Study on Use Case Granularity

Zhang Na
School of Computer
Beihang University
Beijing, China
sdzhna@163.com

Luo Yanjing
School of Computer
Beihang University
Beijing, China

***Abstract*—Use case seems simple, but it is hard to study its details. How to get appropriate use cases, how to control their granularities, and how small their granularities are to clearly describe system, need requirment analysts' consideration. In this paper, it presents some guidelines by deeply analyzing the context and causes of use case granularity.**

Keywords-use case; scope; actor; use case granularity; abstract level

I. INTRODUCTION

The emergence of use case analysis method provides an effective way of capturing, analyzing and describing requirments. Compared with traditional method, use case is easily written and understood, and it describes system from user's perspective. But the hardest is how to control use case granularity. Although it can be seen use case as long as in the form it can be written paths and steps of meeting requirment[1], that how large of use case is appropriate needs to consider other factors.

II. USE CASE GRANULARITY

Scope identifes boundary of analysis, and comes from special system goal, such as business goal, system feature. Actor is a tight set of roles acted by users who interact with use cases. All desires of actors present system goal. Use case is description of an interaction between user and system for certain goal, and it realizes system goal.

When analyze use cases, requirment analysts need to assume a boundary by system goal, and capture requirments in it. The found set of requirments determines final scope. Constantly adjusting and gradually defining requirments causes a more accurate scope. Once scope is determined, actor can be determined easily. Use case comes from actor's desire for system, so use case can be determined finally. Fig.1 gives the relationships between them.

Use case granularity is used to describe size of user's goal[2]. According to the relationships between use case, actor, scope and system goal, it is easy to find that accurately grasping the scope is the key to control the granularity of use cases. Besides, scope determines abstract level of current phase. If scope is unclear, abstract level is unclear and granularity of use cases is inconsistent.

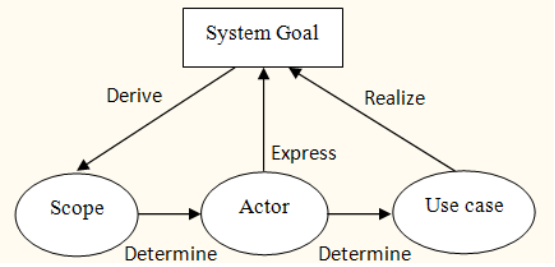

Figure 1. Relationships between use case, actor, scope and system goal

For example, whether "Retrieve Books" is a use case is determined by the defined scope. If the scope is "Library" and library only provides the service of searching books, borrower's desire for library is not satisfied. That is, values provided by library to borrower are not only searching books, but also borrowing books. If the scope is "Retrieval System of Books", borrower's desire for it is retrieving books, not borrowing books. If the scope is changed to "Library Management System", value provided by it to borrower is asking librarian to deal with procedures of lending books. Fig.2~Fig.4 give correct drawings, and Fig.5~Fig.6 give wrong drawings.

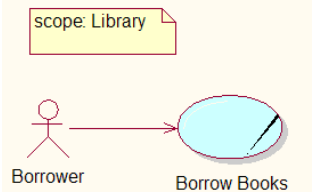

Figure 2. Correct drawing that scope is library

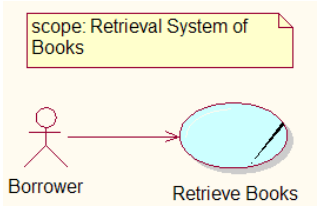

Figure 3. Correct drawing that scope is retrieval system

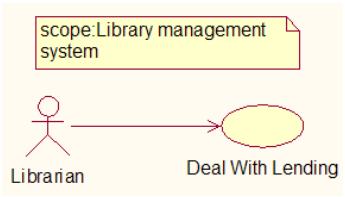

Figure 4. Correct drawing that scope is library management system

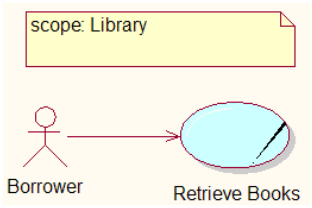

Figure 5. Wrong drawing that scope is library

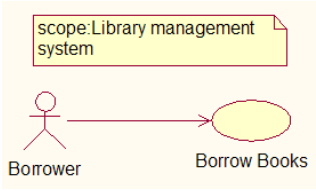

Figure 6. Wrong drawing that scope is library management system

III. CAUSES

Mistakenly taking step as use case is one of the causes. It is a common mistake.The reason is failing to distinguish goal and step. As shown in Fig.7, analysts treat steps of inputing password, checking account and deducting money as use cases. This kind of mistake may cause inaccurate requirments and too small granularity. Fig.8 gives correct drawing.

Use case is description of a coherent functional unit of system or subsystem without details. One use case is one event, or desire of actor for target system. In order to complete the event, it needs many steps, but the steps fail to completely reflect actor's goal, so they cannot be treated as use cases. Therefore, it will lose the significance of decomposing use cases, if analysts simply treat a smaller part of failing to provide valuable information as use case.

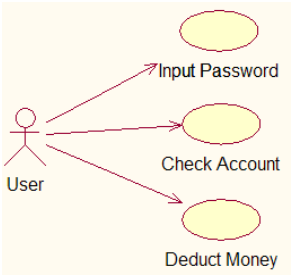

Figure 7. Mistakenly taking step as use case

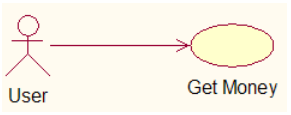

Figure 8. Correct drawing

That thinking system functions are the use cases is another cause. Function decomposing is the enemy of use case analysis method, and many problems about granularity are from function decomposing[1]. System function is viewed from developer's or system's perspective, and divorces user's desire. Use case is viewed from user's perspective and used to describe user's desire and request of system.

IV. GUIDELINES

Essentials of granularity is determination of scope. If requirment analysts fail to recognise scope, they will not know which abstract level current phase is, which causes different granularities in same requirment phase. No matter how to select granularity, one guideline must be remembered, that is in same requirment phase granularities of all use cases shoule be in same magnitude. Besides, there are other guidelines.

A. View Actor as the Center

When analysts capture requirments, they must take actor as the center, and stress actor's goal. Granularity of use case, especially business use case, should be selected according to whether it complete one actor's goal. Different actors should be analyzed separately, which is good for controlling granularity.

B. Determine Function Points

Function point is relatively independent. It can complete a goal individually. It may contain an operation, or a series of continuous operations. Determination of function points is helpful to find use cases. For application software, use cases can be gotten by capturing business. For player software, it has more independent functional points, so analysts can treat a function point as a use case.

C. Try to Decompose Use Cases and Constantly Revise them

Sometimes, analysts fail to start decomposing process because of unclear granularity. So first analysts should try to decompose them and draw a draft, and then take object-oriented method to examine and modify them by adding, deleting or combining some user goals.

D. Gradually Refine Use Cases and Present them Hierarchically

It is better for analysts to decompose use cases from coarse to fine, and constantly revise them. It may cause wrong granularity, if analysts focus on details at first. Right way is that analysts first decompose system into coarse use cases in large granularity, and then describe them by smaller granularity. The hierarchical description avoids the situation that large numbers of use cases are in one drawing. Different granularities presents different abstract levels.

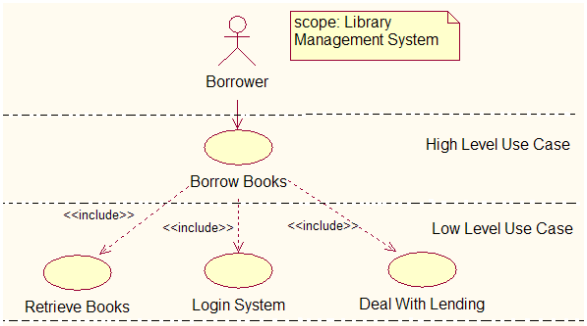

Figure 9. Wrong high level use case

During the process of refining use cases hierarchically, analysts must take care of the substitution of the scope. As shown in Fig.9, the scope is "Library Management System", but analysts change the scope to "Library" when they analyze high level use cases.

E. Different Granularities in Different Requirment Phases

Granularity of use case determines complexity of different use case models. Analysts should control the complexity by specific circumstances of every system.

There are three models in requirment phase, business model, concept model and system model. Three models are the process of gradual refinement. In business model, granularity of use case is appropriate to describe a complete event or business process. In concept model, granularity is appropriate to describe a complete event flow or one step of a business process. In system model, granularity is appropriate to describe an interaction between operator and computer.

F. High Cohesion and Low Coupling

Use cases in the same abstract level should maintain in high cohesion and low coupling condition. It is better to reduce relationships between use cases of different and same actors.

G. CRUD Use Case

Is CRUD(Create, Retrieve, Update and Delete) one use case or four, is different in different requirment phases.

In business model, CRUD is one use case, because it is one business. For example, in user management business process, user must be added first, then it can be modifed or deleted. The series of operations construct actor's goal of management. So user management is one use case, as shown in Fig.10.

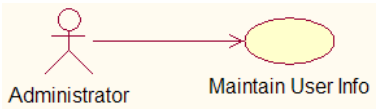

Figure 10. Business use case

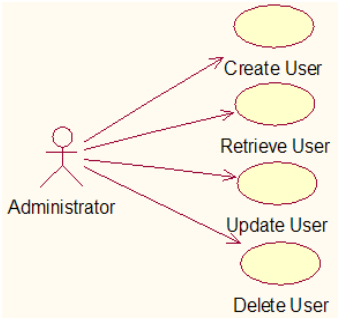

Figure 11. System use case

In system model, purpose of use case is simulating actor's business in computer. In general, CRUD, four actions, can not happen in the same time in computer. So CRUD is divided into four use cases, as shown in Fig.11.

ACKNOWLEDGMENT

In use case analysis, there are a lot of difficulties to study details of use case and list steps of every scenario. Failing to control use case granularity is one of the difficulties. Inappropriate set of use cases may bring a disaster for identifying objects of system and get an irrational design of object structure[3]. In this paper, it analyzes relationships between use case, scope, actor and system goal and finds reasons of granularity problem. In the end, it proposes some guidelines for requirment analysts. Further study will focus on examination on effectiveness of the guidelines.

REFERENCES

[1] Pan Jiayu, Does use case have granularity, Programer[J], 2008.3, PP.72-74.

[2] Wang Xianguo, UML Unified Modeling Practical Guide[M], Beijing: Tsinghua University Press, 2009.4, PP.77.

[3] He Keqing, He Fei, Ying Shi, Role Use Case: A More Complete Analysis Method for UML, Journal of Computer Research and Development[J], 2001.9, PP.1105-1111.