行政院國家科學委員會專題研究計畫　期中進度報告

建立一符合軟體成熟度模式 Level2 之需求管理及 Level3 之需求開發工作流程模板(1/3)

計畫類別： 個別型計畫
計畫編號： NSC94-2213-E-009-112-
執行期間： 94 年 08 月 01 日至 95 年 07 月 31 日
執行單位： 國立交通大學資訊工程學系(所)

計畫主持人： 王豐堅

報告類型： 精簡報告

處理方式： 本計畫可公開查詢

中　華　民　國 95 年 6 月 28 日

行政院國家科學委員會專題研究計畫期中報告

計畫編號：94-2213-E-009-112-
執行期限：2005.08.01 至 2006.07.31
主持人：王豐堅　國立交通大學資訊工程系
計畫參與人員：王靜慧等研究生

Abstract.
Fast change of information techniques introduces the considerable needs of better software integration and generation over various hardware, operating systems, and applications. Facing the highly changeable environment, software developers encounter more and more difficulties and challenges. To extract the requirements of projects is getting more difficult; however, there is no perfect solution currently.
Capability Maturity Model Integration (CMMI) is the most popular approach that can be used to guide process improvement across a project, a division, or an entire organization. One CMMI benefit expected is increased focus and consistency in requirement development and management. There are lots of researches which address related problems in requirement development process, but none of them consider the overall areas of requirement development process. Identify project requirements from particular perspectives reduces the successful opportunity. Furthermore, all of them did not follow the CMMI which take account with all phases of requirement development, so we provide a model which could supports the guidelines of requirement development process area of CMMI level 3. If developers use the model, it can help the organization to achieve the goals of Requirement Development (RD) process area of CMMI Levels 3.

Keyword: Requirement Development, CMMI.

1. Introduction

During requirement development, we encounter many problems that occur again and again. The question we must ask ourselves is how we are going to solve it this time. Documenting useful models is one way that you can reuse. The information associated with the documents that represent how it is better to solve the requirement development problem [1].
On the other hand, CMMI is a process improvement approach that provides organizations with the essential elements of effective processes. It can be used to guide process improvements across a project, a division, or an entire organization. CMMI is claimed to help integrate traditionally separate organizational functions, set process improvement goals and priorities, provide guidance for quality processes, and provide a point of reference for appraising current processes [2-5], but there is not a systematic approach to get the realistic benefit from this process improvement approach.
In addition, requirement development is the hardest work of software life cycle, but it is the most important factor to decide the success of project. Based on above observation, we decide to propose a model to help developers achieve the targets of Requirement Development process area in CMMI Levels 3.
Our model elaborates requirements from goal [1][4][9][19][20], use case [14][21][22], and scenario [8][11-13] view points according to the practices of RD process area in CMMI Level3. The model realizes the requirements specification by questionnaires. Recommendable requirement development processes are provided in CMMI. Our model follows the order of these processes to provide corresponding questionnaires as well as to get the expected products, which fulfill the goals then.
The report is structured as follows: In Section2, we present a short survey on requirement development methodologies, focusing mainly on the relevant researches of CMMI, software patterns, and goal-driven, use-case-driven, scenario-driven approach. In Section 3, we present our model, giving both template and semantics. Our model is defined as a set of questionnaires and a high level execution process. We also present a Workflow system that is used throughout the research to illustrate the presentation. Conclusion and future direction of our work are given in Section 4.

1 The modification of system objectives could violate the organizational needs.

2. Related Works

2.1. CMMI and Related Researches

The first version of Capability Maturity Model (CMM) [2] was released in 1991. The CMM has evolved to CMMI [3], which enables the continual growth and expansion of the CMM concept to multiple disciplines, such as system engineering, software engineering, integrated product and process development, and supplier sourcing.
Requirement Development (RD) Process Area (PA) is concerned necessarily in Level 3 of Capability Maturity Model Integration (CMMI) [4][5]. In RD-PA3 (RD-PA in Level 3), there are several goals to be achieved. Software developers might achieve the specific goals based on the generic goals. RD-PA3 has three types of requirements: customer, product, and product-component requirements. These requirements address the needs of relevant stakeholders, product attributes and constrains for design decision. Above goals are implemented by practices that not only cover the products, but also consider their generation processes and limitations from stakeholders. There is a method [7] creating a meta-model which represents the relationships among elements such as organization policy, restricted resources, and functional requirements. The design of the meta-model is emphasized on requirement management and elicitation traceability, but disregarded other influence factors. This model does not cover the whole improvement essentials of RD-PA3, so we provided a model which could further the organization maturity to RD-PA3.

2.2. Current States of Requirement Engineering

The Goal-Driven (GD) [1][4][9][19][20] approach elaborates software requirement from high-level goals to low-level elements. It also can support goal refinement and element identification. The identification of an element such as component, subsystem, system, et al. contributes to its subject establishment in Use Case-Driven (UCD) approach [14][21][22]. UCD approach aims at functionality refinement and execution scenario extraction of each element identified from GD approach. Scenario Based-Driven (SBD) [8][11-13] approach displays actor execution scenario in interaction diagram, e.g., message sequence chart, sequence diagram of UML, swimlane chart of UML. SBD approach combines scenario(s) of each element to form a complete description of system behaviour. UCD approach can make up for GD approach and the insufficiency of UCD approach can be remedied by the SBD approach. It could earn below benefits to apply three approaches together to analyze software requirements:

1. Support requirement development from elaboration to validation;
2. Provide traceability because the relationships among the products of each approach are generated (automatically);
3. Higher reliability because the software requirements are extracted from original goals;
4. Higher maintainability supports by the requirement traceability;
5. Higher readability because by displaying execution scenario with interaction diagrams and representing functional requirements with use case diagrams;
6. Support many analysis methodologies [8][11-13] of interaction diagrams.

3. The Model

Our model develops and elaborates software requirements from goal, use case, and scenario view points based on the recommended processes of RD-PA3 [3-5]. The model realizes the requirements specification by questionnaires. The replies of questionnaires can be used to construct the work products of RD-PA3.The recommended process has three stages which are development customer requirements, development product and product-component requirements, and analysis and validation requirements.
In development customer requirements stage, developers elicit needs from various stakeholders with questionnaires designed for translating the high-level goals into detailed software functions. Our model also proposes a method to consolidate various inputs from the stakeholders. Moreover, there are two refinement sub-models, Customer Requirement Derivation Model CRDvM and Customer Requirement Decision Model CRDcM, introduced to help developers to obtain more information by reconsidering the replies of TSQ and resolve conflicts by making decision with the information recorded.
All requirements gotten from stakeholders are represented in a domain specific language. Our model transforms customer requirements into technological requirements in development product and product-component stage with actor and use case identification concept. The model establishes product and product-component requirements by identifying the elements boundary, eliciting functional requirements from relevant actors, and defining the communication interfaces of each product and product-component.

The last stage of the recommended process is not concerned in this year.

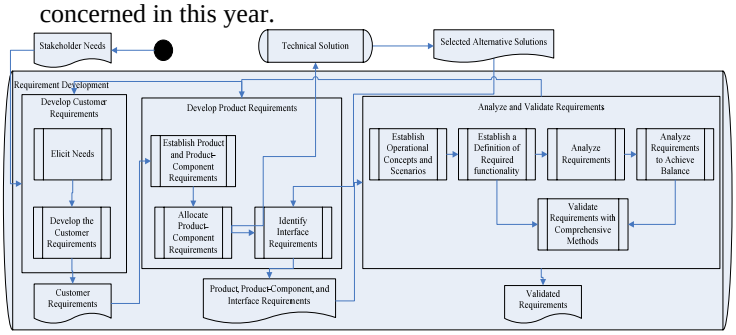

Figure1. The requirement development recommended process of RD-PA3

1 The modification of system objectives could violate the organizational needs.

4. Development customer requirements

Somerville and Sawyer [24] defined a stakeholder as "anyone who has a direct interest in or benefits from the system that is to be developed". A stakeholder could be a project manager, a marketing people, an end-user, a software engineer, a support and maintenance engineer, et al. All information from stakeholders could directly influence the determination of customer requirement adoption [25]. The analyzed results can be transformed into a set of customer requirements.

4.1 Stakeholder selection

One of the difficulties in above works is extracting useful stakeholder(s). The information from a non-appropriate stakeholder neither helps requirement development, nor reduces the development complexity. The stakeholders influence continually the requirement life cycle. Therefore, our model identifies appropriate stakeholders before development customer requirements.

For a project, its importance, system objects, urgency degree, consuming effort might be the most significant factors to affect the selection of stakeholders. Based on these factors, our model is designed with below questions used to identify project stakeholders.

(1) How important is the targeted system for the organization? – The targeted system may be developed because of strategic or operational needs of the organization. The necessity degree of the envisioned software can decide the importance of the target system. An organization could promote stricter criteria of stakeholder selection to more important systems.
(2) What are the objectives of the targeted system? – The organization needs are referred to modify the objects of the targeted system.
(3) How urgent is this project for the organization? – The urgency degree of the project was decided against time limitation, budget of the project, and personnel of the organization. The evaluation of urgency degree is supported by below questions:
    (a) How much available time do you have?
    (b) How much development time does the software need?
    (c) How much cost is needed to build the system?
    (d) How much is the price of the system?
    (e) How many personnel should be used?
    (f) How many personnel can be used?
(4) Who will use the system? – A user interacts with the system and gets the direct benefit from the functionalities provided from the system.

Our model classifies the candidates into four groups (shown in Table 1): development team, supporting team, business team, and users. The developer refers the answers of questions to choose appropriate members into development team, supporting team, and business team.

A system serves various types of users who could play different actors of the system. An actor is interested in some functionalities of the system and interacts with system on specific behaviours. Consequently, all actors of the system are the necessary stakeholders whose use behaviours and functional needs are essential factors that should be considered.

4.2 Needs Elicitation

The requirements of the system will be explored from various points of view. For example, the marketing group is interested in the functions and features that will excite the potential market. End-users may want the features they are familiar with and that are easy to learn and use. In this step, our model provides a set of questions performed in three steps, TSQ, to elicit stakeholders' needs in a proper sequence. TSQ helps developers to progress stakeholder needs step by step.

1. The questions in TSQ's first step

The questions in the first step focus getting contents from stakeholders on goals, market space, economic benefits and limitations. This step helps an organization to evaluate how much confidence and agreeableness stakeholders have for this targeted system.

(1) Do you agree to develop this software? – This question helps an organization to indicate the stakeholders who agree to build the software.
    Who could request for this targeted system? –The requesters are the major customer group. This question contributes to find out the major market of the product.
(2) Who could use the solution? –This question helps to indicate the users who may use this software. The estimation of expected future user group determines the size of potential market of the targeted system.
(3) What are the major functionalities of the targeted software? – This question is designed to gain the (kernel) essential functionalities from various stakeholders' view points.
(4) What economic benefits will be gained from a targeted system? – This question tries to understand the expectation of economic benefits expectation targeted benefits of the system from stakeholders.
(5) Are there other sources, such as unfamiliar skill, software, and hardware, needed for developing the targeted software? – This question helps developers to evaluate the cost of extra requests.

2. The questions in TSQ's second step

The questions in the second step help stakeholders to gain a better understanding of the targeted system. These questions are designed in order to retrieve the functionalities, constraints, potential user groups, and execution environment of the targeted system from each stakeholder.

(1) How many kinds of users for the targeted systems and who are they? – The question guides stakeholders to reply by referring to the results of question (4) in stakeholder selection and questions (4) in stakeholder selection and questions (2) & (3) in step one and thus to identify the user types

1 The modification of system objectives could violate the organizational needs.

(actors). Moreover, these actors can be treated as necessary stakeholders.

(2) What kind of execution environments do the targeted systems operate in? –This question attempt to gain the information about the execution environment of the targeted system for (software) design and implementation.

Table1. The template of stakeholder selection

Project ID: p00001		Project Name: workflow management system			
Description: workflow management system supports electronic office.					
How much importance of the targeted system for the organization?	■ very important	☐ important	☐ common	☐ less important	☐ not important
How urgent is this project for the organization?	☐ very urgent	■ urgent	☐ common	☐ less urgent	☐ not urgent
How much available time do you have?	Six months				
How much development time does the software need?	Five months				
How much cost is needed to build the system?	1 million				
How much is the price of the system?	2 million				
How many personnel should be used?	One manager, one project manager, one software designer, two software engineers, two software engineers, one marketing, and one sale.				
How many personnel can be used?	One project manager, one software designer, one software engineers, one software engineers, one marketing, and one sale.				
What are the objectives of the targeted system?	1. Providing a workflow management system to support office workflow automation. 2. The workflow management system provides the workflow creation function, organization structure design function. 3. Each employee in the company can use this system to deal with the traditional paper work.				
Who will use the system?	Every employee of the company adopted the workflow management system could be the user.				

stakeholders belong development organization			
			The importance of someone for the project
Development Team	Requirement Analyzer/ modeller	Software Designer/e0006/Lily	8
	System Architecture Designer	Software Designer/ e0006/Lily	8
	Implementer	Software Engineer/e0007/Tina	7
		Software Engineer/e0008/Tom	7
		Software Engineer/e0009/Ben	7
		Software Engineer/e00010/Lo	6
	Tester	Software Engineer/e0007/Tina	7
	Deplorer	Software Engineer/e0008/Tom	7
	Maintainer	Software Engineer/e0009/Ben	7
	Others		
Supporting Team	Project Monitor	Project Manager/e0003/Tom	8
	Administrator	Manager/e0004/Chris	10
	Domain Expert	Domain Expert/e0005/Bob	8
Business Team	Marketer	Marketing/e0001/Joy	6
	Sales	Sale/e0002/Mary	6
	Customer Service	Sale/e0002/Mary	6
stakeholders belong to an external organization			
User types	UserType1/NA/Grant		10
	UserType2/NA/Mick		10
	System Manager		8

Notes:
1. The representation form of the stakeholder is organization role/employee id/employee name.
2. The importance of the stakeholder was divided into 10 degree from 1 to 10. 10 is the highest degree. It decreases progressively from 10 to 1. 1 is the lowest degree.

(3) What are the detailed functions of major functionalities in targeted software? – This question guides stakeholders to refer to the suggestions from the identified actors to refine the results of question (4) in the first step further.

(4) What characteristics should target software possess? – The question identifies the stakeholders' expectancy of the targeted system. This information provides to non-functional requirements identification.

3. The questions in TSQ's third step

An appropriate stakeholder can give more suitable answers. On the contrary, the information from a non-appropriate stakeholder can not help requirement development, but increase the development complexity. The quantity of questions is another factor to affect the quality of the answers. Moreover, the questions in steps one and two don't cover all relevant items. If stakeholders want to provide additional information, the questions in this step can help derive the data. A selected stakeholder may indicate someone who is not in the stakeholder list but could contribute toward the requirement development. The questions include:

(1) Are you the right person to answer these questions?

(2) Are your answers official?

(3) Am I asking too many questions?

1 The modification of system objectives could violate the organizational needs.

(4) Can anyone else provide additional information?
(5) Should I ask you anything else?

4. The operation policy of TSQ

TSQ is able to help developers adjust the stakeholder candidates and extend the information collection capability, but that is not enough for guaranteeing the quality of answers. Therefore, all answers we get from stakeholders are required to satisfy the following conditions:

(1) All TSQ repliers should be included in the stakeholder list.
(2) All questions in TSQ need be filled.
(3) When an extra stakeholder is found in the third step, the stakeholder list should be reconsidered. This stakeholder could be added into or replaced with someone in the stakeholder list. The modification policy will be discussed below.
(4) The answer(s) of Question (5) in TSQ step three should be recorded correspondingly.

There are two cases for the modification in (3):

Case1: The stakeholder belongs to the development organization.

The key problem here is the role he/she will play. Every stakeholder has a chance to provide his/her opinion. The degree of influence is calculated based on the importance of the person for the project. If the calculation result is grater than the threshold, the stakeholder will be put into the consideration list.

Each candidate stakeholder in consideration list is corresponding to a data item set $\{si_1, si_2,\ldots si_n\}$ used to introduce a new stakeholder. An item si_k, $1 \le k \le n$, represents a distinct activity type (e.g., replacement in the stakeholder list or add the stakeholder to stakeholder list) for adjusting candidate stakeholder list. Each stakeholder has right to select his own activities and the importance of si_k, is counted by summing all the importance values of the stakeholders who vote si_k,.

Case2. The stakeholder belongs to an external organization.

The key problem here is the user type identification. There is a candidate stakeholder set. Each stakeholder in the set has one or more corresponding activity for adjusting user type list which consists of users of different varieties. The preliminary list is generated by stakeholder selection. The decision strategy of the candidate stakeholders is the same as that in Case 1.

4.3 Specifying the Customer Requirement

The organized ability is important for showing potential information from the elicited answers. The various inputs from the stakeholders must be consolidated, missing information must be obtained, and conflicts must be resolved when documenting the recognized set of customer requirements.

1. Consolidation for the replies from TSQ

The replies from TSQ should be consolidated as a set of statistical values of the stakeholders' opinions. These values will be used in the following step. The replies are mostly written with natural language and their analysis is based on the semantic meanings. Here is not concerned with natural language related problems. Instead, we just provide a method to do information classification.

Let the set of replies of a question q_j be $q_{j\cdot ans}$, $q_{j\cdot ans}=\{a_1,a_2,\ldots,a_n\}$. The set of reply classifications of q_j is denoted as $q_{j\cdot class}$, $q_{j\cdot class} = \{c_1, c_2\ldots\}$. Each element c_i in $q_{j\cdot class}$ is a set of the stakeholders who give the same opinion. The answers from different stakeholders may not equal, so we define another set $q_{j\cdot sim}$ whose elements contain two tuples to represent the similarities: The first is an element in $q_{j\cdot ans}$ and the second in $q_{j\cdot class}$. Two elements in $q_{j\cdot sim}$ which have the same replay but different classifications indicate that the answer is similar to both classifications. Two elements which have the same classification but different answers indicate that both answers are similar to the classification.

Each classification c indicates the stakeholders who give the replies classified as c. Here, c's importance, $c_{\cdot imp}$, is defined as $\sum_{i}^{n} s_{i.imp}$, where s_i is a stakeholder in c and

```
c1·value = a1, c1 = {s1}and PUT c1 into qj·class
For from i=1 to i=n-1, n is the number of stakeholders;
    For from x=1 to x=the size of qj·class
        COMPARE (cx·value, ai+1)
        IF the value of cx·value equal to the value of ai+1
        THEN cx = cx ∪ { si+1} BREAK;
        IF the value of ai+1 is partial equivalence to the value of
        cx·value
        THEN qj·part = qj·part ∪ {cx, ai+1} BREAK;
        IF the value of ai+1 does not equal to the value of cx·value
        THEN qj·class = qj·class ∪ {ci+1} and ci+1={ s1+1} BREAK;
      END
END
```

$s_{i.imp}$ is the importance value for s_i in the project considered. The higher $c_{\cdot imp}$ is, the more people have the same opinion on c. Moreover, from the classification sets, we can know the opinions of various roles and the distribution of stakeholders in these classifications.

2. Customer requirements management

TSQ supports developing preliminary customer requirements. The following development proceeds with the factors such as organizational needs, system objectives, critical success factors, requirements and mandates. TSQ helps developers elicit stakeholders' requirements without explicit identification of semantics of the relationships among these factors. Traceability would greatly benefit requirements management, facilitating requirements understanding, capture, tracking, and verification. A Customer Requirement Derivation Model (CRDvM) is created here for supporting stakeholders to 1) establish traceable links to model requirement dependencies, 2) develop requirements taking account of organizational needs, critical success factors, and mandates, and 3) generate requirement change proposals based on system objectives.

CRDvM represents the interaction relationships among software requirements, organizational needs, resource limitation, constraints, and etc. The replies from TSQ instantiate the elements of CRDvM. Developers follow this model to concern direct or indirect relationships

1 The modification of system objectives could violate the organizational needs.

during the successive customer requirements progress. The structure of CRDvM is shown as Figure2.

The details of CRDvM are described based on the six components below:

(a) Organizational Needs

A targeted system is built to preliminary satisfy organizational needs which could either be long term strategic needs or short term operational needs. An organizational administrator instantiates the

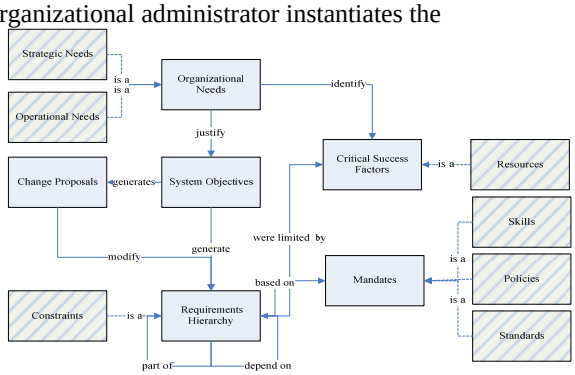

Figure2. The structure of Customer Requirement Derivation Model

organizational needs at very high level. The needs could influence the objectives of the targeted system.

(b) System Objectives

Stakeholders such as customer, program manager, programmer, etc., specify the system objectives. The most primary objectives are gotten at stakeholder selection step and the objectives are revised corresponding to question (4) and (5) at the first step of TSQ. The objectives of the system are also adjusted based on the organizational needs.1

(c) Critical Success Factors (CSFs)

Organizational needs help developers identify some critical success factors. Resources such as cost, time, budget, etc. are examples of CSFs, so the factors concerned in stakeholder selection are CSFs also. Requirements for the system are affected by these CSFs. For example, the time that remains for the development of a project decides the urgency degree of the project. The essential cost is one factor to decide the profit of the software product. During the negotiation associated with the stakeholders, many trade-offs are made in deciding the scope and functionality of the system depending on their CSFs.

(d) Mandates

Requirements development is usually associated with standards, policies, and procedures. These constraints could reduce the flexibility of requirement development. The question (6) in second step of TSQ collects the extra matters needing attention in software development.

(e) Requirement Hierarchy

The hierarchy of requirements abstraction level maintains linkages between each requirement and its sub-requirements created during the requirement development. The requirements of different significance or criticalities are evaluated; requirements may be traced through the lifecycle at different levels.

(f) Change Proposals

Change proposals are extracted from system objectives. The proposals are used to guide the modifications of software requirements.

There may be conflicts among specification, elaboration, decomposition, derivation and modification of requirements, due to different interpretations, assumptions, interests, viewpoints, experience, or objectives of the stakeholders. Information for resolving these conflicts must be maintained throughout the system lifecycle to ensure that customer requirements are understood and satisfied. Therefore, this research proposes a Customer Requirement Decision Model (CRDcM) to support developers to make a better decision.

The relationships of requirement development influenced factors are displayed in CRDvM. The state of the instances of these identified factors could be changed constantly, so it is hard to make a strategic decision during development time. CRDcM is designed to record the progress of a decision making to cope with the changeable environment.

Each type of *object* generations during software development is based on a distinct *rationale*. Both objects and generations could raise their own *conflicts*. CRDcM supports the developers to record the cause of conflicts. The developers evaluate the *alternatives* of each conflict to derive its *arguments* with external *assumptions*. A *decision* making was influenced by *critical success factors*, and *rationale*. CRDcM allows each developer to keep a record of progress of a *decision* making.

The details of CRDcM are described as follows:

(a) Conflicts

The produced objects of requirement development such as customer requirements, organizational needs, and system objectives are related one another; therefore an object modification could raise a conflict(s). Such a modification can be a revision of organizational needs, customer requirement decomposition, or system objective adjustment.

(b) Decisions

Facing these conflicts, the developers could find one or more solution based on some critical success factors. It is assumed that each candidate solution is associated with some arguments for a possible situation. The arguments could support or oppose the solution. Developers make decision in accordance with the arguments, rationale, and resource limitations. All of these inter-mediums are maintained throughout the software lifecycle.

(c) Rationale

In general, rationale construction provides large profits for software development. However, the overhead is high in capturing detailed rationale because it is usually lack of tools for help. CRDcM displays the relationships between the rationale and other factors which maintain the mutual interaction. Again, the details regarding rationale constructions are not discussed here because of space limitation.

1 The modification of system objectives could violate the organizational needs.

5. Development Product Requirements

Customer requirements are analyzed in conjunction with the development of the operational concepts to derive the sets of more detailed and precise requirements called “product and product-component requirements.” In product requirement development

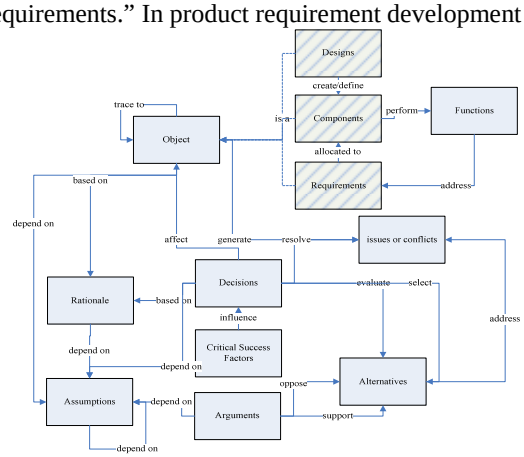

Figure3. The structure of Customer Requirement Decision Model

stage, developers divide the whole software requirements into several product or product-component problems. Deriving the product and product-component requirements addresses the operational concepts implied in customer requirements, the limitations of the architecture to be selected, the design of the targeted system, and the distinct business considerations of developers. There are three essential tasks to develop product requirements which are product and product-component *identification*, *establishment*, and *allocation*.

The incompleteness and inconsistency of requirements are the most general problems during requirement development. A correct identification of a product and product-component boundary could lighten the efforts, so deciding the boundary of the targeted system is very important for establishing product and product-component requirements.

The boundary recognition of product and product-component should be supported by the component identification. Boundary recognition and identification of product and product-components are performed mutually recursively. Besides, after each identification step, our model provides several questions to help developers obtain the actors related to each product or product-component directly. Getting component requirements from these actors could increase the requirements completeness and consistency. The detailed information is shown as below:

(1) Product Component Identification

According to Szyperski [13], the characteristic properties of a component are that it: 1) is a unit of independent deployment, 2) is a unit of third-party composition, and 3) has no observable state. He gives the following definition of a component:

A software component is a unit of composition with contractually specified interfaces and explicit context dependencies only. A software component can be deployed independently and is subject to composition by third parties.

There are several researches [26] which provide the methods to do component identification.

(2) Actors Identification

To identify the actors, all extra elements who or what directly interact with the product or product-component should be concerned. The actors can be categorized into three classes. One is the *major* actor who gets the immediate services from the product, another is the *secondary* actor who keeps the normal operations of the product, and the third is the *supporting* actor who supports some functions to the product. Our model uses below questions to elicit the actors of each product or product-component.

(a) Who or what use the product/product-component?
(b) What roles do they play in the interaction?
(c) Who installs the product/product-component?
(d) Who or what starts and shuts down the product/product-component?
(e) Who maintains the product/product-component?
(f) What other systems interact with this product/product-component?
(g) Who or what gets and provides information to the product/product-component?

(3) Use Cases Identification

The functions provided to the actors are always applied with information store and retrieval. The state transition of a product could be triggered by an actor interaction. The product operations activated by external events generate output reports or interact with outside systems. Our model refers the cooperation relationships among all these supporting actors to design below questions that help developers to elicit functional requirements of each product or product-component.

(a) What functions does an actor want from the product/product-component?
(b) Does the product/product-component store and retrieve information?
(c) Which actor triggers the store and retrieval of this product/product-component?
(d) What happens when the product/product-component changes state?
(e) Is any actor(s) notified when the product/product-component changes state?
(f) Do any external event(s) affect the product/product-component?
(g) Which actor notifies the product/product-component about those events?
(h) Does the product/product-component interact with any external system?
(i) Does the product/product-component generate any reports?

(4) The Project Glossary

Every business domain has its own language. The glossary provided a dictionary of specific business terms and definitions. In the project glossary, developers should

1 The modification of system objectives could violate the organizational needs.

record the preferred terms and list any synonyms under the definition.

Developers repeat the above steps till all functional requirements of the products and product-components are established. These components cooperated together to complete the functionalities of the software product. The requirements of communication interfaces among these components are necessary to be defined. Then, the interfaces of products and product-components are analyzed in conjunction with the communication concepts to derive interface requirements.

(5) Interface Requirements Identification

The last step is to identify the interfaces among products or product-components. Our model uses below questions to elicit the interfaces requirements for products or product-components one by one.

(a) What product(s)/product-component(s) could communicate with the product/product-component?
(b) What functions do they request respectively from this component?
(c) What communication formula is for each of these functions?
(d) What interfaces are summarized for these functions?

6. Conclusion

Although there are many researches and commercial tools used to solve requirement development problems, they can not give an exact solution still. The difficulties of software requirement development increase dramatically in the highly changeable environment. The importance of requirement development manifests clearly on the reasons of a failure project.

Our model elaborates requirements from the viewpoints of goal, use case, and scenario according to the practices of RD process area in CMMI Level3. The model realizes the requirements specification by questionnaires in accordance with requirement development processes provided in CMMI.

Our model collects and accumulates problem solutions on RD and refers the most popular process improvement approach CMMI. This model provides the ability to do the elicitation of customer requirements and the requirements establishment of product and product-component. It provides a high acceptance methodology in questionnaire referred to goal, use case, and scenario driven approaches.

We are currently studying the feasibility of this model by cooperating with some software companies. The future work is to extend this model to achieve the goals of "Analyze and Validate Requirement in RD-PA3", and then to implement a CASE tool to support the process improvement of an organization on RD-PA3.

The major work of the next year is to: (1) extend this model to achieve the goals of "Analyze and Validate Requirement in RD-PA3"; (2) implement a CASE tool to support the process improvement of an organization on RD-PA3; (3) implement the models to accomplish the goals of requirement management process area in CMMI Level2 (RM-PA2); (4) implement a CASE tool to support the process improvement of an organization on RM-PA2.

References

[1] R. Darimont, A. Lamsweerde, "Formal Refinement Patterns for Goal-Driven Requirement Elaboration," Proc. 4th ACM Symposium on the Foundations of Software Engineering, pp. 179-190, Oct., 1996.

[2] M.Paulk et al., "Key Practices of Capability Maturity Model for Software," Version 1.1, Technical Report CMU/SEI-93-TR-25, Software Eng. Inst., Carnegie Mellon Univ., Pittsburgh, Penn.,1993http://www.sei.cmu.edu/pub/documents/93.reports/pdf/tr24.93.pdf.

[3] CMMI Product Team, "Capability Maturity Model, Integration (CMMI)," Version 1.1, Technical Report CMU/SEI-2002-TR-011, Software Eng. Inst.,Carnegie Mellon Univ.,Pittsburgh,Penn.,2002, http://www.sei.cmu.edu/pub/documents/02.reports/pdf/02tr011.pdf.

[4]CMMI Product Team, "CMMISM for Systems Engineering/Software Engineering/Integrated Product and Process Development/Supplier Sourcing, Version 1.1, Staged Representation (CMMI-SE/SW/IPPD/SS, V1.1, Staged)," Technical Report CMU/SEI-2002-TR-012, Software Eng. Inst., Carnegie Mellon Univ., 2002, http://www.sei.cmu.edu/pub/documents/02.reports/pdf/02tr012.pdf.

[5]Capability Maturity Model Integration (CMMI) from the Carnegie Mellon University Software Engineering Institute, http://www.sei.cmu.edu/cmmi/.

[6] L. V. Manzoni and R.T.Price, "Identify Extensions Required by RUP to Comply with CMM Levels 2 and 3," IEEE Trans. Software Eng., vol. 29, no. 2, pp. 181-192, Feb. 2003.

[7] R. Ceron, J.C. Duenas, E. Serrano, and R. Capilla, "A Meta-model for Requirements Engineering in System Family Context for Software Process Improvement Using CMMI," Software Process Improvement: 6th International Conference, June, 2005.

[8] A.V. Lansweerde, L.Willemet, "Inferring Declarative Requirements Specifications from Operational Scenarios," *IEEE Trans. on Soft. Eng*. Vol. 24, NO. 12, pp. 1089-1114, Dec. 1998.

[9] W. Heaven and A.Finkelstein, "UML profile to support requirements engineering with KAOS," IEE proceedings, Software, Vol. 151, NO. 1, pp. 10-27, Feb. 2004.

[10] A.M. Hickey, A.M. Davis, "Requirement Elicitation and Elicitation Technique Selection: A Model for Two Knowledge-Intensive Software Development Processes," Proc. 36th Int. Conf. System Science, pp. 96-105, Jan. 2003.

[11] G.D. Penna, B. Intrigila, A.R. Laurenzi, and S.Orefice, "A XML Definition Language to Support Scenario-Based Requirements Engineering," Int. *Journal of Software Engineering and Knowledge Engineering (IJSEKE)*, vol. 13, no. 3, pp. 237-256, June, 2003.

[12] A.V. Lamsweerde, L. Willemet, "Inferring Declarative Requirements Specifications from Operational Scenarios," *IEEE Trans. on Soft. Eng*., vol. 24, no. 12, pp. 1089-1114, Dec. 1998.

[13] S. Uchitel, J. Kramer, J. Magee, "synthesis of Behavioral Models from Scenarios," *IEEE Trans. on Soft. Eng*.., vol. 29, no. 2, pp. 99-115, Feb. 2003.

[14] W.J. Lee, S.D. Cha, and Y.R. Kwon, "Integration and Analysis of Use Cases Using Modular Petri Nets in Requirements Engineering," IEEE Trans. Software Eng., vol. 24, no. 12, pp. 1115-1130, Dec. 1998.

[15] B. Ramesh, M. Jarke, "Toward Reference Models for Requirements Traceability," IEEE Trans. Software Eng., vol. 27, no. 1, pp. 58-92, Jan. 2001.

[16]Patterns and Software: Essential Concepts and Terminology, http://www.cmcrossroads.com/bradapp/docs/patterns-intro.html.

[17] H.Zhua, L. Jinb, D. Diaperc, G. Baid, "Software requirements validation via task analysis," The Journal of Systems and Software, vol. 61, pp. 145-169,2002.

[18] R.S. Pressman, Software Engineering-A Practitioner's Approach Sixth edition. McGRAW.Hill, 2005.

[19] A. van Lamsweerde, L.Willemet, "Inferring Declarative Requirements Specifications from Operational Scenarios," IEEE Trans. Software Eng., vol. 24, no. 12, pp. 1089-1114, Jan. 2001.

[20] E. Letier , A. van Lamsweerde, "Deriving operational software specifications from system goals," Proceedings of the tenth ACM

1 The modification of system objectives could violate the organizational needs.

SIGSOFT symposium on Foundations of software engineering, November 18-22, 2002, Charleston, South Carolina, USA
[21] B. Regnell , K. Kimbler, A. Wesslen "Improving the Use Case Driven Approach to Requirements Engineering," Proceedings of the Second IEEE Inter. Symposium on Requirements Engineering, March, 1995, York, UK.
[22] Kexing Rui , Greg Butler, "Refactoring use case models: the metamodel", Proceedings of the twenty-sixth Australasian computer science conference on Conference in research and practice in information technology, p.301-308, February 01, 2003, Adelaide, Australia
[23]R. O. Briggs, P. Gruenbacher, "EasyWinWin: Managing Complexity in Requirements Negotiation with GSS," Proc. of the 35th Hawaii International Conference on System Sciences, 2002.
[24] Somerville,I., and P. Sawyer, Requirement Engineering, Wiley, 1997.
[25] C.Szyperski, Component Software: Beyond Object-Oriented Programming, second ed., Addison-Wesley, 2002.
[26] J. Cheesman, J. Daniels, UML Components: A Simple Process for Specifying Component-Based Software, first ed., Addison-Wesley, 2001.

1 The modification of system objectives could violate the organizational needs.