

A Multi-Role Collaborative Method and Platform for Developing Software Requirements*

Chin-Yi Tsai and Chua-Huang Huang
Department of Information Engineering and Computer Science
Feng Chia University
Taichung, Taiwan
{cyt, chh}@pmlab.iecs.fcu.edu.tw

Abstract - *In general, software development consists of requirements analysis, system analysis, system design, implementation, and testing. Among these phases, requirements analysis plays an important role during the software development. How to capture customer's desired requirements precisely is a great challenge. Many efforts have been made in the area of requirements engineering. The most used approach is using goal-oriented approach to capture, analyze, and elaborate requirements. However, few efforts have been invested to investigate software requirements development by a group of team members. In this paper, we present a collaborative method and process to develop software requirements on a network-based collaborative platform. There are multi-roles and each of the roles performs a number of tasks of requirements writing, reviewing, and commenting at the same time on the network-based environment. Different roles are in charge of different tasks. Based on the collaborative method, a number of role relationships are proposed. In order to achieve the purpose of collaborative working, the platform provides several collaborative services to satisfy the need of developing requirements of a software system. All users carry out their work through different collaborative patterns. After all stakeholders finish their authoring of requirements, the integrated software requirements document will be generated.*

Keywords: software engineering, requirements engineering, goal-oriented, collaborative, software requirements.

1 Introduction

In general, software development consists of requirements analysis, system analysis, system design, implementation, and testing. There are many software development methods have been proposed in recent decades, including the

waterfall approach [19], Spiral [4], the iterative and incremental approach [16], RUP [14], XP [3], *etc.* According to different needs and characteristics, a software team may choose a proper model as their software development model. However, no matter which model is used, requirements phase plays an important role during the software development. Most failures of software development are due to the implicitness of requirements such that the later the inadequate requirements are discovered, the higher software development risks one may face.

Requirements engineering is concerned with requirements elicitation, analysis, and elaboration. There have been several studies pertaining to the field of requirements engineering proposed in recent years. A method, called KAOS, has been proposed to develop requirements engineering [10, 20, 21]. One can use KAOS to develop software requirements through goal refinement and goal operationalization. Bolchini *et al.* derives web application requirements through goal-oriented analysis [5, 6]. Yu proposes the i^* framework as an agent-oriented approach to requirements engineering [22]. For the i^* framework [22], the strategic dependency (SD) and strategic rationale (SR), are the two main modeling components. The SD is used to represent the dependency relationship among various actors. As for the SR, it is used to describe stakeholder interests and concern, and how they might be addressed by various configurations of systems and environment. Tropos uses the i^* modeling framework to develop requirements-driven development for agent software [7]. This methodology consists of early requirements, late requirements, architectural, and detailed design phases. Similarly, Formal Tropos adopts the notation of the i^* framework as its modeling framework [12]. In addition, Formal Tropos uses temporal logic [17] which used in KAOS project as its formal specification language. Also, Formal Tropos performs formal verification by using model checker NuSMV [9].

There are two kinds of software requirements, including functional requirements and non-functional requirements. Most researches concentrate on the study of func-

*This work was supported in part by National Science Council, Taiwan, under grant NSC-94-2213-E-035-039.

tional requirements. However, some others select to ignore non-functional requirements because of their difficulty. Chung and Mylopoulos propose NFR framework to deal with non-functional requirements [8, 18]. In NFR framework [8, 18], decomposing NFR softgoals into more specific requirements. The results of goal refinement are expressed in graphs, called softgoal interdependency graphs (SIGs). That is, SIGs are used to represent and record the softgoals decomposition and express the processing of design. Recently, object-oriented software development becomes more and more popular. RUP have been proved is a successful software methodology [14]. Most modern software methodologies use Unified Modeling Language (UML) to specify, document, and visualize software specification [11]. In order to understand the requirements deeply, Ivar Jacobson proposes an use case driven Approach for requirements analysis [13]. Besides, UCM is a scenario-based approach for representing requirements [2].

Considering the development of software requirements, efficient communication among stakeholders and system developers is very important. There are a few works focus on this issue in the last few years. In order to manage software requirements efficiently, [15] develops a tool to support collaborative software requirements management. In this paper, we propose a multi-role collaborative method and platform for developing software requirements. The primary goal of this paper is to enhance the communication among people involved in the software project, including different stakeholders, analysts, managers, and so on. Through the enhancement of communication, analysts can understand the requirements of customers deeply. In addition, customers can represent more proper needs by means of the help of analysts. Due to the characteristics of involved people distributed in different places, a network-based environment is needed. By using such network-based platform, people can complete their own work anytime and anywhere. Hence, it may improve the efficiency as well as save time.

In this paper, we define several kinds of roles and roles relationship. By defining roles relationship, we can design collaborative mechanism which serves people satisfying their work on collaborative platform further. Collaborative platform provides several collaborative services for developing software requirements. We also derive several useful collaborative patterns. Actually, most collaborative works and tasks performed on collaborative platform follow derived collaborative patterns. After all stakeholders finish their authoring of requirements, the integrated software requirements document will be generated.

The organization of this paper is as the following. In Section 2, we define some kinds of roles and role relationship. In Section 3, we will describe the collaborative mechanism for developing software requirements, includ-

ing collaborative services and collaborative patterns. The methods of collaborative software requirements generation are given in Section 4. The process of using collaborative platform for developing software requirements are given in Section 5. Conclusions and future works are given in Section 6.

2 Roles and Role Relationship

Software system development is a cooperative work, especially software requirements development phase. There are some people involved during the development of software requirement. Hence, we define several roles and role relationships.

2.1 Roles

On the multi-role collaborative platform, we can simply divide roles into two kinds of roles. One kind belongs to customers side, the other is system developers side. We totally define six roles in the multi-role collaborative platform. We define client representative, stakeholder, and client manager for customers side. As for system developers side, we define development representative, analyst, and development manager. Different roles have different characteristics and tasks. The description of various role as below:

Client representative. Negotiate with development representative to determine who will involve in the development of software requirement.

Stakeholder. The primary users to use the developed system. Various stakeholders write their needs through the collaborative platform. Upon any comments are made by others, stakeholders can modify their requirements based on the comments.

Client manager. Communicate with stakeholders and development manager. In addition, client manager is responsible for the management and monitoring of requirements from all stakeholders.

Development representative. Negotiate with client representative to determine who will involve in the development of software requirement.

Analyst. Analysts are in charge of system analysis. Besides, analysts also assist stakeholders in authoring their requirements through the collaborative mechanism.

Development manager. Communicate with analyst and client manager. In addition, development manager is responsible for the generation of collaborative software requirements.

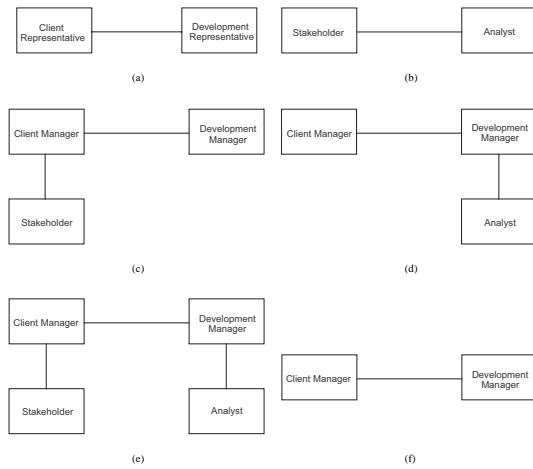


Figure 1: Role Relationship (a) Representative-Representative Negotiation Role Relationship (b) Stakeholder-Analyst Collaborative Role Relationship (c) (Client Manager/Stakeholder)-Development Manager Role Relationship (d) Client Manager-(Development Manager/Analyst) Role Relationship (e) (Client Manager/Stakeholder)-(Development Manager/Analyst) Role Relationship (f) Client Manager-Development Manager Role Relationship

2.2 Role Relationships

In this paper, we define six different roles collaboratively developing software requirements on network-based collaborative platform. In the following, we derive several role relationships among roles. Actually, most collaborative methods follow the role relationship. The detailed descriptions of role relationships as below.

Representative-representative negotiation role relationship. In general, we can divide collaborative software requirements development process into four main phases: role assignment, stakeholders authoring, analyst commenting, and collaborative software requirements generation. This role relationship as showed in Figure 1(a) happened in role assignment phase. In this phase, there are at least one client representative and development representative discuss and negotiate together in order to determine various relationships and assign people to play proper role.

Stakeholder-analyst collaborative role relationship. There are a variety of different stakeholders in the customers side. Different stakeholders use collaborative platform to write their requirements with the help of analysts through the collaborative platform. After stakeholders write their own requirements, analysts can help stakeholders to finish complete requirements by making comments on the inadequate requirements. Then, stakeholders can modify and improve their requirements according to the

comments from analysts. The role relationship showed in Figure 1(b).

(Client manager/stakeholder)-development manager role relationship. This role relationship as showed in Figure 1(c) is composed of three kinds of roles. The main communication is between one client manager and one development manager. In addition to development manager, client manager also communicate with stakeholders. This time, client manager is regarded as the bridge between stakeholders and development manager.

Client manager-(development manager/analyst) role relationship. This role relationship as showed in Figure 1(d) is similar to *(client manager/stakeholder)-development manager role relationship*. The main difference is that development manager is the bridge between analyst and client manager in this role relationship.

(Client manager/stakeholder)-(development manager/analyst) role relationship. This role relationship as showed in Figure 1(e) involved four kinds of roles. This relationship concentrates on the communication between one client manager and one development manager. In addition, client manager may need the support and help from stakeholders. Similarly, analysts can help development manager to make the communication more efficient.

Client manager-development manager role relationship. This role relationship is showed in Figure 1(f). After various stakeholders finish their own requirements respectively, the collaborative software requirements will be generated. Then, we have to perform the validation of collaborative software requirements. Client manager and development manager perform the validation together. For example, one of the validation method is simulation.

3 Collaborative Mechanism

Collaborative platform provides several collaborative services for developing software requirements. The provided collaborative services form some common collaborative patterns. These collaborative patterns are the bases of collaborative platform. Users interact with each other follow collaborative patterns on collaborative platform. These collaborative patterns support the primary collaborative mechanism.

3.1 Collaborative Services

In this paper, we propose four collaborative services to support the development of collaborative software requirements on collaborative platform. In the following, we will give the description of various collaborative services.

Collaborative authoring service: Provide users for writing, viewing, and commenting the requirements.

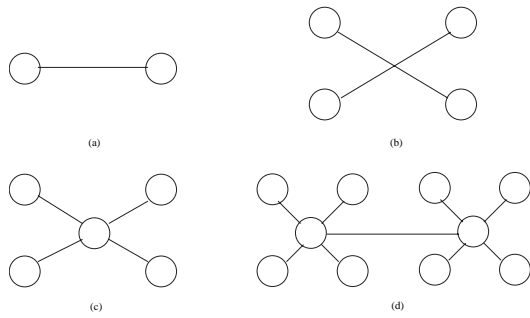


Figure 2: Collaborative Diagram (a) Single peer interaction collaborative diagram (b) Multi peer interaction collaborative diagram (c) Single central collaboration collaborative diagram (d) Multi central collaboration collaborative diagram

Supporting work service: Provide some supports for helping users carry out their work.

Integrated specification generation service: After writing the requirements, this service provides a mechanism for generating software requirements specification.

Collaborative validation service: Provide the environment and method of requirements validation. Development manager and client manager interact with each other in order to validate the requirements specification collaboratively.

3.2 Collaborative Patterns

Considering the development of software requirements, we intend to improve the communication among people involved in the software development project. Hence, we especially highlight *collaboration* among people on network-based environment. In the following, we conclude several common collaborative patterns happened on collaborative platform.

Before giving the description of collaborative patterns, we explain the meaning of titles which are used to describe the collaborative patterns firstly. *Name* title is unique identification for patterns. *Short description* title gives a very short description for patterns. *Roles* title contains all roles within patterns. *Collaborative diagram* title will give a diagram to show the interaction between roles. *Feature* title describe the primary characteristic about the pattern. Finally, *detailed description* title describes all possible interactions of the pattern in detail.

1. Single peer interaction.

Name: Single peer interaction
Short description: One person interact with another person.
Roles: Stakeholders, analyst, client manager, devel-

opment manager.

Collaborative diagram: Figure 2(a)

Feature: The interaction happened between only two people.

Detailed description: Single peer interaction is the most basic collaborative pattern. As a basic interaction pattern, this pattern may work independently, in addition, this pattern also can incorporate with other works to form another collaborative patterns. This collaborative pattern contain two kinds of roles. There are several kinds of composition of roles in this collaborative pattern as below.

- (a) Stakeholder-2-stakeholder
- (b) Stakeholder-2-analyst
- (c) Client manager-2-development manager
- (d) Development manager-2-analyst

2. Multi peer interaction.

Name: Multi peer interaction

Short description: Same kind of role interacts with each other.

Roles: Stakeholder

Collaborative diagram: Figure 2(b)

Feature: All people are the same kind of role.

Detailed description: Actually, this collaborative pattern is concerned with stakeholders mainly. Stakeholders write their own requirements, other stakeholders can review them.

3. Single central collaboration.

Name: Single Central Collaboration

Short description: There is a manager work as administrator. This manager is the role of bridge among other people.

Roles: Stakeholder, client manager, analyst, development manager.

Collaborative diagram: Figure 2(c)

Feature: There is a person works as administrator.

Detailed description: There are tow kinds of collaborative pattern according to the role of manager. Client manager plays the role of administrator, other roles are stakeholders. Development manager plays the role of administrator, other roles are analysts.

4. Multi central collaboration.

Name: Multi Central Collaboration

Short description: Contains two kinds of managers in this collaborative pattern.

Roles: Stakeholder, client manager, analyst, development manager.

Collaborative diagram: Figure 2(d)

Feature: This pattern happened in the late phase.

Detailed description: This pattern concentrates on that client manager communicates with development manager. Besides, client manager might interacts

with stakeholders and development manager interacts with analysts.

4 Collaborative Software Requirements Generation

The ultimate artifact of software requirements development is software requirements specification. IEEE std-830 [1] recommends chapter organization of specification as well as what should be contained in software requirements specification. Most requirements analysts adopt SRS template as chapter structure of software requirement specification. In this paper, we use pruned SRS template as our software requirements specification template. It contains several sections, including introduction, functional requirements, non-functional requirements, and glossary. All stakeholders write their own requirements, respectively. Upon all stakeholders finish the writing of requirements, collaborative software requirements will be generated according to the format of pruned SRS template.

Stakeholders use requirements writing service to write their own requirements on collaborative platform. Generally, requirements are divided into two kinds of requirements, including functional requirements (FR) and non-functional requirements (NFR). There are three scenarios for determining the kind of requirement. Firstly, stakeholders determine the kind of requirements after writing the requirements. Since this requirement is very obvious that requirement belongs to the kind of functional requirement or non-functional requirement, stakeholders can make the determination without the help of analysts. Second, the requirements are not very obvious to be judged which kinds of requirements it belonged to. Even so, stakeholders still determine the type of requirements. Then, analysts give some recommendation about the kinds of requirements written by stakeholders. Afterward, stakeholders can modify the kind of requirement based on the recommendation according to the recommendation. Finally, it is hard for stakeholders to determine the kinds of requirements. Stakeholders only write down the requirement without determining the kind of requirement. The task of determining the kind of requirement is performed by analysts.

Considering the non-functional requirements, there are two kinds of requirements. They are global non-functional requirements that the constraints are suitable for entire system. As for local non-functional requirement, the constraints only have effects on some parts of system. The arrangement of non-functional requirements is based on the properties of global or local.

For the monitoring of progress of software requirements development, we can use a simple approach for the

monitoring of development progress. The monitoring of progress of the development of software requirements development is based on how many stakeholders involved. The rate of how many stakeholders finish the writing of requirements might be regarded as the reference of progress. Hence, according to this, we are aware of the progress of the development of software requirements currently on collaborative platform. In fact, we can adjust the percent of everyone's responsibility according to some factors if necessary.

5 Collaborative Software Requirements Development Process

Requirements engineering and process are concerned with capturing customer's requirements. In this paper, we propose a collaborative software requirements development process for developing software requirements on collaborative environment. The collaborative process consists of five activities, including *role assignment*, *role assignment validation*, *stakeholders authoring*, *analyst commenting*, and *SRS generation*. In addition, activity might produce some artifacts, including *initial requirements or refined requirements*, *comments*, and *final collaborative requirements specification*.

The complete collaborative software requirements development process showed in Figure 3. The detailed descriptions of activities are given in the following.

1. **Role assignment activity.** The primary task of this activity is that preparing for proceeding to collaborative software requirements development. At this activity, client and development representative discuss and negotiate with each other. After that, they determine several roles which involved during the software requirements development.
2. **Role assignment validation activity.** This activity check if the role assignment is adequate. If there are some conflict happened or it is not proper, it is necessary to return to previous activity. If not, this activity will finish, and goes to next activity *stakeholders authoring*.
3. **Stakeholders authoring activity.** The first time enters this activity, stakeholders write their own requirements. This time produces *initial requirements* artifact as well as enters next activity *analyst commenting*. If this activity is not performed at the first time, it will require requirements and comments as input.
4. **Analyst commenting activity.** This activity requires requirements written by stakeholders as input. Analyst will give useful comments to stakeholders if

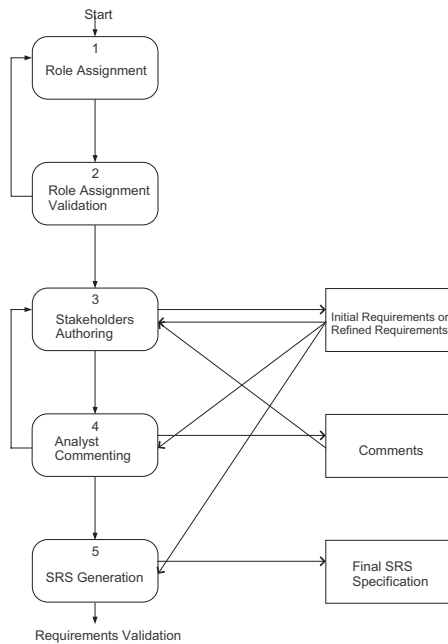


Figure 3: Collaborative Software Requirements Development Process

the requirements are still not complete enough. If requirements are still not finished, it will return to previous activity *stakeholders authoring*. Otherwise, it will enter to next activity *collaborative software requirements generation*.

5. **SRS generation activity.** This activity requires refined requirements as input. it will produce *final collaborative requirements specification* artifact. After that, it will enter to next phase doing requirements validation.

6 Conclusions and Future Works

Capturing requirements from stakeholders correctly and efficiently is always an important and difficult task. In addition to technique aspect, we should also consider domain knowledge aspect as well. There are many people with different backgrounds involved in software project development. Hence, it is very important that making the communication among people more efficient. In the past, interviewing is the most used approach to capture customer's requirements. However, sometimes it spends too much time. Due to the emergence of high-speed network, we can communicate with other people through the network. Similarly, we can elicit customer's requirements on network environment.

In this paper, we propose that using collaborative

method on network-based environment to develop software requirements. Due to the proposed platform is a collaborative platform, we define six roles in the platform, including client representative, stakeholder, client manager, development representative, analyst, and development manager. These different roles perform the task of requirements writing, reviewing, and commenting at the same time on the network-based environment. Different roles have different tasks to accomplish. Based on the collaborative method, there are some kinds of role relationships form on the collaborative platform. In order to achieve the purpose of collaborative working, platform provides several collaborative services to satisfy the need of collaborative working. All users carry out their work by following different collaborative patterns. After all stakeholders finish their authoring of requirements, the integrated software requirements document will be generated.

Communication and collaboration are important factors in software requirements development phase. The primary goal of this paper is to improve the communication among different people during requirements development. We put collaborative mechanism into software requirements development. Through the collaborative mechanism, people cooperate with other people to develop software requirements. In addition to software requirements development, it is potential that applying collaborative mechanism to other work of software development. For example, it is possible to apply collaborative mechanism to project management and monitoring as well as system analysis and design. Our ultimate goal is to develop a network-based platform for facilitating the development of software requirements, system analysis, system design, validation, project management, and so on. In the future, we will develop collaborative project management and monitoring mechanism. Similarly, there are some roles pertaining to project management and monitoring, including project manager, team leader, and team member.

References

- [1] IEEE Std 830-1998. IEEE recommended practice for software requirements specifications, 1998.
- [2] D. Amyot and A. Eberlein. An evaluation of scenario notations and construction approaches for telecommunication systems development. *Telecommunication Systems*, 24(1):61–94, 2003.
- [3] K. Beck. *Extreme Programming Explained*. Addison-Wesley, 1999.
- [4] B. W. Boehm. A spiral model of software development and enhancement. *IEEE Computer*, 21(5):61–72, 1988.

- [5] D. Bolchini and P. Paolini. Capturing web application requirements through goal-oriented analysis. In *Proceedings of the 5th Workshop on Requirements Engineering*, pages 16–28, 2002.
- [6] D. Bolchini, P. Paolini, and G. Randazzo. Adding hypermedia requirements to goal-driven analysis. In *Proceedings of the 11th IEEE International Requirements Engineering Conference*, pages 127–137, 2003.
- [7] P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, and J. Mylopoulos. Tropos: An agent-oriented software development methodology. *Journal of Autonomous Agents and Multi-Agent Systems*, 8(3):203–236, 2004.
- [8] L. Chung, B. Nixon, E. Yu, and J. Mylopoulos. *Non-Functional Requirements in Software Engineering*. Kluwer Publishing, Dordrecht, 2000.
- [9] A. Cimatti, E. M. Clarke, E. Giunchiglia, F. Giunchiglia, M. Pistore, M. Roveri, R. Sebastiani, and A. Tacchella. Nusmv 2: An opensource tool for symbolic model checking. In *Proceedings of International Conference on Computer-Aided Verification*, 2002.
- [10] R. Darimont and A. van Lamsweerde. Formal refinement patterns for goal-driven requirements elaboration. In *Proceedings of Fourth ACM SIGSOFT Symposium on the Foundations of Software Engineering*, pages 179–190, 1996.
- [11] M. Fowler. *UML Distilled*. Addison-Wesley, 2004.
- [12] A. Fuxman, M. Pistore, J. Mylopoulos, and P. Traverso. Model checking early requirements specifications in tropos. In *Proceedings of the Fifth IEEE International Symposium on Requirements Engineering*, pages 174–181, 2001.
- [13] I. Jacobson. *Object-Oriented Software Engineering: A Use Case Driven Approach*. ACM Press, Addison-Wesley, 1992.
- [14] P. Kruchten. *The Rational Unified Process: An Introduction*. Addison-Wesley, 2000.
- [15] M. Lang and J. Duggan. A tool to support collaborative software requirements management. *Requirements Engineering*, 6(3):161–172, 2001.
- [16] C. Larman and V. R. Basili. Iterative and incremental development: A brief history. *IEEE Computer Society*, 36(6):47–56, 2003.
- [17] Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification*. Springer-Verlag, 1992.
- [18] J. Mylopoulos, L. Chung, and B. Nixon. Representing and using non-functional requirements: A process-oriented approach. *IEEE Transactions on Software Engineering*, 18(6):483–497, 1992.
- [19] W. W. Royce. Managing the development of large software systems: Concepts and techniques. In *Proceedings of WESCON*, 1970.
- [20] H. I. Van, A. van Lamsweerde, P. Massonet, and C. Poneard. Goal-oriented requirements animation. In *Proceedings of 12th IEEE International Requirements Engineering Conference*, pages 203–213, 2004.
- [21] A. van Lamsweerde. Goal-oriented requirements engineering: A guided tour. In *Proceedings of the 5th IEEE International Symposium on Requirements Engineering*, pages 249–262, 2001.
- [22] E. Yu. Towards modelling and reasoning support for early-phase requirements engineering. In *Proceedings of the 3rd IEEE International Symposium on Requirements Engineering*, pages 226–235, 1997.