PAPER Special Section on Knowledge-Based Software Engineering

Automating Bad Smell Detection in Goal Refinement of Goal Models*

Shinpei HAYASHI[†a)], ***Member***, **Keisuke ASANO**[†], ***Nonmember***, ***and*** **Motoshi SAEKI**[††b)], ***Member***

SUMMARY Goal refinement is a crucial step in goal-oriented requirements analysis to create a goal model of high quality. Poor goal refinement leads to missing requirements and eliciting incorrect requirements as well as less comprehensiveness of produced goal models. This paper proposes a technique to automate detecting *bad smells* of goal refinement, symptoms of poor goal refinement. At first, to clarify bad smells, we asked subjects to discover poor goal refinement concretely. Based on the classification of the specified poor refinement, we defined four types of bad smells of goal refinement: Low Semantic Relation, Many Siblings, Few Siblings, and Coarse Grained Leaf, and developed two types of measures to detect them: measures on the graph structure of a goal model and semantic similarity of goal descriptions. We have implemented a supporting tool to detect bad smells and assessed its usefulness by an experiment.

key words: *goal-oriented requirements analysis, goal refinement, smell detection*

1. Introduction

Goal-oriented requirements analysis (GORA) is one of the popular techniques to elicit requirements to business processes, information systems and software (simply, systems hereafter), and is being made into practice and worked into university curriculums [2], [3]. In GORA, customers' needs are modeled as goals to be achieved finally by software-intensive systems that will be developed, and the goals are decomposed and refined into a set of more concrete sub goals. After finishing goal-oriented analysis, the analyst obtains an acyclic (cycle-free) directed graph called *goal graph*. Its nodes express goals to be achieved by the system that will be developed, and its edges represent logical dependency relationships between the connected goals. More concretely, a goal can be refined into sub goals, and the achievement of the sub goals contributes to its achievement. We have two types of goal decomposition; one is AND decomposition, and the other is OR. In AND decomposition, if all of the sub goals are achieved, their parent goal can be achieved or satisfied. On the other hand, in OR decomposition, the achievement of at least one sub goal leads to the achievement of its parent goal.

According to the textbook by van Lamsweerde [2], "a *goal* is a prescriptive statement of intent that the system should satisfy through the cooperation of its system components (agents)", while "a requirement is a goal under the responsibility of a single system component ...". It means that requirements are derived by goal refinement, and poor goal refinement leads to missing requirements and to eliciting incorrect ones. In addition, it may cause less comprehensiveness of a goal model, and as a result, communication gaps among stakeholders occur. Goal refinement is really the most crucial step in GORA, and it is difficult even for experienced analysts. It is significant to detect poor goal refinement during the development of a goal model.

Figure 1 illustrates a part of a goal model of a book order system, which includes poor goal refinement. The root goal "Fulfill a book order" is refined into two sub goals with AND decomposition, which is indicated with an arc lying across the edges to the sub goals. The sub goal "Handle a receipt" does not seem to directly contribute to the achievement of its parent goal "Deliver books", rather some sub goals related to a carriage service of books, such as express delivery and designating the date of its arrival, seem to be missing. Although "Handle a receipt" may be one of the necessary functions for the system to be developed, this poor goal refinement leads to missing requirements to achieving "Deliver books". Moreover, generally speaking, sub goals that do not contribute to the achievement of a root goal lead to incorrect requirements, irrelevant to customers' needs. In this case, we can guess the poor refinement from the symptoms or signs that there is only one sub goal and that "Deliver books" is not so semantically related to "Handle a receipt". We have to detect *bad smells* of poor goal refinement, symptoms or signs that indicate a potential of

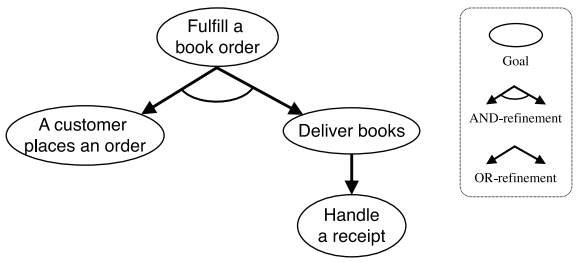


Fig. 1 Example goal graph having poor refinement.

Manuscript received May 10, 2021.
Manuscript revised October 23, 2021.
Manuscript publicized January 6, 2022.

[†]The authors are with Department of Computer Science, Tokyo Institute of Technology, Tokyo, 152–8550 Japan.
[††]The author is with Department of Software Engineering, Faculty of Science and Technology, Nanzan University, Nagoya-shi, 466–8673 Japan.

*This paper is extended based on [1], which appeared in proceedings of the 4th International Workshop on Conceptual Modeling in Requirements and Business Analysis, © 2017 Springer.

a) E-mail: hayashi@c.titech.ac.jp
b) E-mail: saekimot@gmail.com

DOI: 10.1587/transinf.2021KBP0006

a problem against requirements elicitation of high quality while developing a goal model. Note that the goal graph shown in Fig. 1 is just a part of a goal graph, and its complete goal refinement is not shown in this figure. It is illustrated only for readers to understand intuitively what bad smells are and where they can appear. To complete this goal graph, we may need domain-specific knowledge in a book order domain. For example, the knowledge of customer address is necessary to refine "Deliver books" completely, and how to get this kind of domain-specific knowledge such as customer address is a significant topic. The technique to organize and use domain-specific knowledge is out of scope in this paper, and we will discuss this significant topic as a future work later.

This paper proposes an automated technique to detect bad smells of poor goal refinement. As discussed above, we use the measures related to

1. structural characteristics of a goal model as a graph to detect goal refinement having only one sub goal and
2. similarity between goal descriptions of a parent goal and its sub goals to detect their semantic relation.

First of all, to classify bad smells and their features, we make a preliminary analysis where our subjects are required to indicate poor refinement in goal models and express its rationales explicitly. Based on this classification result, we develop several metrics to detect bad smells of goal refinement. Note that this work was done in Japanese, and all the example goal graphs written in English used in the studies were translated into Japanese to apply our tool to them.

The goal graphs dealt with in this paper are most simple ones consisting of goals with no types and refinement relationships as "achievement" relationships only. They do not have various types on nodes such as Task and Quality Goal, or on edges such as "qualification" and "needed-by" of iStar 2 [4] and KAOS [2]. The goals in this paper can have the descriptions on functional requirements, non-functional requirements, requirements that do not have a clear-cut criterion for their achievement, called soft goals [5], etc.

The rest of this paper is organized as follows. We will discuss this preliminary analysis in Sect. 2. Sections 3, 4, and 5 present the developed smell detection techniques, the automated tool, and the experimental evaluation of the techniques, respectively. Sections 6 and 7 are for related work and concluding remarks.

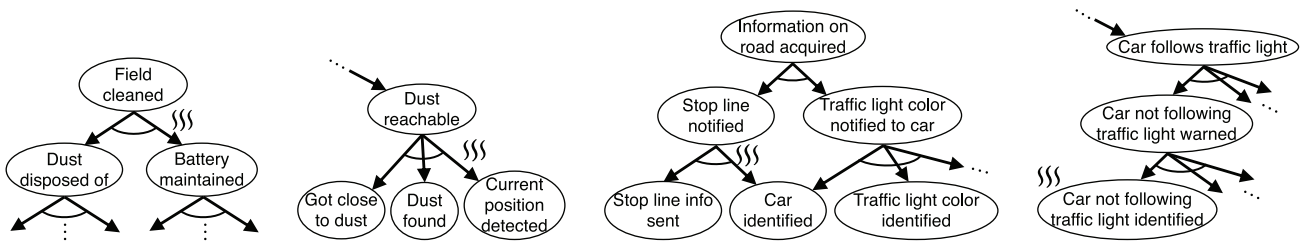
Note that this paper is an extended version of [1], by emphasizing the detailed explanation of our automatic detection technique in Sects. 2, 3, 4, and 5, including what led us to the proposal, and by making deeper experimental analysis in Sect. 5.

2. Preliminary Analysis

In this section, we present a preliminary analysis where we provided our three subjects with real goal models and let to indicate in them poor goal refinement, which may cause any problems in later steps of systems development. The subjects were also required to express explicitly to us the rationales why their indicated refinement is poor. Additionally, in the case when we could not understand their rationales well, we had interviews to clarify them. Our subjects were well familiar with GORA and were experienced in developing goal models. Through the results of this preliminary analysis, we clarified what poor goal refinement is. We selected four goal models from literature, which have from 10 to 22 goals, and showed them to our subjects. We analyzed the indicated poor refinement and its rationales, and as a result, we have four categories of poor goal refinement. Table 1 shows the categories of poor refinement and the number of their indicated occurrences. Figure 2 includes examples, and each of them follows a category of poor refinement in the table. Figures 2 (a) and 2 (b) show excerpt goal models found in a literature [6]. Figures 2 (c) and 2 (d) are from another literature [7]. Note that these excerpts are translated into English since the goal descriptions in the original goal models are written in Japanese.

Table 1 Classification of poor refinement

	Category of poor refinement	# indicated (total)	# one subject indicated	# two subjects indicated
#1	A sub goal does not contribute to the achievement of its parent directly	1	1	0
#2	A sub goal is irrelevant to the achievement of its parent	2	2	0
#3	There is a lacking sub goal to achieve its parent	4	1	3
#4	A leaf goal is not concrete	2	0	2



(a) Category #1. (b) Category #2. (c) Category #3. (d) Category #4.

Fig. 2 Examples of poor refinement.

The first category shows an existence of some logical gaps between a parent goal and its sub goal. Although the sub goal can contribute to the achievement of the parent in a certain degree, it is difficult for stakeholders to conjecture the achievement of the parent from the description of the sub goal. More refined goals are necessary between the parent and this sub goal, i.e., step-by-step refinement should be made. As shown in Table 1, one occurrence of this category was found by one subject. Figure 2 (a) shows an example poor refinement of this category. This goal model is for designing an automated cleaning robot. Here, the goal "Battery maintained" is defined as necessary for the achievement of "Field cleaned". However, the achievement of the goal "Battery maintained" does not *directly* contribute to the achievement of the goal "Field cleaned". To mitigate the logical gap between these two, some intermediate goals such as "Machine electric power supplied" or "Avoid insufficient electric power" are appropriate to be inserted between the two goals to express more natural refinement relationships clearly.

The second and third categories are related to correctness and completeness respectively in quality attributes of requirements specifications [8]. The second category shows that goal refinement derives a sub goal irrelevant to the achievement of a parent goal, and as a result, incorrect requirements, different from real customers' needs, are elicited. Our two subjects indicated two different occurrences of the poor refinement of this category. As for the third category, four occurrences where sub goals were lacking for achieving parents were indicated, and one of them was done by one subject and three by two subjects. Figures 2 (b) and (c) show examples poor refinement of these categories. In the excerpt goal model shown in Fig. 2 (b), although the achievement of the goal "Current position detected" contributes to the achievement of "Dust reachable", its purpose is to achieve the goal "Got close to dust" from the detected current position. Therefore, this goal may be more appropriate for the child of the goal "Got close to dust". In Fig. 2 (c), on the one hand, we can see that the achievement of the sub goal "Traffic light color identified" is required to achieve "Traffic light color notified to car". On the other hand, we can also see that the achievement of "Stop line notified" does not require the identification of stop line. This may indicate the necessity of a lacking goal "Stop line identified" for a sub goal of "Stop line notified".

The fourth category is for refinement insufficient to operationalize leaf goals. Leaf goals, which do not have any sub goals, should be sufficiently concrete to realize them as functions of the system to be developed. Our two subjects indicated that two leaf goals were too abstract to realize them and they should be refined further. It means that the granularity of the goals in refinement is coarse yet. Figure 2 (d) shows an example poor refinement of this category. Although the goal "Car not following traffic light identified" is stated as a leaf, its meaning is not sufficiently concrete to be achieved by a specific actor, and the whole graph lacks to express the details of how to identify the cars that are not following traffic light. This goal may be refined to sub goals expressing the concrete way how to identify such cars, achieved by a specific actor.

The first three categories are the relation between a parent and its sub goal, i.e., an edge between goals in a goal graph, while the fourth one is for a goal, i.e., a node. So, we make our technique indicate as poor refinement edges and leaf goals having specific features. In the next session, we will discuss metrics representing these specific features.

3. Approach

3.1 Bad Smells and Metrics to Detect Them

Based on the results in the previous section, we defined bad smells of four categories and metrics to detect them with GQM approach [9]. Figure 3 shows the process to obtain the smells and metrics. In Fig. 3, "Goal (Category of poor refinement)" is a problem appearing in goal graphs, and "Question (Smell)" is a symptom on the goal graphs that can indicate the potentials of "Goal (Category of poor refinement)". "Metrics" is a measurement that computes on the graph graphs to look for "Question (Smell)". The figure is just for producing computational metrics to detect the potentials of bad smells and not for being developed to uniquely and correctly determine a category of poor refinement.

Note that the category "a sub goal does not contribute to its parent directly" in the figure could not be differentiated from "a sub goal is irrelevant to the achievement of its parent goal" by the measures that computers can calculate. The difference between them is the degree of logical contribution to the achievement of a parent goal, and only human stakeholders and analysts who can deeply understand goal descriptions can differentiate them. So we merged them into the category "A sub goal does not contribute directly or is irrelevant", as shown in the figure.

To decide if a sub goal does not contribute to its parent directly, we focus on the strength of their semantic relationships to detect the smell Low Semantic Relation. If sibling goals, which are sub goals derived from a parent goal, are too many, there are possibilities that irrelevant goals are included in them. This is because of the difficulties of keeping the validity of refinement relationship between a parent goal and a set of many sub goals as a whole. Literature reported the capacity and recognition limits of human beings [10], [11], which may also be evidence of such difficulties. In addition, if the number of the sibling edges is large, i.e., many sibling goals, it can be considered that the parent goal is refined into the many sub goals without any intermediate refinement, and as a result, the many finer-grained sub goals as sibling ones appear to achieve their parent. We capture this symptom as a smell named Many Siblings. If there are lacking sub goals of a parent goal, the existing sub goals of the parent goal do not amount to the number of the sub goals really necessary to achieve the parent. So, as one of the clues of looking for "There is a lacking sub goal" symptom, we can pay attention to the smallness of the sib-

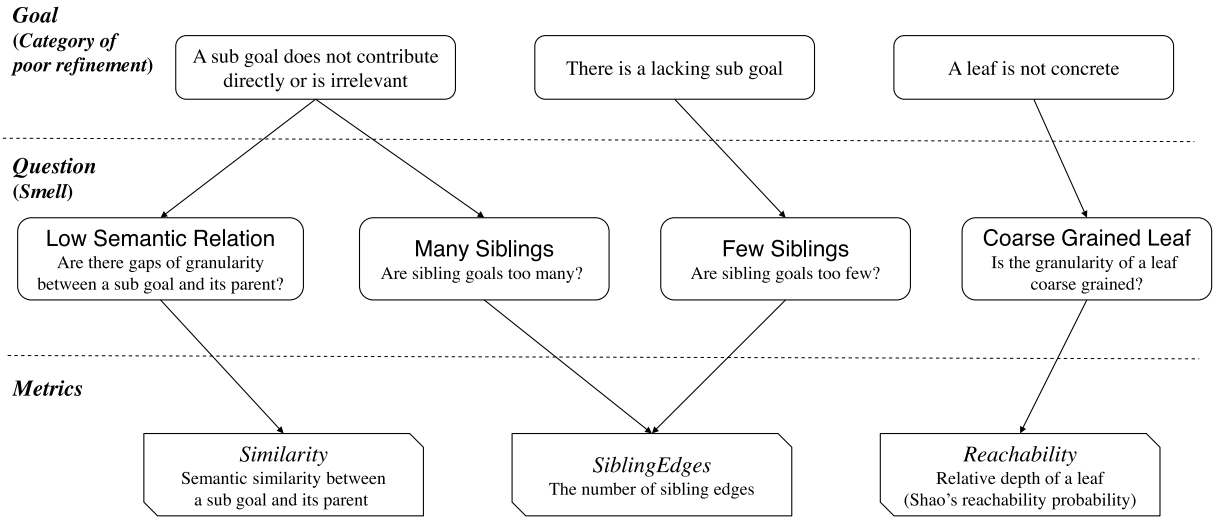


Fig. 3 Extracted smells and metrics using GQM approach.

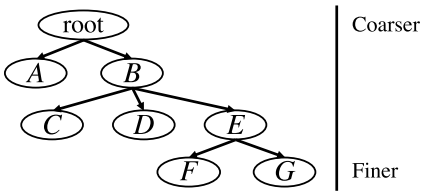


Fig. 4 Granularity of leaf goals.

ling goals refined from the parent goal. As shown in Fig. 3, we put the smell Few Siblings as a question and adopt the metrics *SiblingEdges* to answer the question.

As for the granularity of leaf goals, we calculate the relative depth of a leaf from a root goal in a graph. See Fig. 4. This example has five leaf goals A, C, D, F, and G. According to [2], each of these leaf goals is achieved by a single agent and leads to a requirement that the system to be developed will realize as a function. If they are not sufficiently refined yet, they are vague, and it is difficult to realize them as concrete functions of the system. The depth or distance of the leaf goal A from a root is smaller rather than those of the other leaves, in particular than F and G, and it can be considered that A is necessary to be refined further, compared with F and G. The idea behind this detection is that in a goal refinement, abstract goals closer to the root goal are refined into more concrete goals by the goal decomposition from a parent goal to multiple child goals, and the concept in the root goal is split into all the leaf goals in the end. If a leaf goal is weaker decomposed, i.e., in shallower decomposition depth and/or shorter decomposition breadth, than the other leaf goals in the same goal model, it can be considered to have further decomposition possibility. Thus, we detect the smell Coarse Grained Leaf by focusing on the metric measuring relative depth of goals, and the measure we use has been based on the idea of combining reachability probability and Shannon's information entropy, proposed by Shao [12]. We did not use simple measures such as the distance of the goal from a root goal because the factor of branch breadth as a graph should be considered.

The details of these measures will be mentioned in the next subsections.

3.2 Measuring Similarity of Goal Descriptions

The semantic similarity between goals is calculated with their goal descriptions written in natural language, and therefore we use a lightweight natural language processing technique, the case frame approach [13].

3.2.1 Case Frame

The technique we adopted is based on case frames of Fillmore's case grammar, which are semantic representations of natural language sentences [13]. The technique based on the case frame approach has been used widely in requirements engineering [14]–[16]. A case frame consists of a verb and semantic roles of the words that frequently co-occur with the verb. These semantic roles are specific to a verb and are called *case* (precisely, *deep case*). For example, "Deliver a book" is transformed into the case frame:

$$(\text{Deliver}, {}^{\text{actor:}}\text{–}, {}^{\text{object:}}\text{Book}, {}^{\text{destination:}}\text{–}).$$

Actor, object, and destination are case names, and "–" stands for no words filled in the slot. Generally speaking, a verb has multiple meanings, so to identify its meaning in a goal description, we use constraints on the words co-occurring with it. For example, "deliver" has the meaning of "speak", e.g., delivering a speech, in addition to "convey". When "deliver" is used as the meaning of "speak", the case slot of "object" should be filled with words denoting something abstract concepts such as "speech", "lecture", not words denoting the concepts of physical goods like "book". We use this kind of constraints on the semantic concepts of words that case slots can be filled with. Therefore, we have a

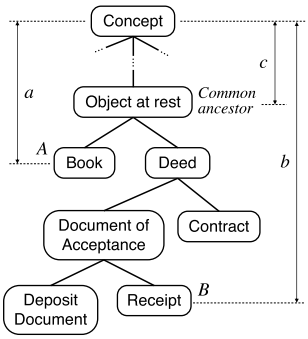


Fig. 5 Similarity of words in hierarchical structure.

dictionary of case frames and that of hierarchical (semantic) concepts. This approach is the same as the work by Nakamura *et al.* [17].

3.2.2 Semantic Similarity

A dictionary of hierarchical concepts is used to calculate semantic similarity between goal descriptions. Suppose that we have three goal descriptions: "Deliver a book", "Handle a receipt", and "Send a receipt" and obtain their case frame representations as follows:

(Deliver, $^{\text{actor:}}$–, $^{\text{object:}}$Book, $^{\text{destination:}}$–),
(Handle, $^{\text{actor:}}$–, $^{\text{object:}}$Receipt, $^{\text{instrument:}}$–),
(Send, $^{\text{actor:}}$–, $^{\text{object:}}$Receipt, $^{\text{destination:}}$–).

The occurrences of words in these case frames are mapped into their semantic concepts using the dictionary of hierarchical concepts, and the similarity measures among these words are calculated. The similarity measure between the word A and B is the relative distance from their common ancestor to them in the hierarchy of the concepts [18].

Figure 5 illustrates how to calculate a similarity measure between the concepts "Book" and "Receipt". In the figure, the graph expresses a hierarchical structure of concepts. For example, the concepts "Concept" and "Receipt" are the root concept and a child of "Document of Acceptance", respectively. Then, the similarity measure between A and B is calculated as follows:

$$Similarity_{word}(A, B) = \frac{2c}{a + b}$$

where a and b are the depth of the concepts A and B from the root concept, respectively, while c is the depth of the common ancestor of A and B. If A and B are the same, i.e., the two words denote the same concept, $Similarity_{word}(A, B) = 1$, the maximum value. If the common ancestor of A and B is the root, i.e., there is no semantic intersection between them, the value is the minimum, 0.

Next, we define how to measure the semantic similarity of goal descriptions using this metric. After transforming a goal description in a case frame, we obtain the concepts of the words with which the slots of the case frame are filled, using the dictionary of hierarchal concepts. We calculate $Similarity_{word}$ to the words appearing in the goal descriptions. In the above example, using the dictionary, we can obtain $Similarity_{word}(\mathsf{Deliver}, \mathsf{Handle}) = 0.53$ and $Similarity_{word}(\mathsf{Deliver}, \mathsf{Send}) = 0.93$. Thus from the viewpoint of verbs, i.e., activities, the description "Send a receipt" is semantically closer to "Deliver a book" rather than "Handle a receipt". Let $cases(g)$ be a set of the concepts in the case frame representation of the description of the goal g, e.g., $cases$("Deliver a book") = { Deliver, Book }. And for the edge e in a goal graph, let $s(e)$ and $t(e)$ be a parent goal and its sub goal connected with e, respectively. The following predicate is defined for deciding if there are semantic relationships between the goals connected with the edge e or not:

$$Similarity(g_1, g_2) = \max_{\substack{w_1 \in cases(g_1) \\ w_2 \in cases(g_2)}} Similarity_{word}(w_1, w_2),$$

$$\mathsf{LowSemanticRelation}(e) = Similarity(s(e), t(e)) < Th_{sim}. \quad (1)$$

If Predicate (1) holds, we decide that the smell of Low Semantic Relation on the edge e is detected. Intuitively speaking, we calculate $Similarity_{word}$ between the words appearing in the descriptions of the parent goal and its sub goal connected with the edge e. The predicate says that these goals have less semantic similarity if its value is less than a certain threshold Th_{sim}. As shown in Fig. 3, we use this smell for finding poor refinement of the category "A sub goal does not contribute directly or is irrelevant".

3.3 Numbers of Sibling Edges

It is easy to calculate the number of sibling edges of an edge. We have two predicates to detect bad smells as follows:

$$\mathsf{ManySiblings}(e) = SiblingEdges(e) > Th_{high}, \quad (2)$$

$$\mathsf{FewSiblings}(e) = SiblingEdges(e) < Th_{low}. \quad (3)$$

$SiblingEdges(e)$ counts the sibling edges of e, including e itself, and it is defined as follows:

$$SiblingEdges(e) = |\{\, e' \mid s(e) = s(e') \,\}|.$$

As shown in Fig. 3, if the Predicate (2) on e holds, we decide that e has the bad smell of Many Siblings, which contributes to finding poor refinement of the category "A sub goal does not contribute directly or is irrelevant." The Predicate (3) on e detects the bad smell of Few Siblings, which leads to finding poor refinement of the category "There is a lacking sub goal" as a sibling edge of e.

3.4 Relative Depth of Goals

Shao *et al.* defined four measures to express the balance of a tree structure. We use the idea from one of them, *equitability* of probabilities of reaching leaf nodes, and adapted it to the measure that can express that a leaf has a depth from a root node on the same level with other leaves. They used the

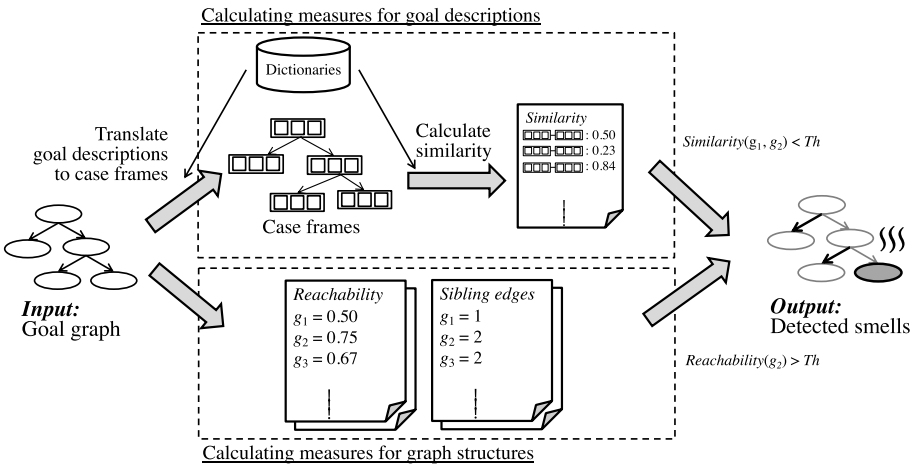


Fig. 6 Overview of bad smell detection.

idea of probability to reach a leaf from a root and Shannon's information entropy. Based on it, we consider that all of the leaf nodes are on the same level of depth or have the same granularity when their reachability probabilities from a root to leaves are the same. Under the assumption that each sibling edge has an equal probability at branching, i.e., the probabilities of reachability from a parent goal to its sub goals are the same, we can have a reachability probability of the edge e as $1/SiblingEdges(e)$. In the example of Fig. 4, the reachability probabilities from *Root* to B and from B to D are $1/2$ and $1/3$, respectively. Let $\langle e_i^g \rangle_{i=0,\ldots,n}$ be the edge sequence from a root goal to a goal g. We can define the reachability probability from a root to g is as follows:

$$Reachability(g) = \prod_{e \in \langle e_i^g \rangle_{i=0,\ldots,n}} \frac{1}{SiblingEdges(e)}.$$

In the example of Fig. 4, we obtain $Reachability(A) = 1/2$ and $Reachability(F) = 1/2 \cdot 1/3 \cdot 1/2 = 1/12$. We have the following expression to identify leaf goals having coarse granularity, i.e., having the possibility to be refined further comparing with the others:

$$\begin{aligned} \mathsf{CoarseGrainedLeaf}(g) &= \frac{Reachability(g)}{1/m} > Th_{reach} \\ &= Reachability(g) \cdot m > Th_{reach} \end{aligned} \tag{4}$$

where m is the number of leaf goals. If leaf goals are the result of uniform refinement, i.e., they have the same granularity, their reachability probability is also the same, i.e., $1/m$ where m is the number of the leaf goals. The Predicate (4) calculates the ratio of the reachability probability of a goal to the *ideal* one, i.e., $1/m$. If this value is greater than a certain threshold Th_{reach}, we detect Coarse Grained Leaf on the goal g as it is not refined well rather than the others. In the example of Fig. 4, the goals A and F have $(1/2) \cdot 5 = 2.5$ and $(1/12) \cdot 5 \simeq 0.42$, respectively. Thus, we can decide that the goal A is coarser-grained than the others and should be refined further.

4. Supporting Tool

We have implemented an automated tool supporting our technique as a plug-in of our already developed tool for the attributed goal-oriented analysis [19]. The smell detection process of our tool is shown in Fig. 6. The input of the tool is a goal graph. The tool analyzes the given goal graph in two different ways: the measurement of goal descriptions and that of graph structures. In the measurement of goal descriptions, the tool translates the goal descriptions in the input to case frames. Then, it applies the similarity measure to the obtained case frames for detecting the Low Semantic Relation smell. For analyzing goal descriptions and calculating their similarity, it uses a part of the EDR electronic dictionaries[†], which includes general-purpose Japanese dictionaries of case frames and hierarchical concepts. In the measurement of graph structures, the reachability probability and the number of sibling edges are measured for detecting Many Siblings, Few Siblings, and Coarse Grained Leaf smells, based on the structural analysis of the input. Finally, the tool aggregates the detected smells and annotates them to the goal graph as an output.

Figure 7 shows a screenshot of the tool. Our tool is embedded in a goal graph editor, which is implemented as a plug-in of Eclipse IDE[††]. In this figure, a requirements analyst is editing a goal graph, which is an excerpt goal model of a wholesale book seller [20]. As already noted in Sect. 1, our technique and tool are for goal models written in Japanese, and the descriptions in the displayed goal model are in Japanese. For the sake of understandability, we annotated the corresponding English translation for each goal description to the figure.

After exercising the smell detection feature, the editor highlights the nodes (goals) and the edges (refinement) having smells with red color (a). Here, two goals and two refinement edges are highlighted, including the edge of the

[†]https://www2.nict.go.jp/ipp/EDR/ENG/E_Intro.html
[††]https://www.eclipse.org/

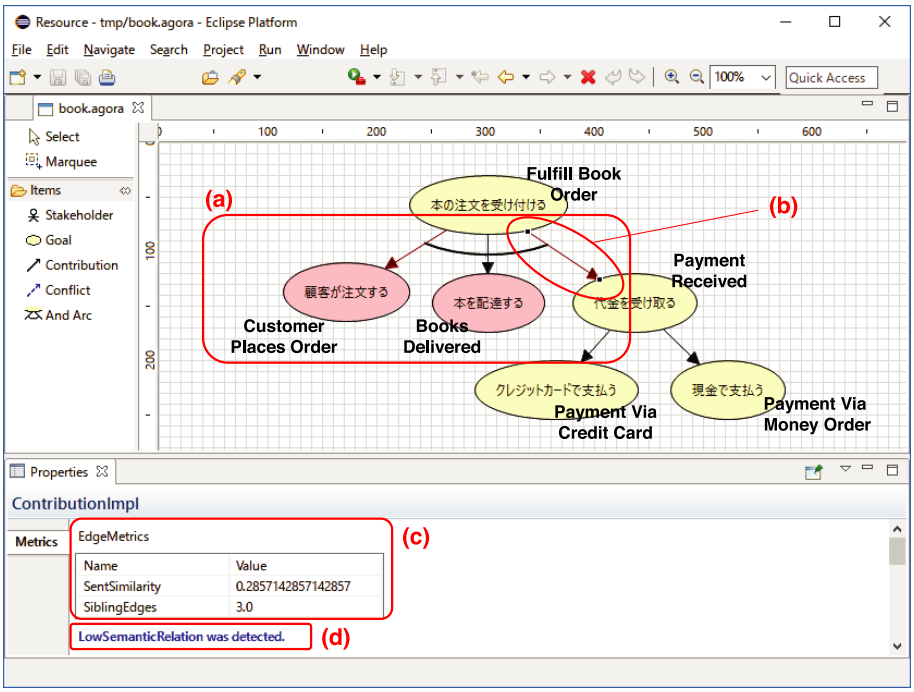


Fig. 7 Screenshot of the tool.

root goal "Fulfill Book Order" to the goal "Payment Received". When the analyst selects one of the smelly items (b), the associated metric values and the name of smells are shown in the Metrics properties view (c & d). In this example, one Low Semantic Relation smell is detected because the metric value of *Similarity* is low (0.285...), as shown in the Metrics view. The requirements analyst examines the detected smell instances and fixes the goal model if they are needed to be fixed. Since the development of goal models and the detection of smells are integrated into a single environment, the requirements analyst can proceed with developing the goal model while examining and selecting the detection results.

Note that the tool shows not a category of poor refinement but the results of the metrics evaluation, the category of the question level of Fig. 3. It helps the analysts to decide a category of poor refinement, i.e., the causes of poor refinement by referring to Fig. 3 and furthermore to improve the poor refinement. The technique to improve poor refinement is out of the scope of this paper but an interesting topic.

5. Experimental Evaluation

We discuss an experiment using the tool to evaluate our approach.

5.1 Aims of an Experiment

The aim of the experiment is to evaluate if our approach and the tool can indicate bad smells, which are symptoms of problems for requirements elicitation. Our experiment is comparative where we compare the correct set of poor refinement with the results of the indicated bad smells by the tool. However, it is difficult to create a complete and real correct set of poor refinement because the judgment of poor or not poor refinement is subjective to human analysts. Although it is difficult to have a complete correct set, we can have an approximate correct set that several analysts agree with, and we considered it as a correct set. Thus, our experimental procedure includes how to construct a correct set of poor refinement. This correct set may include errors, i.e., the occurrences of non-poor refinement and missing poor refinement. So we should check if our technique can detect these errors if any.

Our research questions to be validated in the experiment are as follows:

RQ$_1$ How many of the occurrences of poor refinement indicated by our technique are correct, i.e., what is the precision of our technique?

RQ$_2$ How many of the all correct occurrences of poor refinement can be indicated by our technique, i.e., what is the recall of our technique?

RQ$_3$ Can our technique find missing poor refinement from the correct set if any, and can it do non-poor refinement in the correct set if any?

5.2 Experimental Procedure

Figure 8 shows the overview of our experiment. We had four subjects who had experienced in GORA to detect poor goal refinement in examples of goal models to check the results of indicating bad smells by our tool. They were given two goal models: 1) Mobile Personal Emergency Response System (MPER) [21] and 2) Museum-Guide System (MGS) [22], and their goal descriptions include the aspects

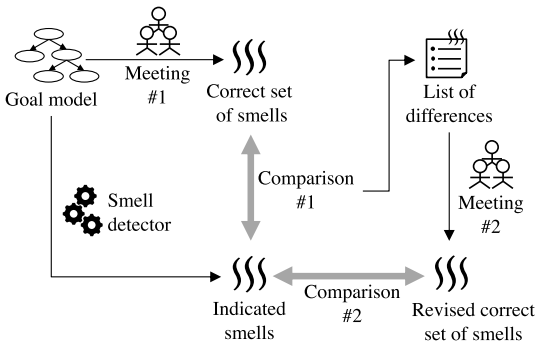


Fig. 8 Flow of the experimental evaluation.

of non-functional requirements and soft goals [5] as well as functional requirements. In the example of MGS, it has the goal "MGS prepare PDA brief simple information", which includes a non-functional requirement (easy for PDA users to read) and a soft goal aspect (brief simple). Our subjects were independently required to indicate the parts of poor goal refinement that may cause the problems of requirements elicitation. They were also required to express explicitly the rationales why the indicated parts were poor refinement. After performing this independent task, they had a face-to-face meeting (Meeting #1) to get a consensus of their indicated results that were different from each other. Their concerted results could be considered as the first version of a correct set of poor refinement. We compared it with the bad smells that our tool indicated (Comparison #1), and its result is shown as "List of differences" in the figure. The threshold values that we used in this experiment were $(Th_{sim}, Th_{low}, Th_{high}, Th_{reach}) = (0.7, 2, 5, 2)$, which were empirically decided based on the analysis of existing goal models, with the following rationales:

- $Th_{sim} = 0.7$ was used because we wanted to specify more candidates of poor refinement under the observed tendency when investigating existing resources. We investigated the distribution of our metric *Similarity* for the pairs of words occurring on existing goal graphs and thesaurus. As a result, we found that the similarity values were less than 0.5 for more than 80% of non-synonym word pairs and were more than 0.7 for most synonym word pairs.
- $Th_{low} = 2$ is intended to capture at least all extreme cases that a goal has only one sub goal.
- $Th_{high} = 5$ was defined as the upper bound of the storage capacity limits of human beings [10].
- $Th_{reach} = 2$ is used to capture the situation that the metric *Reachability* can be improved by adding new at least two sub goals to the identified goal.

We had a postprocess to refine the obtained first version of the correct set. Our tool indicated edges and leaf goals only while our subjects could do any parts in the goal models. For example, the subject indicated a goal as poor refinement, but the tool did the edge incoming to the goal indicated by them. Although the parts that the subjects and tool indicated are different, we should check if they indicated the same refinement or not. In case that such a discordance was found and/or that the subjects missed correct indications or identified wrong indications, the second face-to-face meeting (Meeting #2) was conducted by the same subjects to confirm their indications by the comparison of them with the tool's ones and an additional discussion to fix them if necessary. One of the authors facilitated this meeting, but he just suggested to the subjects the parts that they should discuss. After this meeting, we have got the correct set modified by the subjects. We compared it with the bad smells that the tool indicated (Comparison #2).

Table 2 Detection results

	MPER	MGS	Total
# goals	35	37	72
# tool indicated ($\|T\|$)	28	21	49
# correct set (subjects indicated) ($\|C\|$)	20	17	37
# the same as the correct set ($\|T \cap C\|$)	12	11	23
Precision ($\|T \cap C\|/\|T\|$)	0.43	0.52	0.47
Recall ($\|T \cap C\|/\|C\|$)	0.60	0.65	0.62

The research questions RQ_1 and RQ_2 can be responded with this comparison result, by calculating precision and recall values. Using the set of indications by the tool (T) and the correct set (C), these values are calculated as follows:

$$Precision = \frac{|T \cap C|}{|T|}, \qquad Recall = \frac{|T \cap C|}{|C|}.$$

To respond RQ_3, i.e., whether our tool can find the wrong indications by the subjects such as missing indications of poor refinement and indicating non-poor refinement, we explored the modifications of the first version of the correct sets, which were done during Meeting #2.

5.3 Results

Table 2 shows the number of the goals for each goal model (# goals), the number of indications by the tool (# tool indicated), the number of the correct set, i.e., indications by the subjects (# correct set), and the number of the indications same as the subjects, i.e., the correct indications by the tool (# the same as the correct set), and precision and recall values. In the example of MPER, our tool indicated 28 bad smells, and our subjects agreed that there were 12 occurrences of poor refinement.

As for the errors of the correct set, it included eight and five occurrences of missing poor refinement in MPER and MGS, respectively. Our subjects did not find them during Meeting #1 of Fig. 8. However, after checking the output of the tool, they noticed them as poor refinement during Meeting #2. It means that our tool can find poor refinement that human analysts might miss.

5.4 Discussion

In this subsection, first of all, we discuss the experimental results from the viewpoints of the research questions RQ_1,

Table 3 Analytic results on false positives

Category	# occurrences
Lacking domain knowledge	14
Lacking sufficient semantic analysis	8
Not well-formed sentences	2
Others	2
Total	(49 − 23 =) 26

Table 4 Analytic results on false negatives

Category	# occurrences
No consideration of contexts	5
Lacking sufficient semantic analysis	4
Lacking domain knowledge	2
Others	3
Total	(37 − 23 =) 14

RQ_2, and RQ_3, mentioned in Sect. 5.1.

5.4.1 Response to RQ_1

As shown in Table 2, the precision value of MPER is 0.43, which is less than 50%. Its major reason is that our technique did not consider domain-specific knowledge. To calculate the semantic similarity between a parent goal and its sub goals, we used the knowledge on general concepts, which resulted from the general dictionary of words in EDR. In the example of MPER, a parent goal "Emergency is detected" was refined into "Vital signs are processed". The word "vital sign" is domain-specific and is not included in our dictionary. Thus, our tool failed in deciding the existence of the semantic relationship between these goals. On the other hand, because MGS was more popular than MPER, it led to higher precision.

Table 3 shows the categories of the reasons why our tool indicated non-problematic refinement as poor one, i.e., analytic results of false positives. The total number of the occurrences of false positives was 26 = 49 ($|T|$) − 23 ($|T \cap C|$) as shown in Table 2. The quality of goal descriptions and natural language semantic analysis techniques are also crucial factors to improve our technique.

5.4.2 Response to RQ_2

The recall values were more than 50% in both of the goal models. The major reason is similar to the case of the precision values, i.e., the semantic similarity of goal descriptions. Although our subjects indicated the occurrences of the sub goals semantically less related to their parents, the tool decided that they were similar. Our metric on semantic similarity should be more elaborated. Another reason is that our technique did not consider contexts associated with goal refinement. In the example of MPER, our subjects indicated that the refinement of "Analyse vital signs" into "Detects emergency" was poor because there was the goal "Emergency is detected" as an ancestor goal, i.e., almost the same goal description as that of the sub goal, while the tool did not. Although the meaning of the sub goal was "Detects emergency *with sensed data of vital sign*" in the context of its parent goal "Analyse vital signs", our subjects indicated that this leaf goal was not so concrete and suggested to improve its goal descriptions or to refine it further. We need to develop semantic processing of goal descriptions considering such a context, e.g., considering not only a parent goal and its *direct* sub goals but also its ancestors and descendants.

Table 4 summarizes the categories of the reasons why our tool could not detect poor refinement, i.e., analytic results of false negatives. The total number of the occurrences of false negatives was 14 = 37 ($|C|$) − 23 ($|T \cap C|$).

5.4.3 Response to RQ_3

As mentioned in the previous subsection, the tool could find totally 13 occurrences (8 in MPER plus 5 in MGS) of poor refinement that our subjects did miss. Considering that our correct set has 37 poor refinement ($|C|$ in Table 2), our subjects missed 35% (= 13/37) of poor refinement to be noticed. It means that human analysts can miss about 1/3 of the occurrences of poor refinement even when they review a goal model to find poor refinement. Our tool can be really useful to find these missing occurrences.

Table 5 shows the results for each category of poor refinement. For example, in the correct set our subjects had 22 occurrences of poor refinement of the category "A sub goal does not contribute directly or is irrelevant", and 14 of them were successfully detected by Low Semantic Relation. The smell Many Siblings using the number of sibling edges could not contribute to detecting poor refinement of this category at all, so we should develop other measures for this category. The metric of the number of sibling edges is also used in Few Siblings for detecting "There is a lacking sub goal", but did not result in so sufficient detection, seeing its low recall value. We also need new measures for this category.

In Tables 2 and 5, note that the number of the correct detection by the tool, i.e., 14+0+3+4 = 21 was not equal to 23 shown in Table 2. In these two occurrences (23−21 = 2), the category of poor refinement in the correct set was different from the category to be detected by the measure. These two were correctly detected by Low Semantic Relation but our subjects indicated them as "There is a lacking sub goal". They should have been detected as "A sub goal does not contribute directly or is irrelevant" according to our arguments in Sect. 3 and Fig. 3. However, "A sub goal does not contribute directly or is irrelevant" may lead to "There are a lacking sub goal" as a result. That is to say, our classification is not exclusive.

5.5 Threats to Validity

External Validity. External validity is related to the generality of the obtained conclusions. In this experiment, although we used only two goal models, their problem domains were different, and we believe that they covered varieties of the problem domain in a certain degree. The sizes of

Table 5 Detection results (2)

Category	Correct set ($\|C\|$)	Smell	Tool indicated ($\|T\|$)	$\|T \cap C\|$	Precision	Recall
A sub goal does not contribute directly or is irrelevant	22	Low Semantic Relation	28	14	0.50	0.64
		Many Siblings	0	0	N/A	0.00
There is a lacking sub goal	9	Few Siblings	10	3	0.30	0.33
A leaf is not concrete	6	Coarse Grained Leaf	11	4	0.36	0.67

the goal models were smaller than those in practical levels. Our technique detected real 23 occurrences, and it means that there may be many occurrences in spite of smaller sizes of goal models. Our tool can work well on the smaller sizes of models. The scalability of our approach, e.g., performance, should be checked as one of future research agendas.

Construct Validity. This validity shows that our observed data can really reflect our obtained conclusions. We assessed our technique by the differences between the correct set and the occurrences the tool indicated. The possibility of this threat is the quality of the correct set. The correct set was made by our subjects, so its quality depended on their skills and knowledge. Another factor is the quality of the translations of English goal descriptions into Japanese, and it may lead them to misread. However, we took four subjects experienced in goal modeling, and any decisions resulted from their consensus. So, we consider that the quality of the correct set was sufficiently high.

Internal Validity. We should explore the factors affecting the obtained results other than those of our approach. One of the possibilities of these factors is the bias at Meeting #2 in Fig. 8 that our subjects might have taken their decisions advantageous to our tool. However, they rejected many occurrences ($26 = 49 - 23$ as shown in Table 2) that the tool indicated as poor refinement, and this fact shows the fairness of our subjects. In addition, we took a process of agreement of all subjects to change their decisions.

6. Related Work

There are wide varieties of studies to assist in improving the quality of requirements models, including goal models. Kaiya *et al.* [23] proposed the measures of quality characteristics, and they attach some attribute values such as contribution and preference values attached to goal models. By using the values, the proposed technique measures the correctness, completeness, unambiguity, etc. of goal models. This technique can be extended to reason the changeability of goal models [24], [25]. Giorgini *et al.* [26] developed two measures contribution and denial values of goal achievement, to measure the quantitative quality of goal models. On goal models written in *i** and KAOS, the metrics of their complexity were also developed [27], [28], and almost all of them calculate the numbers of syntactic components of goal models such as agents, edges, etc. Silva *et al.* [29] proposed visualization quality of *i** goal models to improve their understandability. As mentioned above, although there are several work to define metrics of quality and complexity of goal models from various viewpoints, there are none of the metrics on the quality of goal refinement from structural and semantical viewpoints, and they were not for detecting poor refinement. As for quality from a semantical viewpoint, note that Kaiya *et al.* [30] proposed metrics of correctness, completeness, unambiguity, etc. for requirements written in a natural language using mapping words into domain ontology as semantic concepts. To enhance semantic similarity in our approach, using domain ontology is one of the promising approaches.

The second category is the support of goal refinement of higher quality. Refinement patterns can guide goal refinement in KAOS [2]. They are on the logical basis of temporal logic, and they are not for detecting poor refinement. More concretely, refinement following the suitable refinement patterns can guarantee logically achievement of a parent goal by sub goals, but they did not consider the other aspects of quality such as the granularity of refined goals. Our approach focused on bad smell detection of poor goal refinement and not on the improvement. The idea of refinement patterns is complementary to ours when we catalog bad smells and their resolutions as patterns.

The last category is related to the quality of natural language sentences. Our approach did not evaluate goal descriptions themselves but their semantic relationships. To measure the quality of goal descriptions, we should adopt natural language processing techniques to analyze the ambiguity and vagueness of natural language sentences. Yang *et al.* [31], [32] discussed nocuous ambiguity such as the scope of conjunctives "and" and "or", and automated to detect its occurrences. They applied their approach to usual natural language sentences, not goal descriptions. Goal descriptions are short sentences and may be incomplete. Although their approach is complementary to ours, we cannot apply it as it is. We need some techniques to supplement descriptions with adding missing words [33] and then apply them to the supplemented descriptions.

Bad smells, symptoms that the application of *refactoring* is needed, have been used mainly in source code level [34], but the framework and terminology are not limited to the area related to source code. The framework, i.e., detecting quality problems of software artifacts as smells and fixing them by refactoring to improve the quality with preserving their core aspects, is straightforward and is applicable to software artifacts in the scope of requirements engineering, such as use case models [35], [36], use case descriptions [37], feature models [38], and natural language documents [39], [40]. Our approach can be classified as the same category; it regards the detection of problematic portions in goal graphs as bad smells.

7. Conclusion

This paper discussed the automated technique to detect the occurrences of poor refinement in a goal model. After clarifying bad smells of poor refinement, we defined two types of measures: structural characteristics of a goal model as a graph and semantic aspects of goal descriptions. Our experimental results showed that our direction is promising and some improvement is necessary.

Future work can be listed up as follows:

- More experimental analyses on goal models of wide varieties of problem domains and in practical level.
- Enhancing metrics, in particular for detecting bad smells of the first category, i.e., a sub goal does not contribute directly or is irrelevant, including more elaborated natural language processing.
- Technique to decide threshold values of metrics systematically. As for our current metrics, it is significant how to decide their suitable threshold values in a more systematic and reliable way.
- Technique to support the improvement of detected bad smells, so-called *refactoring* of goal refinement.
- More elaborated technique to process the goal graphs having various types of nodes and their relationships other than "achievement" relationship, e.g., quality goals, tasks, resources as nodes, and qualification, "needed-by" as relationships in iStar 2 [4].
- Technique to utilize reusable assets. For example, the hierarchical relationship between NFR concepts appearing in NFR framework [41] can be considered as "achievement" relationship between non-functional requirements categories, and thus we can use it as a basis of goal refinement on non-functional requirements. Another example can be Refinement Patterns [2].
- Combining the techniques of domain ontology [30], [42] with our approach, in order to deal with domain-specific knowledge. In the construction of domain ontology, several techniques [43], [44] can also be considered.
- Technique to apply the concepts of bad smells and their detection to other modeling techniques having refinement or hierarchical decomposition mechanisms such as Work Breakdown Structure [45] and Structured Analysis [46].
- Technique to apply to a combined approach of GORA and use case modeling [47]. We explore a technique to support refinement of a use case model by detecting bad smells of a combined goal graph and by improving them, and vice versa.

Acknowledgments

This work was partly supported by JSPS Grants-in-Aid for Scientific Research Nos. JP21K11823, JP18K11237, and JP18K11238.

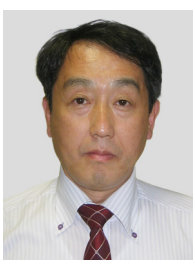
References

[1] K. Asano, S. Hayashi, and M. Saeki, "Detecting bad smells of refinement in goal-oriented requirements analysis," Workshop Proc. 36th International Conference on Conceptual Modeling (ER 2017), Lecture Notes in Computer Science, vol.10651, pp.122–132, 2017.
[2] A. van Lamsweerde, Requirements Engineering: From System Goals to UML Models to Software Specifications, Wiley, 2009.
[3] E.S. Yu, "Social modeling and i*," Conceptual Modeling: Foundations and Applications, Lecture Notes in Computer Science, vol.5600, pp.99–121, 2009.
[4] F. Dalpiaz, X. Franch, and J. Horkoff, "iStar 2.0 language guide." https://arxiv.org/abs/1605.07767, 2016.
[5] E. Yu, "Towards modeling and reasoning support for early-phase requirements engineering," Proc. 3rd IEEE International Symposium on Requirements Engineering (RE'97), pp.226–235, 1997.
[6] H. Nakagawa, A. Ohsuga, and S. Honiden, "A software evolution method based on goal-oriented requirements description forming (in Japanese)," IPSJ Journal, vol.53, no.10, pp.2328–2344, 2012.
[7] J. Nomura, R. Naruse, K. Hokamura, N. Ubayashi, T. Shidai, and A. Iwai, "A goal-oriented requirements analysis method for distributed autonomous transportation systems," IEICE Technical Report, vol.109, no.307, pp.13–18, 2009.
[8] IEEE, "IEEE recommended practice for software requirements specifications," tech. rep., IEEE Std. 830-1998, 1998.
[9] V. Basili, C. Caldiera, and D. Rombach, "Goal, question, metric paradigm," Encyclopedia of Software Engineering, vol.1, pp.528–532, 1994.
[10] N. Cowan, "Metatheory of storage capacity limits," Behavioral and brain sciences, vol.24, no.1, pp.154–176, 2001.
[11] G.A. Miller, "The magical number seven, plus or minus two: Some limits on our capacity for processing information," Psychological Review, vol.63, no.2, pp.81–97, 1956.
[12] K.T. Shao, "Tree balance," Systematic Biology, vol.39, no.3, pp.266–276, 1990.
[13] C.J. Fillmore, "Lexical entries for verbs," Foundations of Language, pp.373–393, 1968.
[14] A. Ohnishi, "Software requirements specification database based on requirements frame model," Proc. 2nd International Conference on Requirements Engineering (ICRE 1996), pp.221–228, 1996.
[15] C. Rolland and C. Proix, "A natural language approach for requirements engineering," Proc. 4th International Conference on Advanced Information Systems Engineering (CAiSE 1992), Lecture Notes in Computer Science, vol.593, pp.257–277, 1992.
[16] M. Saeki and H. Kaiya, "Supporting the elicitation of requirements compliant with regulations," Proc. 20th International Conference on Advanced Information Systems Engineering (CAiSE 2008), Lecture Notes in Computer Science, vol.5074, pp.228–242, 2008.
[17] R. Nakamura, Y. Negishi, S. Hayashi, and M. Saeki, "Terminology matching of requirements specification documents and regulations for compliance checking," Proc. 8th IEEE International Workshop on Requirements Engineering and Law (RELAW 2015), pp.10–18, 2015.
[18] Z. Wu and M. Palmer, "Verbs semantics and lexical selection," Proc. 32nd Annual Meeting on Association for Computational Linguistics (ACL 1994), pp.133–138, 1994.
[19] M. Saeki, S. Hayashi, and H. Kaiya, "A tool for attributed goal-oriented requirements analysis," Proc. 24th IEEE/ACM International Conference on Automated Software Engineering (ASE 2009), pp.670–672, 2009.
[20] S. Liaskos, S.A. McIlraith, S. Sohrabi, and J. Mylopoulos, "Integrating preferences into goal models for requirements engineering," Proc. 18th IEEE International Requirements Engineering Conference (RE'10), pp.135–144, 2010.
[21] D. Mendonça, R. Ali, and G.N. Rodrigues, "Modelling and analysing contextual failures for dependability requirements," Proc.

9th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS 2014), pp.55–64, 2014.
[22] R. Ali, F. Dalpiaz, and P. Giorgini, "A goal-based framework for contextual requirements modeling and analysis," Requirements Engineering, vol.15, no.4, pp.439–458, 2010.
[23] H. Kaiya, H. Horai, and M. Saeki, "AGORA: Attributed goal-oriented requirements analysis method," Proc. 10th Anniversary IEEE Joint International Requirements Engineering Conference (RE'02), pp.13–22, 2002.
[24] S. Hayashi, D. Tanabe, H. Kaiya, and M. Saeki, "Impact analysis on an attributed goal graph," IEICE Trans. Inf. & Syst., vol.E95-D, no.4, pp.1012–1020, 2012.
[25] D. Tanabe, K. Uno, K. Akemine, T. Yoshikawa, H. Kaiya, and M. Saeki, "Supporting requirements change management in goal oriented analysis," Proc. 16th IEEE International Requirements Engineering Conference (RE'08), pp.3–12, 2008.
[26] P. Giorgini, J. Mylopoulos, E. Nicchiarelli, and R. Sebastiani, "Reasoning with goal models," Proc. 21st International Conference on Conceptual Modeling (ER 2002), Lecture Notes in Computer Science, vol.2503, pp.167–181, 2002.
[27] P. Espada, M. Goulão, and J. Araújo, "A framework to evaluate complexity and completeness of KAOS goal models," Proc. 25th International Conference on Advanced Information Systems Engineering (CAiSE 2013), Lecture Notes in Computer Science, vol.7908, pp.562–577, 2013.
[28] C. Gralha, J. Araújo, and M. Goulão, "Metrics for measuring complexity and completeness for social goal models," Information Systems, vol.53, pp.346–362, 2015.
[29] L.F. Silva, A. Moreira, J. Araújo, C. Gralha, M. Goulão, and V. Amaral, "Exploring views for goal-oriented requirements comprehension," Proc. 35th International Conference on Conceptual Modeling (ER 2016), Lecture Notes in Computer Science, vol.9974, pp.149–163, 2016.
[30] H. Kaiya and M. Saeki, "Ontology based requirements analysis: Lightweight semantic processing approach," Proc. 5th International Conference on Quality Software (QSIC 2005), pp.223–230, 2005.
[31] H. Yang, A.N. De Roeck, V. Gervasi, A. Willis, and B. Nuseibeh, "Extending nocuous ambiguity analysis for anaphora in natural language requirements," Proc. 18th IEEE International Requirements Engineering Conference (RE'10), pp.25–34, 2010.
[32] H. Yang, A. Willis, A.N. De Roeck, and B. Nuseibeh, "Automatic detection of nocuous coordination ambiguities in natural language requirements," Proc. 25th IEEE/ACM International Conference on Automated Software Engineering (ASE 2010), pp.53–62, 2010.
[33] Y. Negishi, S. Hayashi, and M. Saeki, "Supporting goal modeling for eliciting regulatory compliant requirements," Proc. 19th IEEE Conference on Business Informatics (CBI 2017), pp.434–443, 2017.
[34] M. Fowler, Refactoring: Improving the Design of Existing Code, Addison Wesley, 1999.
[35] W. Yu, J. Li, and G. Butler, "Refactoring use case models on episodes," Proc. 19th IEEE International Conference on Automated Software Engineering (ASE 2004), pp.328–335, 2004.
[36] J. Xu, W. Yu, K. Rui, and G. Butler, "Use case refactoring: a tool and a case study," Proc. 11th Asia-Pacific Software Engineering Conference (APSEC 2004), pp.484–491, 2004.
[37] Y. Seki, S. Hayashi, and M. Saeki, "Detecting bad smells in use case descriptions," Proc. 27th IEEE International Requirements Engineering Conference (RE'19), pp.98–108, 2019.
[38] V. Alves, R. Gheyi, and T. Massoni, "Refactoring product lines," Proc. 5th International Conference on Generative Programming and Component Engineering (GPCE 2006), pp.201–210, 2006.
[39] E.R. Harold, Refactoring HTML: Improving the Design of Existing Web Applications, Addison-Wesley, 2012.
[40] L. Aversano, G. Canfora, G. De Ruvo, and M. Tortorella, "An approach for restructuring text content," Proc. 35th International Conference on Software Engineering (ICSE 2013), pp.1225–1228, 2013.
[41] L. Chung, B. Nixon, E. Yu, and J. Mylopoulos, Non-Functional Requirements in Software Engineering, Kluwer Academic Publishers, 1999.
[42] M. Shibaoka, H. Kaiya, and M. Saeki, "GOORE: Goal-oriented and ontology driven requirements elicitation method," Workshop Proc. 26th International Conference on Conceptual Modeling (ER'07), Lecture Notes in Computer Science, vol.4802, pp.225–234, 2007.
[43] M. Kitamura, R. Hasegawa, H. Kaiya, and M. Saeki, "An integrated tool for supporting ontology driven requirements elicitation," Proc. 2nd International Conference on Software and Data Technologies (ICSOFT 2007), pp.73–80, 2007.
[44] J. Kato, M. Saeki, A. Ohnishi, S. Hayashi, H. Kaiya, and S. Yamamoto, "Integration of thesauruses for requirements elicitation (in Japanese)," IPSJ SIG Technical Reports, vol.2021-SE-208, no.4, pp.1–8, 2021.
[45] G. Haugan, Effective Work Breakdown Structures, Management Concepts, Inc., 2002.
[46] T. DeMarco, Structured Analysis and System Specification, Yourdon Press, 1978.
[47] C. Rolland and C. Salinesi, "Supporting requirements elicitation through goal/scenario coupling," Conceptual Modeling: Foundations and Applications, Lecture Notes in Computer Science, vol.5600, pp.398–416, 2009.

Shinpei Hayashi is an associate professor in School of Computing at Tokyo Institute of Technology. He received a B.E. degree in information engineering from Hokkaido University in 2004. He also respectively received M.E. and D.E. degrees in computer science from Tokyo Institute of Technology in 2006 and 2008. His research interests include software evolution and software development environment.

Keisuke Asano received a B.E. and M.E. degrees in computer science from Tokyo Institute of Technology in 2015 and 2017, respectively. His research interests include requirements engineering and quality assurance for software engineering artifacts.



Motoshi Saeki received D.E. degree in computer science from Tokyo Institute of Technology, in 1983. He was a professor in School of Computing at Tokyo Institute of Technology. He is currently a professor in Department of Software Engineering at Nanzan University. His research interests include requirements engineering, software design methods, and computer supported cooperative work (CSCW).