PAPER Special Section on Knowledge-Based Software Engineering

Project Management Patterns to Prevent Schedule Delay Caused by Requirement Elicitation

Shozo HORI[†a)], ***Nonmember***, **Takako NAKATANI**[††], **Keiichi KATAMINE**[†††], **Naoyasu UBAYASHI**[†††], ***and*** **Masaaki HASHIMOTO**[†††], ***Members***

SUMMARY We propose PM (Project Management) patterns to prevent schedule delays caused by changes in requirements on empirical studies. Changes or late elicitation of requirements during the design, coding and test processes are one of the most serious risks, which may delay project schedules. However, changes and late elicitation of requirements are usually accepted during development processes. Therefore, the PM methods for preventing schedule delays caused by changes and late elicitation of requirements during development processes are an important area of study. In this study, we examined the actual conditions of various projects which succeeded in preventing schedule delays resulting from changes and late elicitation of requirements during development processes. We were able to extract various typical PM techniques for preventing these schedule delays. The techniques, known as "PM patterns", were also applied to other projects. The patterns were arranged on a two-dimensional framework. We discuss a framework of PM patterns aimed at solving the problems caused by changes in requirements.

key words: *project management pattern, requirement elicitation, project management knowledge, project management process, framework*

1. Introduction

In most projects, the software requirements go through iterative processes of change and elicitation during the design, coding and test phases. Changes in requirements and late elicitation throughout the project can be one of the most serious causes of project schedule delays [1]. However, to achieve customer goals, it is often inevitable that we must accept requirement changes and late elicitation after the requirement analysis phase has been completed [2].

To clarify how we cope with problematic requirement changes, we examined the Project Management (PM) techniques of an actual project which was successfully completed within its scheduled budgets and timescale parameters, even though the project accepted changes in requirements and late elicitation after the requirement analysis phase had been completed [3]–[5]. As a result, we derived three types of requirement elicitation processes and eleven PM patterns useful in preventing schedule delays caused by the changes in requirements.

Manuscript received July 4, 2009.
Manuscript revised October 19, 2009.

[†]The author is with Yaskawa Information Systems Corporation, Kawasaki-shi, 215–0004, and is with Kyushu Institute of Technology, Iizuka-shi, 820–0067 Japan.
[††]The author is with University of Tsukuba, Tokyo, 105–0011 Japan.
[†††]The authors are with Kyushu Institute of Technology, Iizuka-shi, 820–0067 Japan.
a) E-mail: hori@ysknet.co.jp

DOI: 10.1587/transinf.E93.D.745

The purpose of this paper is to propose PM patterns that prevent schedule delays caused by changes in requirements. We have arranged the patterns on a two-dimensional framework. The first dimension is a set of nine knowledge areas of PM such as scope, time and cost management. The second dimension is a group of PM processes such as planning, executing and controlling processes. In the case of an actual example project, such as the short term redevelopment of systems for future modifiability, we break down the project goals into problems to be solved such as elicitation of requirements and develop the system by appointed date, and arrange them on the framework. Then, we discuss the relationship between the project goal and PM patterns on the framework.

In addition, our research on PM patterns is based on the concept of Alexander's pattern which shows that although every building is unique, each may be created by following a collection of general patterns [6]. However, such PM patterns including anti-patterns have been mainly studied in universities by using the data obtained from students' PBL (Problem Based Learning) courses and literatures [7]–[9]. In the industrial world, PM techniques have has been arranged as best practice [10], [11]. Therefore, we are studying to define industrial techniques for solving serious problems such as requirement elicitation as PM patterns and their framework. In this paper, Sect. 2 describes three types of requirement elicitation processes and requirements of our research. Sections 3 and 4 propose the framework and PM patterns respectively. Section 5 discusses our research.

2. Requirements of Our Research

We first give a brief description of the types of requirement elicitation processes. Second, we specify the requirements of our study on PM patterns for managing these types of requirement elicitation processes.

2.1 Empirical Study

Although changes in requirements and elicitation during development projects are generally accepted, we studied an actual project which was successfully completed within its schedule. This project was a restaurant ordering system. To clarify the requirement elicitation process, elicitation situations, such as changes in requirements, were examined from the data on the development history of an actual project.

Copyright © 2010 The Institute of Electronics, Information and Communication Engineers

These changes were classified for the various architectural components of the system, and the rate of requirement elicitation was analyzed [3], [4]. The reason for this classification was that the basis of this project management exercise was software architecture.

The result is shown in Fig. 1. The vertical axis indicates the requirement elicitation rate of each software component. The horizontal axis indicates the elapsed date of the project. The rate is defined as follows:

(Requirement elicitation rate) = cumReq/allReq*100

Where allReq is the total number of requirements elicited for the software component until the end of the project, and cumReq is the cumulative number of requirements elicited until the target elapse date.

In Fig. 1, TT_C (Table Terminal Communicator), DB (Data Base), OES_M (Order Entry System Monitor), etc. are the software components of the system. Our examination and analysis show the following [3], [4]:

- Since DB is able to be reused by the existing system, the elicitation of requirements should be carried out during the early stages.
- Since TT_C has a user interface for which it is easy to accept changes in requirements, these can be continually elicited from the start of development to the end.
- OES_M needed unexpected requirements which were only understood in the late stages of development after connection to external apparatus.

According to our analysis, the studied processes for eliciting requirements could be categorized into three types shown in see Fig. 2. Each type is as follows:

– E_type (Early maturation type): It completes the requirement elicitation in the early stages of the development. It appeared in the software components for which the existing system had similar or reusable components.
– L_type (Later period maturation type): It represents a continuous process of requirement elicitation throughout the development period. This type is observed in the software components, such as the user interface which needs to compete with other companies' products.
– U_type (Unforeseen maturation type): It represents the requirement elicitation process that occurs unexpectedly in the later stage of the development. It is observed in the software components that have an interface connected to external components developed by third companies.

Of course, the E_type process is desirable, since eliciting all the requirements of the system in the early stage of the development is advantageous. However, the process may become time consuming if there are few skilled engineers. Furthermore, the quality of the requirements can be doubtful. Consequently, the processes for L_ and U_types were more acceptable.

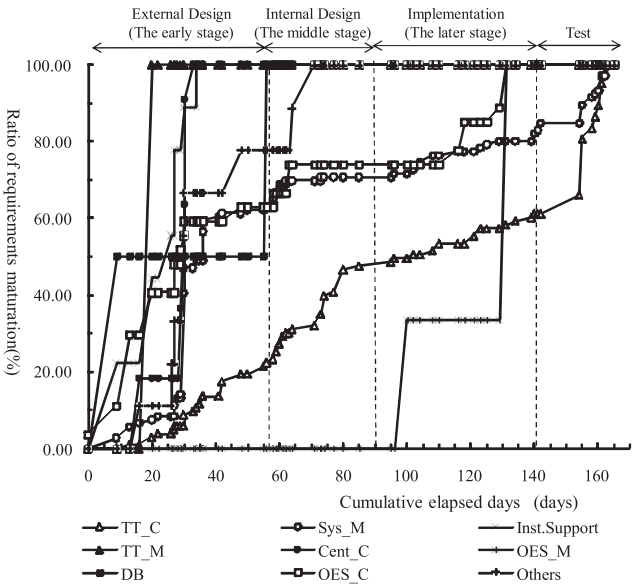

Fig. 1 Requirement elicitation ratio.

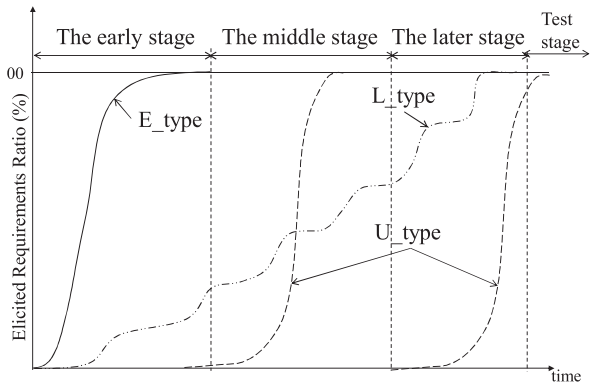

Fig. 2 Types of requirement elicitation process.

The question that arises is how to manage these types in real projects. By clarifying this, we believe that our empirical studies will be useful for an actual development project.

2.2 Requirements of Our Research

The above described E_type requirement elicitation process is the most desirable in software development. However, although L_type and U_type may pose a risk by causing project delays and disruption of the planned schedule, they are accepted. Consequently, the project is either successful or fails to meet its schedule. It is desirable to clarify the methods of PM applied to L_ and U_types. Extracting the method of PM as a PM pattern is required.

3. Framework of PM Patterns

To define PM patterns and select them for application to a project, this section describes a framework for the patterns and the decomposition of project goals into problems that the PM Patterns can solve.

3.1 Framework of PM Patterns

To construct a framework of PM patterns, we propose two dimensions for the framework structure. One dimension covers the nine areas of PM knowledge such as project scope, time and cost management. These areas are specified in the leftmost column of Table 1. The other dimension covers the five groups of PM processes such as initiating, planning and executing process groups. The five groups are specified in the right part of the top row of Table 1. We have followed the knowledge areas and PM process groups in *A Guide to the Project Management Body of Knowledge* (PMBOK) [12] which is widely accepted in the industrial world. However, this paper mainly discusses the management methods of requirement elicitation during development, the initiating and closing processes are not the points of argument.

To relate a project goal, such as prevention of schedule delay, to PM patterns, we decompose the project goal into various problems to be solved with the PM patterns. We arrange the problems such as variety of functions and external interfaces in the knowledge areas as specified in the second leftmost column in Table 1.

We also decompose a PM pattern into the solutions, which are PM activities, intended by the pattern. We arrange the solutions, such as scope planning and definition, as the cells of the matrix that are defined by the two dimensions. The matrix cells are specified in the five columns on the right-hand of Table 1. The individual identifications following the solutions in the cells indicate the individual PM patterns described in Sect. 4.2. If the solutions of the PM pattern can solve the problems in each of the nine areas, the PM pattern may be applicable to the project.

Next, we confirm that there is no contradiction in the PM patterns selected for the project. No contradiction in the PM patterns means that all the PM activities in the solutions of the PM patterns can be executed together with the other PM activities of the project. If there is no contradiction, all the PM patterns selected are applicable to the project.

3.2 Project Goal Decomposition

To select PM patterns applicable to a project, the goal of the project and its environmental constraints are broken down into the problems to be solved. The granularity of decomposed problems depends on the PM patterns. As described later in Sect. 4, each PM pattern has descriptions of a problem which can be solved by it. The granularity of decomposed problems should correspond to that of the descriptions to select the patterns. The granularity of the descriptions varies since PM patterns range from large to small. Therefore, project managers who decompose project goals should have knowledge of the PM patterns.

We describe the goal and specific environmental constraints of the actual example project, and its problems in

Table 1 Framework of project management patterns.

Knowledge Area Processes	Project Problem	Project Management Process Group				
		A) Initiating Process Group	B) Planning Process Group	C) Executing Process Group	D) Monitoring &Controlling Process Group	E) Closing Process Group
1) Project Management Integration	- Solution for various project problems	- Develop Project Charter:	- Develop Project Management Plan: L-1), L-2), L-3), L-4), L-5), L-6), L-7)	- Direct & Manage Project Execution: L-1), L-2), L-3), L-4), L-5), L-6), L-7)	- Integrated Change Management: L-1), L-2), L-3), L-4), L-5), L-6), L-7)	- Close Project:
2) Project Scope Management	- Variety of function and external interface		- Scope planning & definition: L-1), L-2), L-3), L-6) - WBS definition: L-1), L-2), L-3), L-6)		- Scope Verification & Control: L-1), L-2), L-3), L-6)	
3) Project Time Management	- Short development period		- Project scheduling: L-1), L-2), L-3), L-4)		- Schedule Control: L-1), L-2), L-3), L-4)	
4) Project Cost Management	(- Fund margin of training team)		- Cost Estimating: L-1), L-2), L-3), L-4), L-7)		- Cost Control: L-1), L-2), L-3), L-4), L-7)	
5) Project Quality Management	- No detailed system specifications		- Quality planning: L-1), L-2), L-3), L-6)	- Perform Quality assurance: L-1), L-2), L-3), L-6)	- Perform Quality Control: L-1), L-2), L-3), L-6)	
6) Project Human Resource Management	- No experience of similar systems - Training maintenance engineer		- Human Resource Planning: L-2), L-3), L-7)	- Acquire Project Team: L-2), L-3), L-7) - Develop Project Team: L-7)	- Manage Project Team: L-2), L-3), L-7)	
7) Project Communications Management	- Inefficient communications among stakeholders - No efficient communication paths to external interface specifications		- Communications planning: L-2), L-4), L-5), L-6)	- Information Distribution: L-2),L-4),L-5),L-6)	- Performance Reporting: L-4), L-5), L-6) - Manage Stakeholders: L-2), L-4), L-5)	
8) Project Risk Management	- Risk to guarantee delivery date		- Risk response planning: L-2), L-3), L-4), L-5)		- Risk monitoring & control: L-2), L-3), L-4),	
9) Project Procurement Management			- Plan Contracting:	- Select Sellers:	- Contract Administration:	- Contract Closure:

WBS: Work Breakdown Structure

the following [4]:

Goal: An existing restaurant ordering system which was developed by another company is redeveloped in a short period to guarantee the delivery date and, to ensure its maintainability. This goal is broken down into the following problems:

a. Guaranteeing the delivery date
b. Short development period
c. Variety of function and external interface

Environmental Constraints: The redevelopment environment constrained the project with the following problems:

a. No detailed system specifications
b. No experience with similar systems
c. No efficient communication paths to external interface specifications
d. Inefficient communications among stakeholders
e. Training maintenance engineers

Environment Margin: The redevelopment environment has the following margin:

a. Fund margin of training team

The above-mentioned problems are described in the Column "Project problem" of Table 1. In the table, each identification of the PM patterns described in Sect. 4.2 is specified on the side of the solution of the PM pattern. To ensure that all the PM activities in the solution of the PM patterns can be executed together with other PM activities of the project, that is, there is no contradiction in the PM patterns, we have simulated all the activities. Moreover, all the problems specified in the Column "Project problem" are related to their solutions given by the PM patterns. Therefore, we can conclude that the problems of the project goal are solved by the PM patterns.

4. PM Patterns

This section describes how to define PM patterns, derives typical PM patterns to prevent schedule delays caused by changes in requirements based on the results of our empirical study shown in Sect. 2.1, and introduces other application example projects which have applied the typical PM patterns.

4.1 Definition of PM Patterns

This subsection describes how to derive PM patterns and define them. According to the PMBOK [12], all PM activities are included in the PM processes in the above-mentioned five groups, and concern the PM knowledge of the above-mentioned nine areas. Therefore, all the PM activities can be arranged on the above-mentioned framework because it is defined by the two dimensions of the five PM process groups and the nine PM knowledge areas. Using the framework, PM experts can derive a standard set of activities, as a PM pattern, to solve a certain typical PM problem on the basis of their experience.

In the practical domain of requirement elicitation, inevitably the three types of requirement elicitation processes mentioned in Sect. 2.1 are encountered. The processes of L_ and U_types frequently delay project schedules. These are the typical serious PM problems in software development. Therefore, we derived the PM patterns shown in Fig. 3 to solve the typical problems based on the results of our empirical study shown in Sect. 2.1.

In Fig. 3, the PM patterns are arranged in three layers. A lower level pattern is part of the solution for its higher level pattern [13]. That is, to realize a higher level pattern, its lower level patterns serve as solutions. Each of the patterns consists of the following items [13]:

– Pattern name: The name which identifies a pattern.
– Context: Background which limits the user, as the user of the pattern is assumed. It is here that a problem domain is pinpointed.
– Problem: Description of the problem which a pattern solves.
– Force: The situation and constraints that require the solution of this pattern to become effective. A problem domain is limited to choose a problem-solving measure.
– Solution: Solution over a problem.
– Resulting context: A newly generated problem that arises after solving a problem with the application of a pattern.

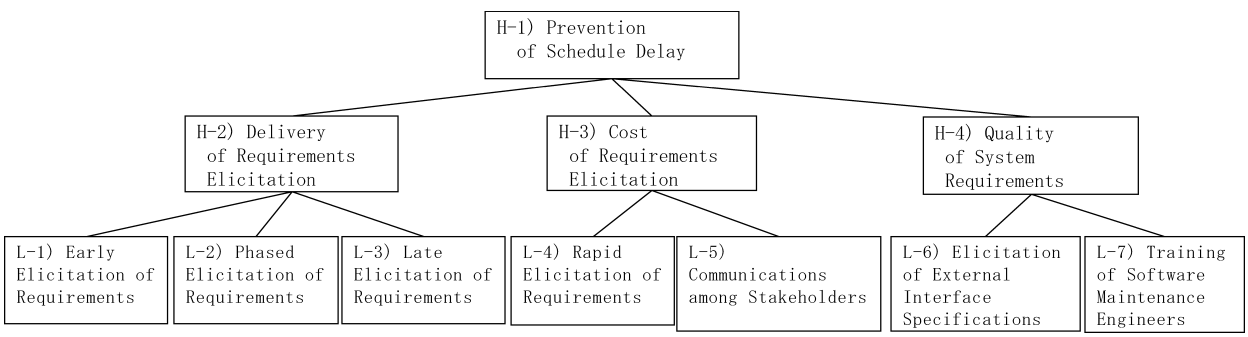

Fig. 3 Project management patterns to prevent schedule delays.

Although there are items: example and rationale in [13], this subsection omits them because of space limitation of this paper. Experienced PM experts can utilize PM patterns with the above mentioned items.

4.2 PM Patterns

In this subsection, we introduce higher level patterns and lower level patterns.
The higher level patterns are described as follows:
(H-1) Prevention of Schedule Delay

- Context: Market needs have been grasped and the system which secures customer satisfaction will be developed. Change and an addition of requirements are accepted during development.
- Problem: There is much reworking of development and a schedule is delayed.
- Force: To secure customer satisfaction, requirement changes must be accepted during development.
- Solution: The development side takes the lead, and requirements are elicited efficiently at a planned and suitable stage. Moreover, the limitation of the domain knowledge of a developer is compensated for.
- Resulting context: Requirement elicitation cannot be performed without mutual understanding and cooperation between stakeholders.

(H-2) Delivery of Requirement Elicitation

- Context: Change and additional requirements are accepted during development.
- Problem: As a result of straining oneself at an early stage and carrying out requirement elicitation, the requirement became ambiguous which led to a drop in quality.
- Force: To secure customer satisfaction, ambiguous requirement elicitation must be avoided.
- Solution: Requirements are elicited at a suitable stage of development. This includes the early stage of development, the late stage of development, and gradually throughout development.
- Resulting context: The risk of a drop in quality or schedule delays accompanies late requirement elicitation.

(H-3) Cost of Requirement Elicitation

- Context: Two or more companies co-operate with each other, and develop one system together.
- Problem: Conflicts between stakeholders so requirements cannot be elicited smoothly.
- Force: To prevent schedule delays, conflicts must be solved in the early stage. Moreover, requirements must be elicited efficiently.
- Solution: Communication between stakeholders is planned, and requirements are elicited rapidly and efficiently.
- Resulting context: A security problem concerning the communication information may occur.

(H-4) Quality of System Requirements

- Context: An inexperienced domain is developed. Moreover, a newcomer is added to the development project.
- Problem: The developer does not have sufficient domain knowledge or technological experience for development.
- Force: If the domain knowledge and engineering technological experience for development are insufficient, compensation of knowledge is necessary.
- Solution: The developers involved in development share domain knowledge, or obtain the product knowledge of a competitor privately and acquire knowledge. Moreover, the future problem is anticipated and the maintenance staff for the software is increased.
- Resulting context: There are conflicts between stakeholders and communication takes time. Schedule delays are influenced as a result.

The lower level patterns are described as follows:
(L-1) Early Elicitation of Requirements

- Context: The existing system is redeveloped.
- Problem: Since there was no documentation for the existing system, they cannot guarantee the delivery date and cost projection.
- Force: The delivery date and cost projection must be guaranteed.
- Solution: The requirements of the existing components are elicited in the early phase.
- Resulting context: If the quality of the existing components is not good, a problem may occur in the last phase of development.

(L-2) Phased Elicitation of Requirements

- Context: There are a lot of functions to be redeveloped. However, the project members do not have the domain knowledge.
- Problem: If there was no domain knowledge, a risk of decreased quality and schedule delay arises.
- Force: When a development risk is large, we have to avoid developing too rapidly.
- Solution: The requirements are elicited gradually. The customers become involved in the elicitation process [14], [15].
- Resulting context: If the process of requirement elicitation is not monitored and controlled adequately, a risk of decreased quality and schedule delay arises.

(L-3) Late Elicitation of Requirements

- Context: The system to be developed has connection with a subsystem developed by a competitor and new apparatus.
- Problem: Changing the requirements in the last phase of development may not be able to guarantee the delivery date.
- Force: The delivery date must be guaranteed.

– Solution: In the last phase of development, skilled engineers analyze the requirements needed for change, and determine whether they should be accepted or left for the next version.
– Resulting context: If the process of requirement elicitation is not monitored and controlled adequately, a risk of decreased quality and schedule delay arises.

(L-4) Rapid Elicitation of Requirements

– Context: A system is developed in a short development time.
– Problem: The usual requirement elicitation may cause schedule delay.
– Force: The delay of requirement elicitation must be avoided.
– Solution: The special measures are planned, and requirements are elicited rapidly.
– Resulting context: If the process of requirement elicitation is not monitored and controlled adequately, a risk of cost overrun and schedule delay arises.

(L-5) Communications among Stakeholders

– Context: Two or more stakeholders work together to develop the system.
– Problem: The communications among the stakeholders are inefficient. Therefore, they may cause a schedule delay.
– Force: Confrontation must be resolved in the early stages and cooperation must prevail.
– Solution: Problem solving and a progress situation are shared among stakeholders, and communication is planned.
– Resulting context: A security problem concerning the communication information may occur.

(L-6) Elicitation of External Interface Specifications

– Context: The system is connected to a product developed by another company.
– Problem: There is no information on the product specifications among the development team. Moreover, there is no efficient communication path to obtain the specifications.
– Force: We have to acquire the knowledge of interface specification by a certain method.
– Solution: The maker of product is required to provide its interface specifications. The developer participates in the presentation meeting held by the maker. Thus, the requirements are elicited.
– Resulting context: If the specifications obtained in the open presentation meetings are insufficient, a problem of interface mismatching arises in the integration test phase.

(L-7) Training of Software Maintenance Engineers

– Context: To extend a system, it is maintained and developed continuously.
– Problem: There are no software engineers who possess the knowledge of the system requirements.
– Force: We have to train a specialist with domain knowledge.
– Solution: Software maintenance engineers are trained from the early phase.
– Resulting context: The cost of training is incurred.

4.3 Application Examples of PM Patterns

In this subsection, we describe application examples of actual projects which have applied PM patterns. The first and third examples are business systems. The second example is an FA (Factory Automations) system. The fourth example is an embedded system. After explaining the features of the projects, we describe how PM patterns were utilized by indicating the identifications of patterns with parentheses.

(1) Restaurant ordering systems
This is a system for managing orders of a restaurant. Two or more stakeholders co-operate with each other, and redevelop the existing system together [4]. Four problems and solutions on development are described as follows:

-Problem 1: There was almost no development documentation for the existing system, and details were not known.
Solution: Requirements of the existing system were elicited by on-site inspection in the early stages of development (L-1).
-Problem 2: Since it was an inexperienced domain, there was no domain knowledge, and the validity of the requirements could not be judged.
Solution: The customer requested a person who knew the existing system well to participate in the design meeting, and the requirement specifications were elicited gradually (L-2). Moreover, knowledge was obtained by listening to the customer (L-5).
-Problem 3: The cooperative relationship between stakeholders was not established, and development efficiency fell.
Solution: The negotiations with the subcontractors for their cooperation were carried out by the customer (L-5).
-Problem 4: Since the interface specifications of the apparatus were not provided, a design schedule was affected by it.
Solution: The customer required the maker of apparatus to provide its interface specifications, and the developer participated in the presentation meeting held by the maker (L-6). It enabled the quick elicitation of the requirements (L-4). Although the unexpected requirements of the apparatus connection were elicited at the integration test phase (L-3), a skilled engineer was added to the project.
-Result: Although requirements were elicited during the development, there was almost no delay to the schedule.

(2) Document storage systems
This is an automatic warehouse system for raising the

storage efficiency of documents. Three problems and solutions on development are described as follows:
-Problem 1: The standard system of an automated warehouse was customized and no customer requirement was available.

Solution: The design was advanced together with the customer from the stage of deciding requirement specifications (L-5), and the requirements were elicited gradually (L-2).
-Problem 2: Although this system had an interface with equipment, it was the first project for the developer to use the equipment and an experienced person was not available.

Solution: The interface specifications were negotiated with the maker of the equipment (L-6), and became to be available in the early stages (L-1).
-Problem 3: The equipment was not developed according to the specifications. It was found in the integration test phase.

Solution: To solve the problem, the software requirement change was elicited rapidly in the late stages (L-3, L-4).
-Result: Although the date for delivery was shifted by two months for the customer's convenience, there was almost no delay in the schedule.

(3) Medical ordering systems
This is a redevelopment of an existing medical ordering system, and the basic portion of a new system was developed using a medical software package. The subject of the project was how to redevelop a medical ordering system by utilizing an existing medical software package. Two problems and solutions on development are described as follows:
-Problem 1: There was no knowledge of a medical software package.

Solution: The design working group started working with the customer and the medical software package maker, and the knowledge was obtained (L-5). The requirements were elicited gradually (L-2).
-Problem 2: The timing of the contract with the package maker was delayed, and the requirements of a package could not be elicited timely.

Solution: Requirement elicitation by using the package in practice took a long time (L-3). However, requirements were quickly elicited by requesting strong cooperation from the maker (L-4, L-5, L-6).
-Result: Although the date for delivery was shifted by two months, the situation was understood by the customer and it did not become a problem.

(4) Data gathering systems
This is a system which gathers power consumption data. This system had connections to an apparatus controller which another maker developed. Two problems and solutions on development are described as follows:
-Problem 1: Since the interface specifications of the apparatus controller were not provided by the maker, whether the requirements of the system could be realized was not judged.

Solution: The purchase of the apparatus controller was requested from the customer in the early stages of development (L-1, L-6). Furthermore, the interface specifications were elicited quickly (L-4).
-Problem 2: The developer did not have development experience of the apparatus connection.

Solution: The technical assistance and a review by an experienced person were carried out. Furthermore, training maintenance engineers was scheduled (L-7).
-Result: The above-mentioned matter was carried out and managed in the daily meetings. The development was completed without decreased quality or delays in delivery.

Table 2 Application of project management patterns.

PM Pattern IDentification \ Project Area	Restaurant ordering systems	Document storage systems	Medical ordering systems	Data gathering systems
	Business	FA	Business	Embedded
L-1	○	○		○
L-2	○	○	○	
L-3	○	○	○	
L-4	○	○	○	○
L-5	○	○	○	
L-6	○	○	○	○
L-7				○

Table 2 summarizes the application of PM patterns to the example projects. The left vertical axis shows the identification of PM patterns, and the horizontal axis shows an application system name. The round mark in Table 2 means an application, while a blank refers to those without an application.

5. Discussion

The PM methods similar to the patterns described in Sect. 4.2 were used in many projects. Since some of the projects were successfully completed like the examples mentioned in Sect. 4.3, the PM patterns are useful. However, some of the projects failed. One of the main causes of the failures was that an unexpected large quantity of requirements to be elicited arose in the later stage of U_type processes. This kind of failure was found in embedded system projects in which hardware and software were concurrently developed. When a problem within the hardware was decided to be solved through the software, an unexpected large quantity of software requirements arose. This delayed the project schedules. This is a kind of limitation in applying PM patterns. The PM patterns for U_type processes should be carefully applied with necessary and sufficient risk management. In the future, the risks should be analyzed and incorporated into PM patterns.

PM patterns do not include the whole PM knowledge, but provide a portion of it. Therefore, PM experts who have the whole PM knowledge can utilize the patterns, but non-experts of PM hardly use the patterns immediately in an

appropriate manner. This is another kind of limitation of applying PM patterns. At present, PM patterns are useful for PM experts to manage projects efficiently and train non-experts of PM.

After PM patterns are selected for applying to a project, no contradiction whether all the patterns can be utilized together in the same project must be approved to make an efficient schedule for the project. No contradiction can be approved by simulating all the solution activities defined by the patterns first in each of the matrix cells in the framework, second in each of the PM knowledge areas, third in the whole project. A project developing a large scale system is divided into subprojects each of which develops an architectural component of the system. In such a case, no contradiction should be approved by simulating them in each of the subprojects, and then in the whole project.

In the future, short-term requirement elicitation in complicated project environment is one of the most serious concerns of practical software engineering. Therefore, we will concentrate our study on PM patterns for such requirement elicitation. For example, we will define PM patterns which have conditional solution to meet complicated development environment. Furthermore, we will define specific PM patterns to individual member roles in projects in order to make project organization efficient.

6. Conclusions

In this paper, we proposed typical PM patterns to prevent schedule delays caused by requirement elicitation. Moreover, we proposed a framework for implementing project goal through the PM patterns, and approving no contradiction among the PM patterns. This study showed that the PM patterns for preventing schedule delays were useful.

In the future, short-term requirement elicitation in complicated project environment is one of the most serious concerns of practical software engineering. Therefore, we will concentrate our study on PM patterns for such requirement elicitation.

Acknowledgments

This work was funded by the Joint Forum for Strategic Software Research (SSR). We would like to thank Ms. Mari Inoki and Mr. Michio Tsuda at SSR. We would also like to thank Mr. Masao Shimoji and his colleagues at the Yaskawa Information Systems Corporation who cooperated on our study as interviewees.

References

[1] J. Johnson, "Chaos: The dollar drain of IT project failures," Application Development Trends, pp.41–47, 1995.
[2] A.M. Davis, Just Enough Requirements Management: Where Software Development Meets Marketing, Dorset House, 2005.
[3] T. Nakatani, S. Hori, N. Ubayashi, K. Katamine, and M. Hashimoto, "A case study: Requirements elicitation processes throughout a project," Proc. 16th International Requirements Engineering Conference (RE'08), pp.241–246, 2008.
[4] S. Hori, T. Nakatani, N. Ubayashi, K. Katamine, and M. Hashimoto, "Towards risks avoidance by efficient requirements elicitation," Proc. Society of Project Management (in Japanese), pp.470–475, 2008.
[5] S. Hori, T. Nakatani, N. Ubayashi, K. Katamine, and M. Hashimoto, "Project management patterns to prevent schedule delay caused by requirements changes," Proc. ICSOFT 2009 (in Bulgaria), pp.115–120, 2009.
[6] C. Alexander, The Timeless Way of Building, Oxford University Press, 1979.
[7] T. Iba, Y. Yumura, T. Wakamatsu, and K. Furuichi, "Proposing and evaluating a pattern language for promoting project," http://ilab.sfc.keio.ac.jp/2007/output/files/jwein2007-pattern.pdf, 2007.
[8] A. Hatch, L. Burd, C. Ashurst, and A. Jessop, Project Management Patterns and the Research-Teaching Nexus, http://www.ics.heacademy.ac.uk/events/8th-annual-conf/Papers/Andrew%20Hatch%20final.pdf
[9] I. Stamelos, "Software project management anti-patterns," J. Syst. Softw., vol.83, pp.52–59, 2010.
[10] K. Murakami, K. Watanabe, S. Satho, and Y. Akasaka, "Extraction and development activities of the "best practices" based on SECI model," Proc. Society of Project Management (in Japanese), pp.390–394, 2007.
[11] M. Rudolf, Patterns of Project Management Success, http://www.sandhill.com/opinion/editorial.php?id=173, 2008.
[12] Project Management Institute, A Guide to the Project Management Body of Knowledge (PMBOK Guide), third ed., PMI, 2004.
[13] L. Rising, Customer interaction Patterns, http://jerry.cs.uiuc.edu/~plop/plop98/final_submissions/P11/P11.htm, (Pattern Languages of Programs'98), 1998.
[14] E. Carnel, R. Whitaker, and J. George, "PD and joint application design. A transatlantic comparison," CACM, vol.36, no.4, pp.40–47, 1993.
[15] K. Beck, M. Beedle, A. Bennekum, A. Cockburn, et al., Manifesto for agile software development, http://agilemanifesto.org/, 2001.

Shozo Hori graduated from the Kyushu Institute of Technology information engineering in 1980, and working with the Yaskawa Information Systems Corporation. He has performed embedded software development and management for 20 years. He has been engaged in process improvement of an organization using ISO9001 or CMM since 1998. His interest area is software engineering, requirements engineering, and project management.

Takako Nakatani graduated from Tokyo University of Science, she worked for several software development companies. She received Ph.D. form the University of Tokyo in 1998. She has been a board member of S-Lagoon since 1995, and an associate professor of University of Tsukuba since 2006. Her interest area is software engineering, requirements engineering, and modeling techniques.

Keiichi Katamine received the Ph.D. degree in engineering from Kyushu Institute of Technology, Japan. He is an assistant professor at Kyushu Institute of Technology. He is a PSP instructor of Software Engineering Institute of Carnegie Mellon University, and a trainer of S-DBR and CCPM of Goldratt School. His research interests include software engineering, software and knowledge modeling techniques, and project management. He is a member of the Information Processing Society of Japan and the Society of Project Management.

Naoyasu Ubayashi is an Associate Professor at Kyushu Institute of Technology. He received his Ph.D. from the University of Tokyo in 1999. He is a member of ACM SIGPLAN, ACM SIGSOFT, IEEE Computer Society, and Information Processing Society of Japan (IPSJ). He received "IPSJ SIG Research Award 2003" from IPSJ. His current research interests include aspect-oriented programming, extensible languages, and model-driven development.

Masaaki Hashimoto received his M.S. in electronics in 1970 from Kyushu University. He joined NTT in the same year. From 1993, he was a Professor at Kyushu Institute of Technology. He retired in 2009 and is a Professor Emeritus of the Insitute. He is a PSP instructor of SEI of CMU, and trainer of SDBR and CCPM of TOC. His current research interest includes software engineering, project management and knowledge engineering. He is a member of the Information Processing Society of Japan and the Society of Project Management.