Research on Support System for Addition of Use Case Using Theorem Proving Technique

Ryohei Ushio (510014)

School of Information Science,
Japan Advanced Institute of Science and Technology

February 8, 2007

Keywords: software engineering, UML, use case, theorem proving, first-order predicate logic, Prolog.

1 Introduction

Recently, In object-oriented software development, we have mostly used UML, as a language for the system analysis and the system design. Use case diagram and use case description which are element of UML are used for the requirements analysis. Use cases express the interaction between actor and system as an event series, and we cover the details of the interaction as use case description in each use case in the use case diagram. Actor expresses a human being and an outside system which interacted with a system as role. When we add a new use case to an existing system, some existing use cases can often be reused. Cellphone application is listed as an example. We assume that we add a new use case "transmitting mail with photograph" to the existing application. If use case "transmitting mail" have already existed in the existing application, we can use it in the new use case "transmitting mail with photograph" as a portion of scenario. This relationship that some use case is utilized as a portion of other use case is called "include" in use case modeling. But in actual systems, the number of the use cases may be too large and a use case description tends

to be complicated and ambiguous, so it is difficult to understand which of the existing use cases is related to a new use case.

Purpose of this research is to develop a mechanism which understand which of the existing use cases is related to a new use case. In this research, I develop a system which supports the addition of a new use case by searching use cases which have possibility of reuse with "include" relationship. Various addition of a new use case can be thought, but in this research, I target one that existing use cases can be recycled partially in a new use case without changing.

2 Research Approach

In UML, precondition and postcondition are described in the use case description. Precondition means contract about that system must be satistied before a use case starts, and postcondition means contract about that system must be satistied before a use case ends, In use case description with natural language, precondition and postcondition are mostly described with only one sentence. But restrictions of the hardware, etc, are really included in those contracts as tacit one. In this research, contract included with tacit restrictions are expressed by the first-order predicate logic formerlly. In this paper, in use case relationship "include", I call use case which includes "sub use case" "base use case". And I call use case which is included in "base usecase" "sub use case". In this research, I defined the logical relationship in precondition and postcondition expressed by the first-order predicate logic. And I search use cases which have possibility of reuse by inspecting the logical relationship.

3 Convert to First-Order Predicate Logic From Use Case

I propose method to convert precondition and postcondition to the first-order predicate logic. A predicate is consist of predicate name and parameters. In this research, parameters are decided from class diagram and use case diagram. And predicate name is represented as state and role of the

parameters. And a contract is made from combination of predicates. Each contract is tied with and operation. So precondition and postcondition are made from it.

4 Search "Sub Use Case" Algorithm

I propose algorithm that search "sub use case" which can be included in a new usecase by inspecting logical relationship in precondition and postcondition expressed by the first-order predicate logic. The input of this algorithm is a new use case's precondition and postcondition expressed by the first-order predicate logic. The output of this algorithm is combination of "sub use cases" which have possibility of reuse according to order. And all candidates are output as well.

5 Development of tool

I have developed tool with prolog to implement Search "sub use case" algorithm. The reason for having developed tool with prolog is that prolog is appropriate programming language to handle first-order predicate logic and recursive processing. It is possible to comb all use cases at the time of each processing by this tool. When we try searching "sub use cases" which have possibility of reuse with this tool, We previously need to describe a use case information (use case name, precondition, postcondition) as a rule in source program.

6 Apply to Case Study

I tried applying Search "sub use case" algorithm to the case study of the elevator control system of Gomma.

Firstly I described precondition and postcondition with first-order predicate logic in all use cases of the elevator control system. Next I added new use cases to the elevator control system, and described precondition and postcondition with first-order predicate logic in new use cases. Finally I tried searching "sub use cases" which have possibility of reuse with tool which I developed.

In this applying, I designed a new use case as it include existing "sub use cases". In a word, because the "sub use cases" that can be included in a new use case was understood in advance, it was able to judge whether output result was appropriate or inappropriate. From a result of applying, appropriate combination of "sub use case" were sarched. But I have found out that inappropriate one were really sarched. In this applying, an appropriate output was able to be expected beforehand. However, when a new use case is added to a existing system, an existing use case that relates to it cannot be generally expected. So the user need to examine whether the output is appropriate or inappropriate from output result.

7 Future Work

The method to add a new use case was developed in this research so that the user might apply the theorem proof system with recycling existing use cases. The tool I developed in this research can search "sub use cases" which have possibility of reuse, if we can realize a new use case by combining existing use cases. But in case of using existing use case with modifying the portion of it, The tool cannot correspond the case. So I want to realize the system which can inspect how influence does modification of precondition and postcondition to other use cases, when we modify a existing use case.