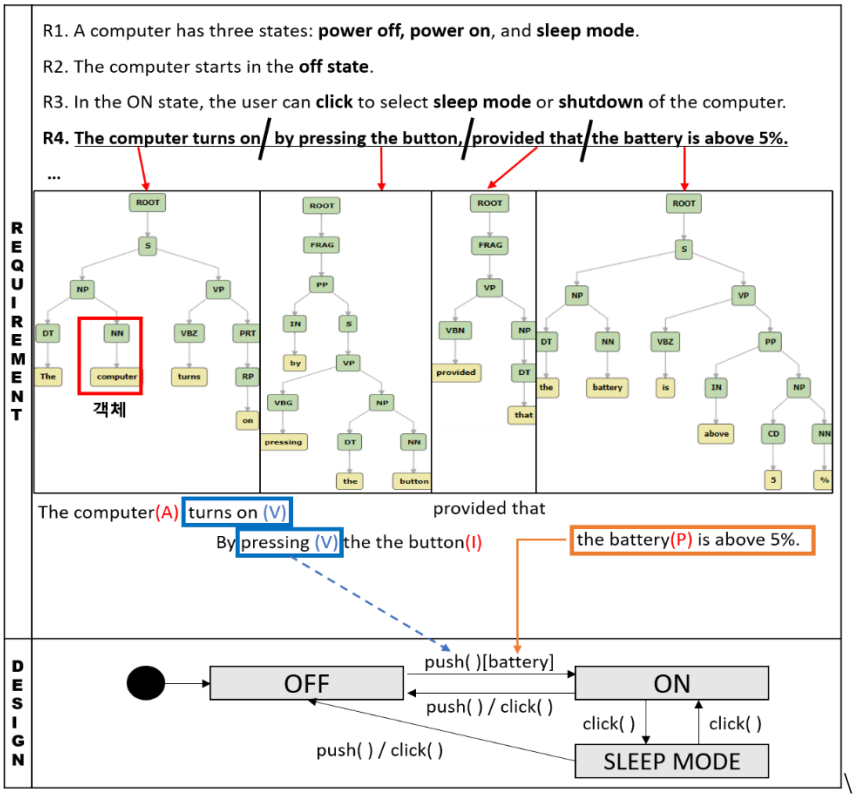

Figure 1. The process of extracting state diagrams from requirements

In Figure 1, we analyzed the sentence from requirement R4 of the system. Complex sentences were broken down into simple ones, and each simple sentence was further divided into clauses, with part-of-speech tags identified at the word level for each clause. Role notations were appended alongside the parts of speech, which were then used to extract the state diagram.

4. Result

In this study, we propose a mechanism for analyzing natural language-based requirements through Fillmore's linguistic theory and representing them as designs. By employing the proposed mechanism, communication between customers and developers will be significantly enhanced through the extracted designs. Additionally, we anticipate that systematic mechanisms will greatly facilitate the identification and

management of informal requirements.

5. Acknowledgement

This research was supported by Culture, Sports and Tourism R&D Program through the Korea Creative Content Agency(KOCCA) grant funded by the Ministry of Culture, Sports and Tourism(MCST) in 2023(Project Name: Development of AI-based user interactive multi-modal interactive storytelling 3D scene authoring technology, Project Number: RS-2023-00227917, Contribution Rate: 100%)

Reference

[1] R. Kaur, and J. Sengupta, "Software process models and analysis on failure of software development projects," *IJSER*, vol.2, 2011.

[2] B. K. Park, and R. Y. C. Kim, "Effort estimation approach through extracting use cases via informal requirement specifications," *Applied Sciences*, vol.10, no.9, 2020.

[3] C. Y. Seo, J. H. Kim, and R. Y. C. Kim, "Applied Practices on Codification Through Mapping Design Thinking Mechanism with Software Development Process," *KIPS Transactions on Computer and Communication Systems*, vol.10, no.4, pp.107-116, 2021.

[4] C. J. Fillmore, *The Case for Case. Universals in Linguistic Theory*, HR&W, New York, U.S America, 1968.

[5] C. D. Manning, M. Surdeanu, and J. Bauer, "The Stanford CoreNLP Natural Language Processing Toolkit," *In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp.55-60, 2014.