Extracting Domain Models from Japanese Natural-Language Requirements

Mingzhe Yu[1,a), Kenji Hisazumi[2,3,b), Akira Fukuta[3,c)

Abstract: Domain modeling that illustrates the elements that make up the development target system and their relationships is an important step in the transition from natural-language requirements to precise specifications. However, in a large-scale system, a large number of target documents are required, and it takes a lot of man-hours to construct domain models manually. Most of the existing methods define rules for extracting domain model elements and build domain models based on them, but many propose only rules for only English grammar. As a result, it cannot be applied directly to Japanese requirements documents, nor can it handle ambiguities in documents written in Japanese.

Therefore, in this research, firstly, we divert extraction rules based on English grammar and propose extraction rules for Japanese required documents. We keep the extraction rules proposed in the existing research that can be used for the results of Japanese grammar analysis, correct or remove the inappropriate rules and define the Japanese unique rules. Moreover, in order to reduce the ambiguity of the document described in Japanese, we propose a method to support the improvement of the unambiguity of the document by changing the model interactively. Furthermore, the prototype of a domain model extraction tool based on these proposed methods is implemented as a plug-in of the existing UML creation tool which shows the result of extraction and support changing on it.

1. INTRODUCTION

Natural language (NL) is used prevalently for expressing systems and software requirements [1]. Building a domain model is an important step for transitioning from informal requirements expressed in NL to precise and analyzable specifications [2].

A domain model is a system of abstractions that describes selected aspects of a sphere of knowledge, influence or activity. The model can then be used to solve problems related to that domain. The domain model is a representation of meaningful real-world concepts pertinent to the domain that need to be modeled in software. The concepts include the data involved in the business and rules the business uses in relation to that data.

It is necessary that the engineers examine the requirements and ensure that all the concepts and relationships relevant to the requirements are included in the domain model when building a domain model that is aligned with a given set of requirements. This is a laborious task for large applications, where the requirements may constitute tens or hundreds of pages of text. Automated assistance for domain model construction based on NL requirements is therefore important.

Additionally, considering the current situation of the Japanese software industry, the waterfall model and other development methods that rely heavily on the requirements document are still the mainstream. The tools for automatically implementing the domain model extraction are still very necessary. However, most of the existing methods define rules for extracting domain model elements and build domain models based on them, but many propose rules for only English grammar. As a result, it cannot be applied directly to Japanese requirements documents, nor can it handle ambiguities in documents written in Japanese. So, at this time, we try to: (1) Divert extraction rules based on English grammar and propose extraction rules for Japanese required documents. (2) Propose a method to support the

1.九州大学大学院システム情報科学府
Graduate School of Information Science and Electrical Engineering, Kyushu University, Motooka 774, Nishi-ku, Fukuoka 819-0395, Japan.
2.九州大学システム LSI 研究センター
System LSI Research Center, Kyushu University.

improvement of the unambiguity of the document by changing the model interactively. (3) Implemented as a plug-in of the existing UML creation tool which shows the result of extraction and support changing on it.

The paper is structured as follows. Section 2 proposes the extraction rules to extract domain models from Japanese NL documents. We explain our approach in section 3 and shows implementation in section 4. Section 5 describes the results of an experiment to demonstrate our proposal. Section 6 concludes this paper.

2. Related works

As we mentioned above, there are many researches about extracting domain models from English requirement documents. One of the most influential studies on our research is the rules and extractor built by Arora, Chetan, et al. [3], which organizes and defines a very complete English extraction rule set, and develops a corresponding extractor based on it.

Shigeo Kaneda, et. al.[8] point out that the use of class diagrams has a natural disadvantage for Japanese software engineers because it originated in the English language circle. In order to solve this problem, they analyze the correspondence between seven common sentence patterns in English with the sentence elements in Japanese, and based on their theory, they propose a guideline to draw class diagrams from Japanese requirement which would be useful for Japanses IT students and engineers.

Ouyang Liubo et. Al. propose a kind of automatic analysis modeling method based on the structural description of domain requirements [4]. This kind of automatic modeling is achieved based on modeling elements identification under structural description through predefined conversion rules and finally forms the UML graphical analysis results.

However, the first study did not include a useful GUI tool for software production, while the second study focused on the ability of Japanese-speaking engineers to efficiently draw class diagrams from requirements documents, they did not try to develop an effective tool to help engineers get rid of manual works. And the last one required the author of the requirements document writes the requirements document under structural description through predefined conversion rules, which is not so convenient in our opinion.

3. Finding rules for Japanese

3.1. Existing domain model extraction rules

Arora, Chetan, et al.[3] integrate the existing extraction rules and developed extraction tools to prove the validity of these rules in the actual requirements document. So, this time, our research considers migrating these extraction rules to the domain model extraction of the Japanese requirements document.

In order to convert the English extraction rules into Japanese, we need to find a Japanese grammatical structure whose analysis method corresponding to the English general grammatical. Here we choose case grammar.

Table 1 Extraction rules for Japanese

	A1	要件内のすべての名詞句(NP)は概念候補
	A2	再出 NP は概念
	A3	要件の主語は概念
	A4	要件の目的語は概念
	A5	要件のサ変動詞は概念
	B1	他動詞は関連
	B2	助詞を含む動詞は関連
	B3	「<R> <A>は<B>」という形式の要件の<R>は、関連付けである可能性
	B4	"含む"、"は"、"含"で構成され、[...]は集約/構成を示唆
	B5	"です"、"の種類"、"のようなもの"、“かも"、[...]は一般化の可能性
	C1	関連の元の概念が複数形でありもしくは普遍的な数量詞を持ち、ターゲットの概念が一意の存在量詞を持つ場合、関連は多対一
	C2	関連の元の概念が単数形であり、ターゲットの概念が複数形/定冠詞によって数量化されている場合、その関連は 1 対多
	C3	関連付けの元の概念が単数形で、ターゲットの概念も単数形である場合、関連付けは 1 対 1
	C4	概念の前の明示的な数値は基数
	D1	「識別される」、「認識される」、「持っている」[...]は属性を示唆
	D2	例えば NP の NP は属性を示唆する。
	D3	形容詞的に修正された NP の形容詞は属性を示唆
	D4	副詞を持つ他動詞は属性を示唆

Case grammar is a grammatical analysis theory proposed by American linguist Charles J. Fillmore [5] to correct the conversion grammar. The deep structure in the lattice grammar is represented by a central verb and a set of noun phrases. There is a semantic relationship between these noun phrases and verbs, which is called “deep case”.

Using the case grammar, we can get the Japanese version of the above rules like Table 1. Of course, at the current stage, this is just a blunt translation. We also need to add, delete, and improve this Japanese version of the rules in practice.

In fact, we did not implement the Rule B4, B5, C3, C4 and D1 because that the implementation of these rules is subject to differences between English and Japanese, lack of data or technical limitations. Although we do not rule out the possibility of continuing research on their implementation in the future, at this stage we believe that these rules are difficult to implement in Japanese language requirements extraction.

4. APPROACH

4.1. Pre-processing

Due to the use of mature tools, the pre-processing of this study no longer requires us to manually implement normal natural language processing like splitting words, removing stop words, sentence dependency analysis, etc.; but we also need to:

(1) Split sentences. We input our requirement document sentence by sentence.

(2) Remove scattered words or short sentences that are obviously not logically related. Retain long sentences containing subject-predicates.

(3) Remove or convert Arabic numerals or Roman words to standard Japanese. If you have a good text, you can ignore the above two steps.

In fact, we do not make special appointments or writing methods for the author of the requirements text, as same as the required text conforms to the daily Japanese usage rules.

4.2. Case Structure Analysis

Case Structure Analysis is the most important part of extracting processing. Because we use the Japanese case structure to map the English grammar structure in our research. However, the case structure analysis of Japanese is very different from that of English. In Japanese, postpositions are used to mark cases. Frequently used postpositions are “ga”, “wo” and “ni”, which usually mean nominative, accusative, and dative [6].

4.3. Generate Model Set

We traverse the analyzed sentence elements（mainly predicate verbs with various case elements） and collide them against the above rules. Then we save the eligible sentence elements in a set for further processing.

5. Implementation

There are many mature tools on the natural language processing in Japanese. this time, we chose KNP to analyze the Japanese sentences in our research.

KNP is a natural language processing tool developed by Daisuke Kawahara and Sadao Kurohashi [6].

We use the Astah UML software as a basis to try to develop convenient tools for our extracting program. we choose Astah as our host software because it supports plug-in development well.

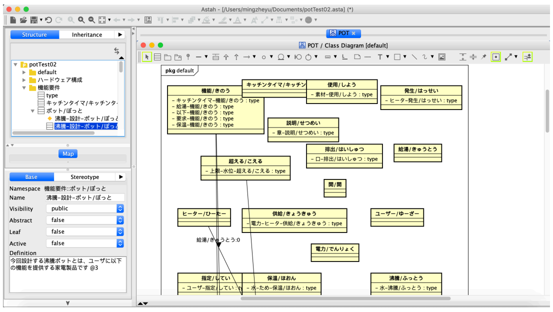
6. Experiment

This section describes an experiment to demonstrate our proposal, which is a relatively complex example about how to produce a boiling pot.

6.1. Boiling Pot

We try our Astah plug-in by analyzing a much more complex example, the first and second chapter of the famous topic boiling pot 6th edition[7] and draw the following Figure 1.

Figure 1 shows us how the extracted results are drawn in astah. The left column of the software shows the extracted domain model, and the right main canvas draws



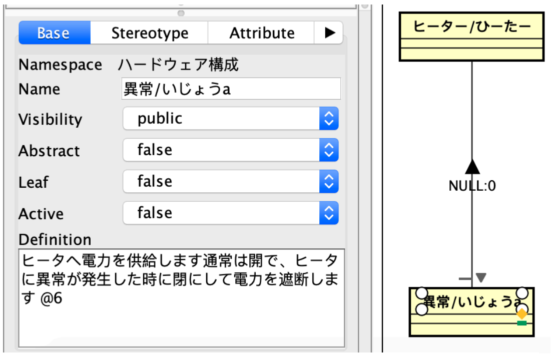
Astah - [/Users/mingzheyu/Documents/potTest02.asta] (*)

- potTest02
  - default
  - ハードウェア構成
  - 機能要件
    - type
    - キッチンタイマ/キッチンタ
    - ポット/ぽっと
      - 沸騰-設計-ポット/ぽっと
      - 沸騰-設計-ポット/ぽっと

Namespace 機能要件::ポット/ぽっと
Name 沸騰-設計-ポット/ぽっと
Visibility public
Abstract false
Leaf false
Active false
Definition
今回設計する沸騰ポットとは、ユーザに以下の機能を提供する家電製品です @3

Structure
Inheritance
Map
Base
Stereotype
POT
POT / Class Diagram [default]
pkg default
機能/きのう
- キッチンタイマ-機能/きのう : type
- 給湯-機能/きのう : type
- 以下-機能/きのう : type
- 要求-機能/きのう : type
- 保温-機能/きのう : type
キッチンタイマ/キッチン
使用/しよう
- 素材-使用/しよう : type
発生/はっせい
- ヒータ-発生/はっせい : type
説明/せつめい
- 章-説明/せつめい : type
排出/はいしゅつ
- 口-排出/はいしゅつ : type
給湯/きゅうとう
超える/こえる
- 上限-水位-超える/こえる : type
開/開
ヒーター/ひーたー
供給/きょうきゅう
- 電力-ヒータ-供給/きょうきゅう : type
ユーザー/ゆーざー
給湯/きゅうとう:0
電力/でんりょく
指定/してい
- ユーザ-指定/してい : ty
保温/ほおん
- 水-ため-保温/ほおん : type
沸騰/ふっとう
- 水-沸騰/ふっとう : type

Figure 1: The extraction result displayed in Astah



Namespace ハードウェア構成
Name 異常/いじょうa
Visibility public
Abstract false
Leaf false
Active false
Definition
ヒータへ電力を供給します通常は開で、ヒータに異常が発生した時に閉にして電力を遮断します @6

Figure 2: View original text

the models and their relationship. We develop the “View original text ” feature showed in Figure 2

6.2. Discussion

When evaluating the automatic extraction result, we find:

1) The extraction of concepts is relatively successful. Most of them will have some redundancy, but there are very few missing parts.

2) There is also a certain accuracy rate for the association extraction because most of the rules (B1, B2, and B3) are extracted based on the grammatical structure.

3) For the extraction of multiplicity and attributes, automatic extraction is much more difficult, because these rules are mostly based on semantics, word meaning rather than grammatical structure.

7. Conclusion

We present an automated approach based on Natural Language Processing for extracting domain models from unrestricted Japanese requirements. The main technical contribution of our approach is transferring the existing set of model extraction rules to adapt to Japanese requirement documents. We also build a plug-in based on a UML tool to support for manual improvement of generated class diagram quality and meet the needs of production practices.

As mentioned above, there is a lot of room for us improving our extraction method and plugin. In order to improve our extraction method, we may need an effective semantic analysis tool or dictionary in the future to tell us which words or sentences that Japanese writers would like to use to express "contain", "is made up of", "include", "identified by", "recognized by", "has",which are used frequently in English, in Japanese requirements or which Japanese word is usually plural or unique.

We will also add more features to the plugin to help users to get the final class diagram results faster. They maybe include: (1) A more clear class diagram with fewer association lines crossing.(2) Multi-document linkage.

参考文献

[1] K. Pohl and C. Rupp. Requirements Engineering Fundamentals. Rocky Nook, 2011.

[2] T. Yue, L. Briand, and Y. Labiche. A systematic review of transformation approaches between user requirements and analysis models. Requirements Engineering, 16(2), 2011.

[3] Arora, Chetan, et al. "Extracting domain models from natural-language requirements: approach and industrial evaluation." Proceedings of the ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems. ACM, 2016.

[4] Ouyang Liubo, Guo Hailin. Automatic analysis modeling method based on structural description of domain requirements. Computer Engineering and Applications, 2016

[5] Fillmore, Charles J. (1968) "The Case for Case". In Bach and Harms (Ed.): Universals in Linguistic Theory. New York: Holt, Rinehart, and Winston, 1-88.

[6] Daisuke Kawahara and Sadao Kurohashi. A Fully-Lexicalized Probabilistic Model for Japanese Syntactic and Case Structure Analysis, In Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL2006), pp.176-183, 2006.

[7] 組込みソフトウェア管理者・技術者育成研究会（SESSAME）, 話題沸騰ポット要求仕様書 (GOMA-1015 型) 第6版

[8] Kaneda, Shigeo, Akio Ida, and Takamasa Sakai. "Guidelines for Class Diagram Design based on English Sentence Patterns and Functional Dependency." TechnicalYreport, SIGYKBSE, IEICE of Japan,(March, 2014)(In Japanese) (2014).