ソフトウェア工学 140－1
（２００３．３．６）

ゴール指向分析法とユースケース法を用いた要求獲得支援法

綿引 健二　　佐伯 元司
東京工業大学 大学院
情報理工学研究科 計算工学専攻

概 要

ゴール指向分析法とユースケース法は，ともに実用的な要求獲得方法論として知られている．概して，ゴール指向分析法は分析対象の制約の獲得に，ユースケース法は具体的な振る舞いの獲得に適しており，両方法論は相補的な関係にある．このため，両方法論を統合することによって，より有効な要求獲得方法論を得ることができると考える．そこで本稿では，ゴール指向分析法とユースケース法を効果的に統合した新しい要求獲得方法論を提案する．本方法論では，分析対象の制約をゴール指向で，対象の振る舞いを階層化したユースケースによって詳細化を行う．また，ゴールとユースケースを明確に関連付けることによって，両者を互いに連携させて分析を行う．さらに，本方法論を２つの事例に適用し有効性の評価を行った．

Combining Goal-Oriented Analysis and Use Case Analysis

Kenji Watahiki　　*Motoshi Saeki*
Department of Computer Science,
Graduate School of Information Science and Engineering,
Tokyo Institute of Technology

Abstract

Goal-oriented analysis and use case analysis are well known requirements analysis methods and are putting into practice. Roughly speaking, goal-oriented methods are suitable for eliciting constraints to a system and use case analysis methods elicit concrete system behavior. Thus these methods are complementary and their integration into a new method allows us to get a more powerful requirements elicitation method. This paper proposes a new method where both of the methods are amalgamated. In our method, constraints to the system are refined by goal-oriented style, while system behavior are described with hierarchical use cases. Since a use case is made relate to goals during our elicitation processes, the decomposition of goals and use cases are complementally supported. Furthermore we applied our method to a couple of development projects and assessed its effectiveness.

1 Introduction

Requirements analysis, especially requirements elicitation is one of the most significant phases to develop software of high quality since it is the first phase in software development processes. To support this phase, we have many methods or methodologies. Above all, a family of goal-oriented methods such as KAOS [6] and i* [10], scenario analysis and use case analysis methods have been widely used in practical situation, and successful results have been obtained. For example, Potts and Anton reported the experiences in applying goal-oriented method to industries [3]. CREWS project analyzed the usage of scenario analysis methods in industries [1] and Pohl et al surveyed the usage of scenario analysis methods in industries [9]. Although, as excellent previous case studies reported, they can be considered as promising methods for requirements elicitation, they are not all-powerful, i.e. they have both advantages and shortcomings.

Goal-oriented methods are a technique where a goal to be achieved is hierarchically decomposed into more concrete sub-goals in a top-down manner. A resulting artifact is an AND-OR graph whose nodes are the decomposed goals and whose edges express decomposition. Since the steps of goal-oriented methods begin with customers' needs, the contents of the goals are more abstract and are frequently described as constraints. As Yu et al. showed [5], it could be applied to the elicitation of non-functional requirements as constraints to be achieved. On the other hand, in scenario analysis and use case analysis, an analyst describes the sequence of actions and/or of interactions among a system and users. It allows us to capture the whole image of the system concretely. That is to say, goal-oriented methods are suitable for eliciting constraints to a system and scenario and use case analysis methods elicit concrete system behavior. Thus these methods are complementary and their integration into a new method allows us to get a more powerful requirements elicitation method.

Anton [2] or Rolland et al. [7] reported the case studies of the experiences in which both a goal-oriented method and scenario analysis were applied, and they discussed the benefits. However, in these case studies, both methods were not integrated or amalgamated into a method. Rather the methods were just used in requirements elicitation activities, and none of new effective methods was established. Santander et al. proposed a method for deriving use cases from an organizational model of i* (strategic dependency model and strategic rationale model) and the final artifact in the method is a use case model [8]. In this method, use case analysis does not contribute to decompose the goals, i.e. to construct the organizational model.

In goal-oriented methods, we specify constraints and tasks as goals and it leads us to difficulties in decomposing the goal. i* [10], proposed by Yu et al., provided the solution of this mixture, i.e. i* separated the constraints from tasks on notation. In our approach, we explicitly distinguish constraints from task descriptions and adopt two methods, one is suitable for constraint decomposition, i.e. goal-oriented analysis and the other is for task decomposition, i.e. use case analysis. Their combination leads to complementary supports of decomposing constraints and tasks. In this paper, we analyzing the characteristics of goal-oriented analysis and use case one, and propose a technique for amalgamating them into a new method so that it makes the best use of their merits.

The rest of the paper is organized as follows. In the next section, we comparatively discuss the characteristics of goal-oriented analysis and use case one. Sections 3 and 4 present the overview of our new method and its experimental assessment respectively. In particular, section 4 includes the guidelines to use our method effectively, which have been extracted from the case study. Section 5 is a concluding remark.

2 Goal-Oriented Analysis and Use Case Analysis

Roughly speaking, requirements descriptions can be categorized into constraints and actions. For example, the requirement "calculating accurately a price including VAT (value added tax)" can be divided into the action "calculating a price including VAT" and the constraint "the calculation should be accurate". Focusing these facets of requirements, we can consider two dimension of decomposition; one is a constraint axis and the other is an action one.

Figure 1 shows two-dimensional directions of requirements decomposition and refinement. For example, the requirement "calculating accurately a

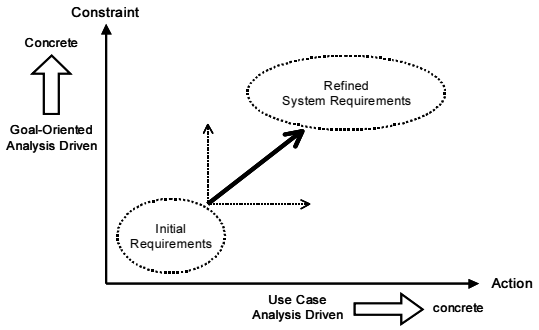

Figure 1: Two-dimensional Directions of Requirements Decomposition and Refinement

price including VAT" is decomposed into the following two requirements, by focusing on how to calculate a price including VAT; 1) calculating accurately VAT, and 2) summing up accurately the VAT and the price of the good. In this case, we decompose the original requirement along horizontal direction, i.e. an action axis. On the other hand, when we focus on the constraint "accurately" and refine the original one into "calculating a price including VAT to the first decimal place", it can be considered as a constraint refinement and along the vertical direction. In requirements elicitation processes, we need to refine vague requirements from customers along both of these two axes completely. However, in general cases, since an analyst goes ahead his analysis activities without recognizing both of the two axes, he gets insufficient requirements elicitation in the sense that it is incomplete in either of the axes. Actions and constraints are essentially different and it is not suitable for us to apply the same analysis method to requirements elicitation without separating the facets of actions from the constraints ones.

Goals appearing in goal-oriented analysis are suitable for describing constraints. In particular, goal-oriented analysis is fit for eliciting and refining the constraints that should be satisfied just before, during or after an action or a task is performed. These conditions are called pre conditions, invariants and post conditions respectively. On the other hand, since we specify the sequences of actions in a use case, use case analysis is suitable for modeling the contents of tasks, which consists of actions. Furthermore, we can hierarchically decompose and refine a use case into more fine-grained use cases (called sub use case). Suppose that constraints such as pre, post conditions and invariants are associated with a use case denoting a task. The hierarchical decomposition of the use case allows us to decompose the pre, post conditions and invariants along with decomposing the use case and we can use it to decompose and refine the goals denoting these conditions. That is to say, associating use cases with goals and vice versa, and decomposing and refining both of them simultaneously, we can elicit concrete requirements along with the two axes shown in Figure 1 and as a result, we achieve requirements elicitation of high quality. We call this association between a use case and a goal a *context link*.

In the next section, we propose the analysis method where goal-oriented analysis and use case analysis are integrated.

3 Our Proposed Method

In this section, we illustrate our proposed method with a simple example of analyzing a supporting system for a small library in a university laboratory.

3.1 Overview of Our Method

Roughly speaking, as shown in Figure 2, our method is divided into the following three steps.

1) Identifying Initial Goals and Use Cases

   First of all, an analyst identifies the goals and use cases which are starting points of refinement activities. We call them initial goals and initial use cases respectively. They can be extracted from information that he gets from customers and/or current situation before starting this step.

2) Refining Goals and Use Cases

   This step is a major one of our method and it consists of five sub-steps. The analyst iterates these sub-steps cyclically to gradually make the goals and the use cases more concrete. As this step proceeds, hierarchical structures of goals and use cases grow up by means of generating sub goals and sub use cases. After finishing the step 2), he gets two graphs denoting hierarchical structures of goals and use cases respectively and links for relating each node in a graph to a node or nodes in another graph. These links represent the relationship between actions and constrains, i.e. context links mentioned in section 2.

3) Selecting System Requirements

   From the graphs, requirements related to the software system to be developed are extracted. These system requirements are used for composing software requirements specifications following standards such as IEEE 830-1998.

In the next subsections, we explain the details of the above steps together with the simple example.

3.2 Identifying Initial Goals and Use Cases

The first step is identifying initial goals and use cases, which are used for starting points of refinement, i.e. setting up root nodes of the graphs. An analyst recognizes the business tasks that the newly developed system will have a great effect on and lists up these tasks as candidates of the initial use cases. He does not describe the action sequences of the use cases yet, but extracts pre, post conditions and invariants of the recognized tasks as constraints. These constraints are considered as initial goals and he establishes context links between the initial use cases and the initial goals.

Let's explain this step more concretely by using the example. Consider the tasks that will be affected when the support system of our small library is completed and employed. The analyst extracts the tasks "Manage literature information"

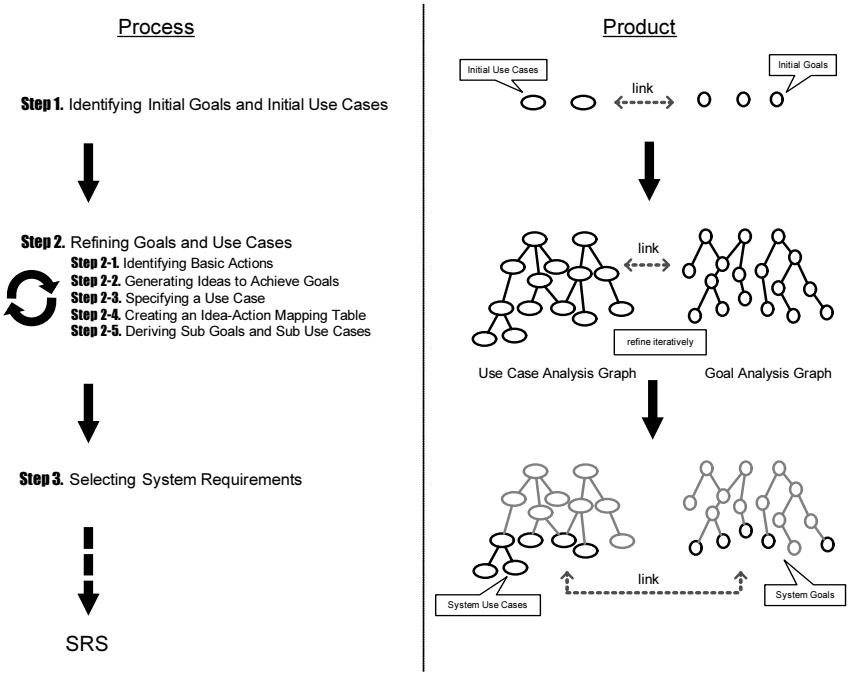

Figure 2: Overview of Our Method

and "Manage the transaction information of borrowing literatures" as initial use cases. For each of the use cases, he identifies the constraints by making interview with users of the library and by investigating the business of the librarians. For example, he extracted the initial goals "The literature information should be correct" and "The librarians can manage information efficiently" for the use case "Manage literature information", and creates context links between them and this initial use case.

3.3 Refining Goals and Use Cases

This step consists of five sub steps as follows;

2-1) Identifying basic actions

The analyst picks up the use case that is not specified yet, from the use case graph, and identifies actions that its action sequence is composed of. Suppose that he picks up "Manage literature information". To clarify its action sequence, he identifies the basic actions "Register literature information", "Edit literature information" and "Delete literature information", which seem to be included in its action sequence.

2-2) Generating ideas to achieve goals

For the use case that he picked up in the step 2-1), he focuses on the goals that the use case is related to with context links. He generates ideas how to achieve these goals, e.g. by using idea generation methods such as KJ method. For example, he has the goals "The literature information should be correct" and "The librarians can manage information efficiently" for the use case. To achieve the goal "The librarians can manage information efficiently", he generates two ideas (I1) Simple operations and (I2) Can edit information by a browser. As for the goal "The literature information should be correct", he generates four ideas to achieve it as follows; (I3) Register information accurately, (I4) Possible to edit information afterward, (I5) Avoid deleting information mistakenly and (I6) Prohibit anyone except librarians from having the operations.

2-3) Specifying a use case

This step is for specifying the action sequences in the use case that was picked up in the step 2-1). To describe the action sequences, the identified basic actions are very helpful. Figure 3 shows this process. For example, the analyst produces the following description for the use case "Manage literature information".

Use Case Manage literature information
objective : A librarian manages literature information.
actor : librarian
pre condition : none

post condition : The information of all of the literatures on the book shelves is recorded.
normal action sequence :
A1. A librarians registers literature information.
A2. A librarian edits literature informationwith a browser.
A3. A librarian deletes literature information.
alternate action sequence :
none

The ideas generated in the step 2-2) contribute to form the use case descriptions. For example, the action of editing literature information is done using a browser, because the idea (I2) "Can edit information by a browser" suggests how to perform the action.

2-4) Creating an Idea-Action Mapping Table

This step is a significant step to derive sub use case and sub goals. The idea-action mapping table represents the correlations between the ideas and the actions appearing in the use case. As shown in Figure 4, the column stands for the generated ideas and the row is for the actions. If the action A_j is necessary to achieve the idea I_i and the achievement needs to impose the constraint C on the action, we write the constraint C in the (i, j) cell. If no constraints are necessary to be imposed, we put the symbol "NC" on the cell. In the figure, the idea I2 can be achieved by the action A1 under the situation when "Constraint1" holds. Table 1 illustrates the table of our example. The idea I5 "Avoid deleting information mistakenly" can be achieved by means of performing the action A3 under the condition that the delete operation causes less human errors.

2-5) Deriving sub goals and sub use cases

Based on the idea-action mapping table, the analyst can derive sub goals and sub use cases systematically. All of the actions in the table can be considered as sub use cases of the use case which was picked up in the step 2-1).

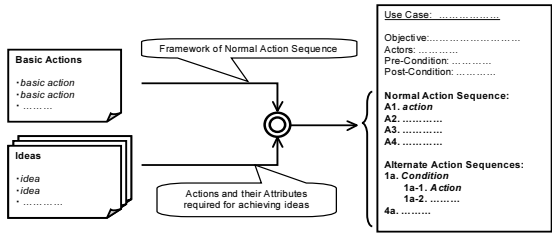

Figure 3: Specifying a Use Case

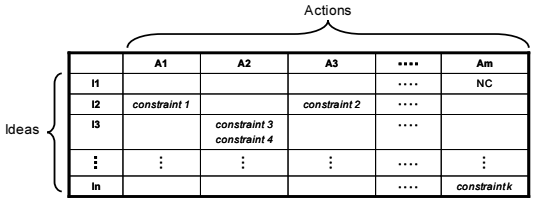

Figure 4: Idea-Action Mapping Table

That is to say, the actions A_i $(1 \leq i \leq m)$ appearing in Figure 4 are sub use cases. Some constraints are specified in a (i, j) cell, they can be considered as sub goals of the goal linked to the use case with the context link. Figure 5 illustrates how to grow up the goal graphs and use case ones by using the Idea-Action Mapping Table. In the figure, sub nodes A1, ..., Am are sub use cases of UC and they result from the first row of the Idea-Action Mapping Table. Since the (1, 2) cell has a constraint C1, we have a sub goal C1 of Gx, which is linked to the use case UC. The reason why C1 is the sub goal of Gx is that the idea I1 is related to Gx. Turn back the simple example. The constraint, extracted in the step 2-4), "less human errors" is one of the sub goal of the goal "The literature information should be correct" which is linked to the use case "Manage literature information" with a context link. In addition, the analyst establishes new context links between the derived sub use cases and sub goals. Figure 6 shows derived sub use cases and sub goals.

3.4 Selecting System Requirements

The produced use case graphs and goal ones includes the parts that are not relevant to a system to be developed, e.g. human tasks, business processes and so on. This final step selects use cases and goals relevant to the system from the graphs. Roughly speaking, the analyst focuses on the use cases whose actor is the system or the component of the system. After selecting the use cases (called system use cases), he traces context links from the selected use cases to goals. The goals to which the

Table 1: an Example of Idea-Action Mapping Table

	A1	A2	A3
I1	simple operations	simple operations	simple operations
I2		browser based user interface	
I3	correctly		
I4		NC	
I5			less human errors
I6	allow only librarians to operate	allow only librarians to operate	allow only librarians to operate

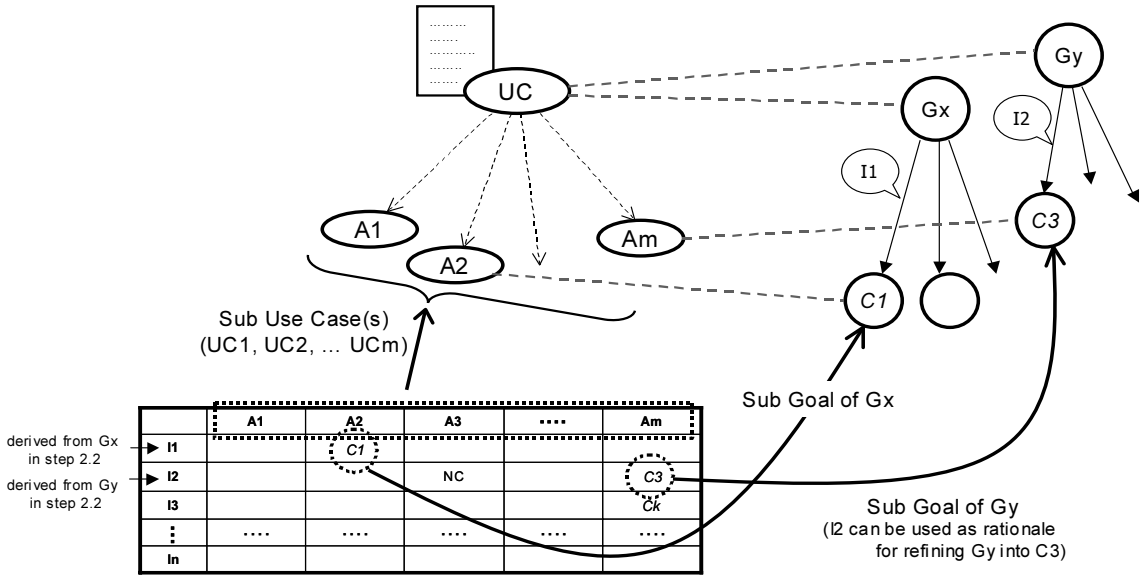

Figure 5: Deriving Sub Goals and Sub Use Case from a Idea-Action Mapping Table

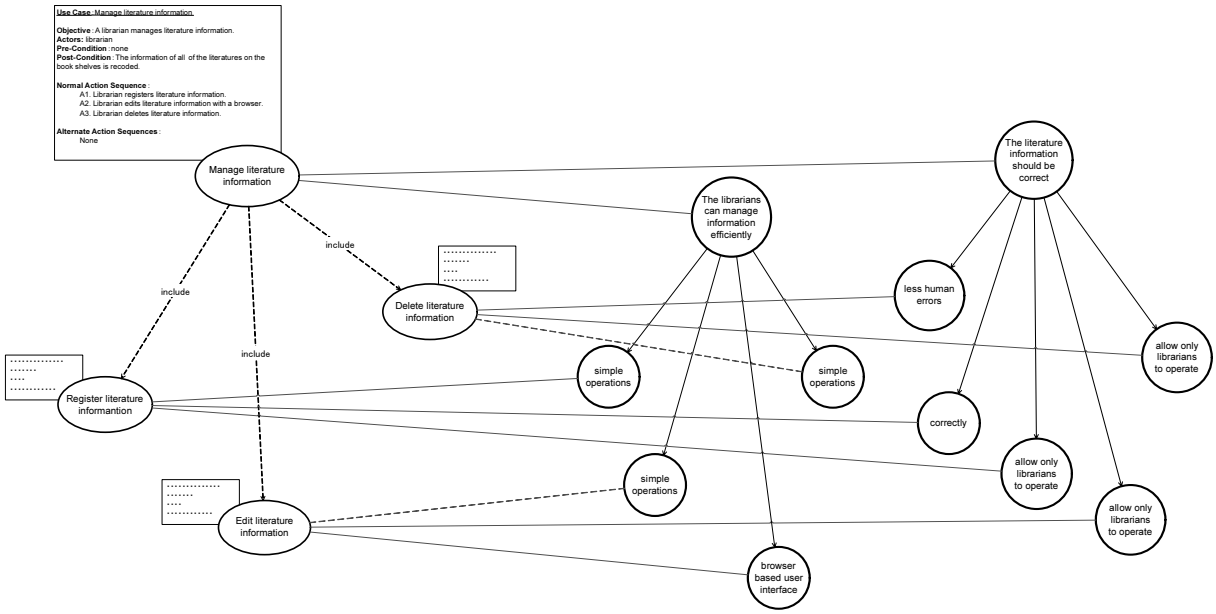

Figure 6: an Example of the Graphs after Deriving Sub Goals and Sub Use Cases

Table 2: Experimental Results

	Project 1	Project 2
time(hours)	approx. 5	approx. 4
number of initial goals	2	2
number of initial use cases	1	1
total number of goals	28	21
total number of use cases	34	22
number of specified use cases	5	5
number of system goals	10	15
number of system use cases	4	4

system use cases are linked are relevant goals to the system and he picks up them as system goals.

4 Evaluation

To assess the feasibility and effectiveness of our method, we applied it to a couple of development projects. One of the projects was the development of a supporting system for a laboratory administrator that manages miscellaneous tasks in a university laboratory, such as human resource allocation, progress management of laboratory projects and so on. The other one was a development of a supporting system for a library in a university laboratory. For each development project, we had an analyst who had much experienced in applying both of goal-oriented analysis and use case analysis. Before starting our experiments, we provided the analysts with 30 minutes' lecture to introduce our method. The analysts used Microsoft Visio, to which we added the stencils for goal graphs and use case graphs, in order to draw the graphs. Table 2 shows the summary of the results.

After applying our method, we interviewed and discussed with the analysts to compare it with each of goal-oriented analysis and use case analysis, and assess its effectiveness. The summaries of its benefits identified during this discussion is as follows;

- *Our method can elicit various types and granularities of requirements.*

  In the example of the support system for a laboratory administrator, the analyst elicited "the tasks should be completed by their deadlines". This requirement can be categorized into a constraint, a goal in our words, and it cannot be elicited with use case analysis alone. Furthermore it derived sub use cases "allocating tasks to the staffs", "reporting progress status of the allocated tasks" and "monitoring and checking progress status". That is to say, the goal contributed to the refinement of use cases. In particular, the analyst could never derived the latter two sub use cases in this example if he had not focused on the goal "the tasks should be completed by their deadlines". Many similar cases appeared in our experiments. Our method is very useful to elicit various types and granularities requirements without omission from the two viewpoints of constraints and actions.

- *Our method is helpful to get the requirements that are correct to customers' needs and that are traceable.*

  The analysts frequently traced back to the roots of the goal graphs and the use case graphs along their edges, in order to check if the initial needs of the customers were correctly decomposed and refined. Our graphs express the histories of deriving the concrete requirements from the customers' abstract ones, and it is helpful to validate the correctness and to trace the processes of requirements analysis backward. Note that the ideas generated in the step 2-2) played a role of rationales of the refinement activities during checking the correctness.

- *The analysts can concentrate themselves on the analysis from only one aspect of requirements; constraint or action, at a time, because these two aspects are separated explicitly in our elicitation process.*

  In our example, the use case "allocating the tasks to the staffs" and the constraint "the human allocation is done without fail" are individually refined. To refine the use case, the analyst could bring his concern to the sub use cases to perform "allocating the tasks to the staffs", without considering "the human allocation is done without fail". As a result, he could refine it easily and got the sub use cases of high quality. Suppose that the he should refine their mixture version "allocating the tasks to the staffs without fail" all at once. His concern would have been diverged to the refinement "allocating the task" and to the "without fail". The separation of concerns is significant to requirements elicitation and our method can support it partially.

- *Explicit relationships between goals and use cases (context links) make both goal analysis and use case analysis more effective.*

  In our experimental cases, about 20% of the goals and the use cases were derived due to the effects. Major effects are as follows;

  - The context of a goal is clarified with behavioral information from a use case related to it.
  - Inconsistent decomposition of goals that have the same context is suppressed because all goals associated with a use case (as their context) are analyzed simultaneously.
  - When an analyst describes action sequences of a use case, he can be certain to take related constraints into account.

Table 3: Categories of the Guidelines

	# of guidelines
identifying inital goals and use cases	8
identifying basic actions	4
generating idea to achieve goals	4
specifying a use case	5

- *Our method provides analysts with the well-defined concrete process for effective requirements elicitation.*

During our case studies, we found some helpful knowledge and heuristics for analysts to apply our method effectively. We organized them as method guidelines categorized into four groups as shown in Table 3. For instance, Guideline "When specifying action sequences of a use case, identified ideas may include action attributes, i.e. subject, time, place and so on, actions not identified as basic action, and conditions for execution of alternate action sequences" is provided for the step 2.3). This guideline helps us to specify the action sequences of the use case because it says that the generated ideas include significant information on the action sequences. We have five guidelines in total for this step as shown in Table 3. Note that, for steps not shown in Table 3, i.e. creating an Idea-Action Mapping Table, deriving sub goals and sub use cases, and selecting system requirements, guidelines may not be necessary because these steps are almost automatically executable.

5 Conclusion and Future Work

This paper proposed a new method which goal-oriented analysis and use case one are integrated into, and discussed its assessment by a couple of experiments. And we summarized useful findings in our experiments as guidelines, and they are useful to apply our method to practical development projects although our experiments were in laboratory level. The experimental results suggested that our method can improve the shortcomings that goal-oriented analysis and use case one have, e.g. the difficulties of decomposition and refinement into sub goals and sub use cases.

The essential point of our method is that goal-oriented analysis contributes to the refinement of use cases very well and vice versa. In this sense, our method integration approach aims at complementary integration where both of the methods work well complementary. This style of method integration is very sufficient for us to get powerful methods and method engineering techniques can support it [4]. Meta modeling of our method is one of the future work.

The future work related to our method itself is listed up as follows;

- Certain patterns of generating ideas were found in our experiments. It means that there is a possibility to have idea-generation patterns and catalogue them so that analysts can reuse them.
- The process of our method could be applied to business process structuring.
- A supporting tool that automates analyst's activities and manages intermediate products is required.
- More case studies, in particular large-scale developments projects in practical level, are necessary.

References

[1] C. B. Achour, C. Rolland, N. A. M. Maiden, and C. Souveyet. Guiding use case authoring: Results of an empirical study. In *Proceedings of the 4th IEEE International Symposium on Requirements Engineering (RE 99)*, pages 36–43, 1999.

[2] A. I. Anton, R. A. Carter, A. Dagnino, J. H. Dempster, and D. F. Siege. Deriving goals from a use-case based requirements specification. *Requirements Engineering Journal*, 6:63–73, 2001.

[3] A. I. Anton and C. Potts. The use of goals to surface requirements for evolving systems. In *Proceedings of the 20th International Conference on Software Engineering (ICSE 98)*, pages 157–166, April 1998.

[4] S. Brinkkemper, M. Saeki, and F. Harmsen. Meta-modelling based assembly techniques for stuational method engineering. *Information systems*, 24(3):209–228, May 1999.

[5] L. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos. *Non-Functional Requirements in Software Engineering*. Kluwer Academic, 1999.

[6] A. Dardenne, A. van Lamsweerde, and S. Fickas. Goal-directed requirements acquisition. *Science of Computer Programming*, 20(1-2):3–50, 1993.

[7] C. Rolland, C. Souveyet, and C. B. Achour. Guiding goal modelling using scenarios. *IEEE Transaction on Software Engineering*, 24(12):1055–1071, December 1998.

[8] V. F. Santander and J. F. Castro. Deriving use cases from organizational modeling. In *Proceedings of the 10th Anniversary IEEE Joint International Conference on Requirements Engineering(RE'02)*, pages 32–39, 2002.

[9] K. Weidenhaupt, K. Pohl, M. Jarke, and P. Haumer. Scenarios in system development: Current practice. *IEEE Software*, 15(2):34–45, Mach/April 1998.

[10] E. Yu. Towards modeling and reasoning support for early-phase requirements engineering. In *Proceedings of the Third IEEE International Symposium on Requirments Engineering(RE'97)*, pages 226–235, 1997.