A MATURITY MODEL OF EVALUATING

REQUIREMENTS SPECIFICATION TECHNIQUES

A Thesis

by

YONGHEE SHIN

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

August 2003

Major Subject: Computer Science

A MATURITY MODEL OF EVALUATING

REQUIREMENTS SPECIFICATION TECHNIQUES

A Thesis

by

YONGHEE SHIN

Submitted to Texas A&M University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Approved as to style and content by:

Hoh In (Chair of Committee)	Jyh-Charn Liu (Member)
Thomas L. Rodgers (Member)	Valerie E. Taylor (Head of Department)

August 2003

Major Subject: Computer Science

ABSTRACT

A Maturity Model of Evaluating

Requirements Specification Techniques. (August 2003)

Yonghee Shin, B.S., Sookmyung Women's University

Chair of Advisory Committee: Dr. Hoh In

It is important to evaluate and understand the state-of-art technologies to position our research and invest our energy and resources in more effective ways. Unfortunately, no systematic approach has been introduced to evaluate the maturity of technologies except a few models such as Redwine/Riddle's model (Redwine and Riddle, 1985), which does not contain a significant concept, "goals" of technologies.

A new goal-oriented, technology maturity evaluation model has been proposed in this present study. The model aids to measure how a technology meets the goal of the technology along with a well-defined procedure. The model has applied to evaluate the maturity of the requirements specification technology as a case study. The results showed that this approach promoted effectiveness of measuring the technology maturity and understanding the state-of-art technology.

To my parents and beloved friends

ACKNOWLEDGMENTS

I would like to thank God for giving me this great opportunity learning how to conduct research. I would like to express my sincere gratitude to my advisor, Dr. Hoh In, for his insightful advice and patience. I would also like to thank my committee, Dr. Thomas Rodgers and Dr. Steve Liu, for their advice and encouragement. I give thanks to my family and my friends for their encouragement and prayers. My special appreciation goes out to Sungoh, Eunsun, and Sujung for their heartfelt advice. Lastly, I appreciate Raghav for sharing information during the progress and reviewing my thesis. Thank you all.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

CHAPTER I

INTRODUCTION

Since technology in the computer science field changes rapidly, it is vital to evaluate the state-of-art technologies with appropriate research method to utilize our energy and resources in preferred effective ways. Although surveys have been used to help understand the current status of technologies, only few of them used a systematic approach to evaluate the maturity of technologies. Therefore, when researchers try to evaluate a technology, it is quite difficult to know where to start and how to progress. Besides the evaluation results are often different among researchers. To avoid those matters, a systematic approach should guide finite steps of a procedure so that there is not much variance in the evaluation results among researchers. The approach should also lead to a precise evaluation in an efficient way. A precise evaluation means that the evaluation results reflect all the aspects of a technology properly. For this purpose, the evaluation method could identify characteristics of a technology and the relationships among them. In addition, the evaluation method could take into account the different levels of importance of the characteristics. An efficient method means a method which can reduce the overall time and efforts by using it. Considering the importance of technology evaluation, it is helpful to develop an evaluation model so that it can be

---

This thesis follows the style and format of *Automated Software Engineering*.

generally applied to all kinds of software technologies.

From this motivation, this present study has proposed a new goal-based technology maturity evaluation model based on Redwine/Riddle's model. The model adopted five maturity levels to measure how much a technology achieves its goal. The study also has proposed the way to estimate the maturity levels by implementing a well-defined procedure to gather data. Therefore, the proposed method is quantitative rather than qualitative. The procedure consists of several steps which will be explained in Chapter III. While applying the procedure repeatedly, more precise data can be collected by analyzing a technology into sub-technologies and grouping them according to their patterns. In this way, the model helps to analyze the current technology status in a systematic way and leads to a more exact evaluation. In addition, the model provides a simple and integrated perspective into a technology which consists of various sub-technologies.

In summary, this present study proposed a goal-based maturity model of software technologies and its supporting procedure, and showed the effectiveness by applying it to the requirements specification technology. The suggested goal-based systematic approach helps to measure the technology maturity more exactly.

The rest of the chapters consists of the following. Chapter II introduces related work including Redwine/Riddle's model. Chapter III describes the proposed goal-based, technology maturity evaluation method. Chapter IV describes a case study experience in which the method was applied to requirements specification. Chapter V concludes with a summary and future work. Appendix A summarizes the survey on requirements

specification to support the case study.

CHAPTER II

RELATED WORK

Redwine/Riddle's software technology maturity model (Redwine and Riddle, 1985) and Pohl's three dimensional approach (Pohl, 1993) are the representative related works of this study. Redwine/Riddle's model has applied in a simplified way in this present study in order to develop a new evaluation model of technology maturity. Pohl's three dimensional approach has provided concept on identifying the goal of requirements specification.

2.1 Redwine/Riddle's Software Technology Maturity Model

Redwine/Riddle suggested a maturity evaluation model of software technology (Redwine and Riddle, 1985). This paper defines six levels of maturity: basic research, concept formulation, development and extension, internal enhancement and exploration, external enhancement and exploration, and popularization. This paper also suggests transition points from one level to the next level. For example, if there are seminal papers on a technology, it means indicate that the maturity level of the technology progressed from the basic research level to the concept formation level. If there are commercialized tools, it means that the maturity is in the popularization level. Redwine and Riddle (1985) also characterizes the critical factors which help broad use of a

technology, the inhibiting factors of the maturation process and the facilitating factors to distribute the technology broadly. Figure 1 represents the six levels of Redwine/Riddle's model.

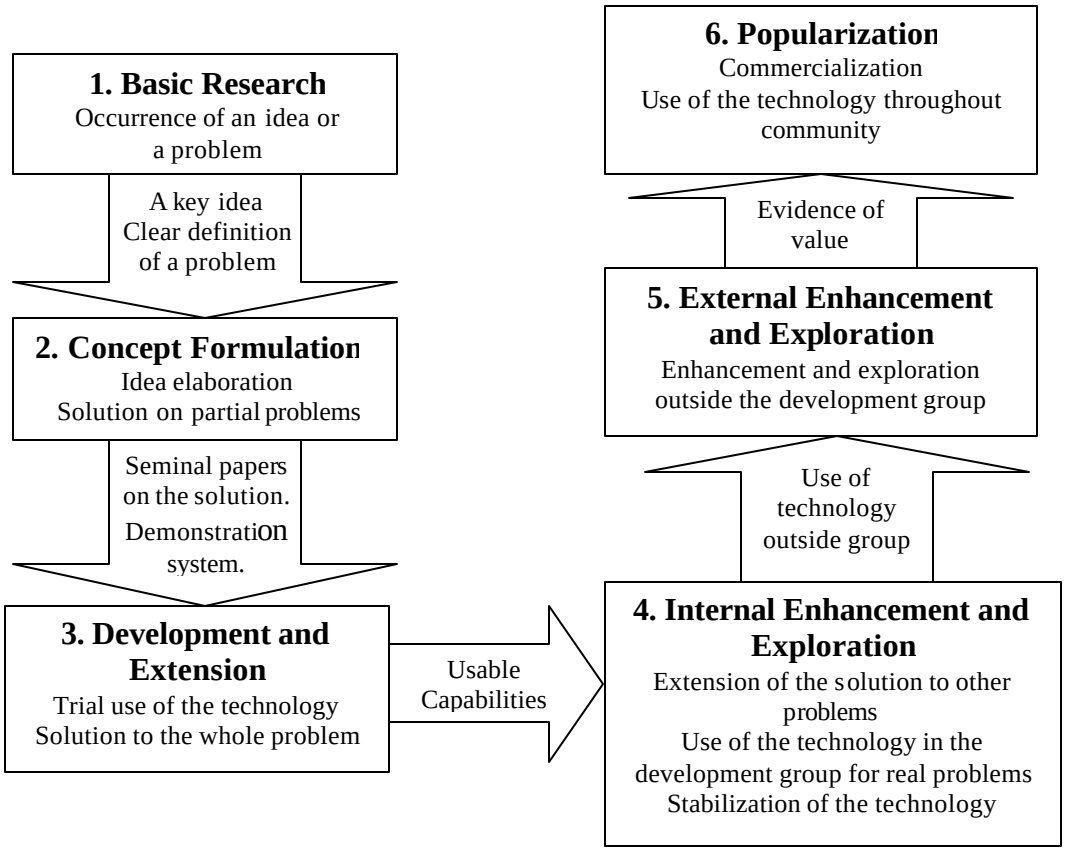

Figure 1. Six levels of Redwine/Riddle's software technology maturity

However, this model evaluates the maturity based on how broadly a technology is used. Even though a technology is broadly used, it could be used as an alternative method because there are no proper tools supporting a desired goal. To supplement this weakness, the proposed approach in this present study identifies the goal of a technology first, and then applies the proposed procedure to measure how much a technology

matures in terms of the goal.

2.2 Pohl's Three Dimensional Approach

Pohl suggested three dimensions of requirements engineering (Pohl, 1993): the specification dimension, the representation dimension and the agreement dimension. Specification dimension represents how much a specification is complete and how much understanding a specification gives of the system. Representation dimension represents how much a specification is formal. Agreement dimension represents how much a specification reflects a common view among stakeholders. Within the three dimensions, initial input is an opaque personal view using informal representation languages, and desired output is a complete formal agreed specification. The purpose of RE (Requirements Engineering) process is to lead from the initial input to the desired output. Figure 2 shows Pohl's RE framework.

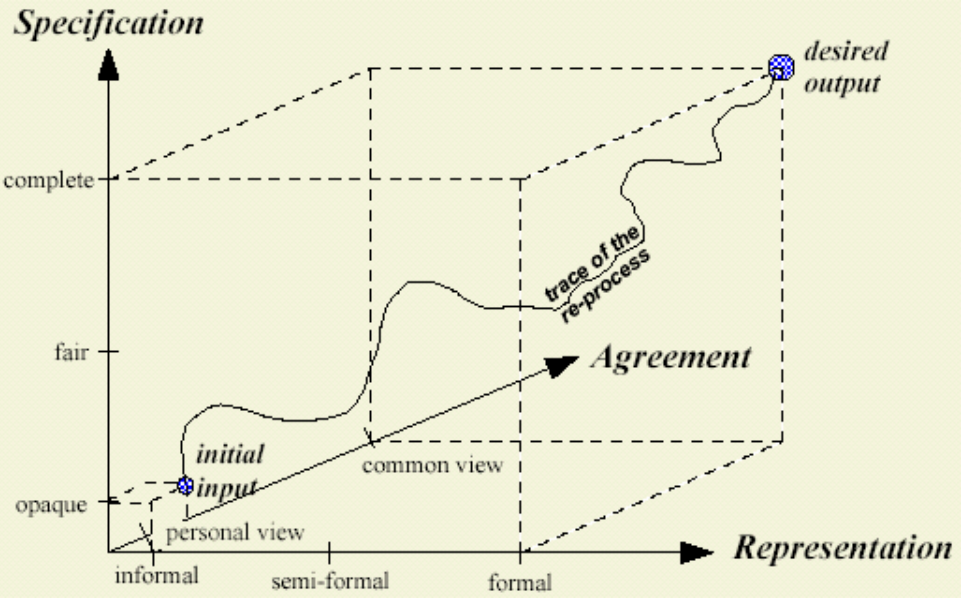

Figure 2. Pohl's requirements engineering frame work (Pohl, 1993)

This framework shows the importance of having goal in developing a technology, even though its purpose is not for evaluating technology maturity. The goal concept has applied to the proposed technology maturity evaluation method. In Chapter IV, the goal of requirements specification has identified and the proposed technology maturity evaluation method has applied to measure how much the goal was achieved by current technologies as a case study.

CHAPTER III

A GOAL-BASED TECHNOLOGY MATURITY EVALUATION METHOD

In this chapter, after discussing the importance of a new model, the new model and its benefits are described.

3.1 Need for a New Model

A goal is the purpose toward which an endeavor is directed (The American Heritage, 2000). Any organizations have goals. Any systems have goals. A goal makes our efforts concentrate on the desired purpose. As a natural result of this, a goal makes every action more clear. In order to accelerate the improvement of a technology in the right direction, the goal of the technology should be identified and its achievement should be able to be measured.

In software technology, it is very difficult to expect technology trends, because they are influenced by various factors such as change of business environment, change of other related technologies and invention of new technologies. However, the goal of a technology can be identified based on the users' needs regardless of these changes.

In order to measure the technology maturity, a systematic method should be offered. Even though Redwine/Riddle's model provides a way to measure the maturity in terms of broad usage, a new method is necessary to measure the maturity in terms of goal achievement.

3.2 Proposed New Model

This study proposes a technology maturity evaluation model to provide a solution to the need described in Section 3.1. Rather than measuring the maturity only in terms of how broadly a technology is used, this model measures the maturity in terms of how much a technology meets the goal. A technology consists of sub-technologies which satisfy different characteristics of a technology. If all of the sub-technologies are mature, it means that the technology has achieved a goal. If only some sub-technologies are mature, it means more research efforts on the immature sub-technologies are required. If the partial solutions are not mature, it means that research focus should be on unit technologies rather than integrated technologies. In this way, the goal-based technology maturity model follows an analytical approach to understand the problem.

Five levels of the goal-based technology maturity are defined as follows:

1. *Need*: A problem is identified and people recognize the necessity of a technology. Basic ideas are invented and published.
2. *Attempt*: Most of the partial solutions are developed and they are used only in the development group.
3. *Usable*: Most of the partial solutions are developed and they are used outside the development group.
4. *Useful*: Most of the partial solutions have commercial product quality and they are broadly used.
5. Indispensable: All of the partial solutions have commercial product quality and

they are broadly used. This means that the goal of the technology was achieved. Figure 3 summarizes the five levels of software technology maturity.

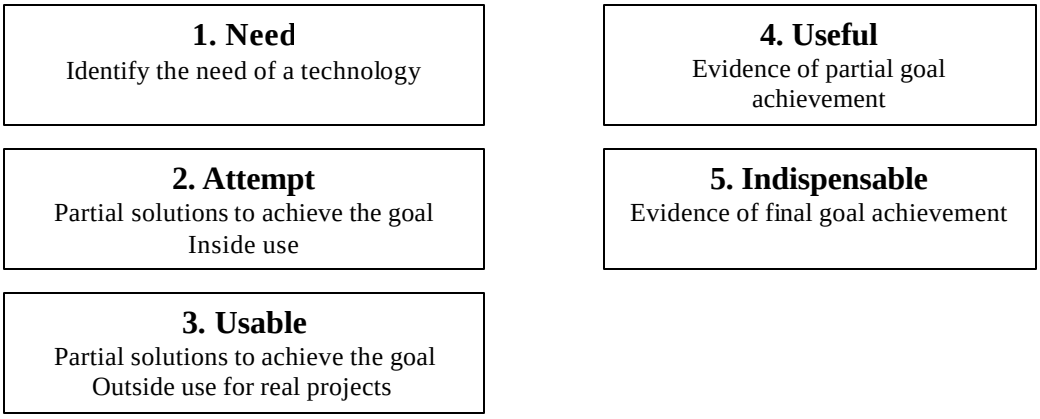

Figure 3. Five levels of software technology maturity

In order to decide a maturity level, this present study also proposes a well-defined procedure which should be followed before deciding the maturity level. Each partial solution is measured by Redwine/Riddle's model. Then, the overall maturity is calculated by using both of the measurement results and other data which are gathered while applying the procedure. Figure 4 shows the goal-based maturity evaluation procedure.

Step 1 and Step 2 can be vague at the first evaluation. However, in Step 3, more properties or property groups can be found. In this case, the overall steps should be repeated until all the properties and groups are found properly. The steps are explained as follows:

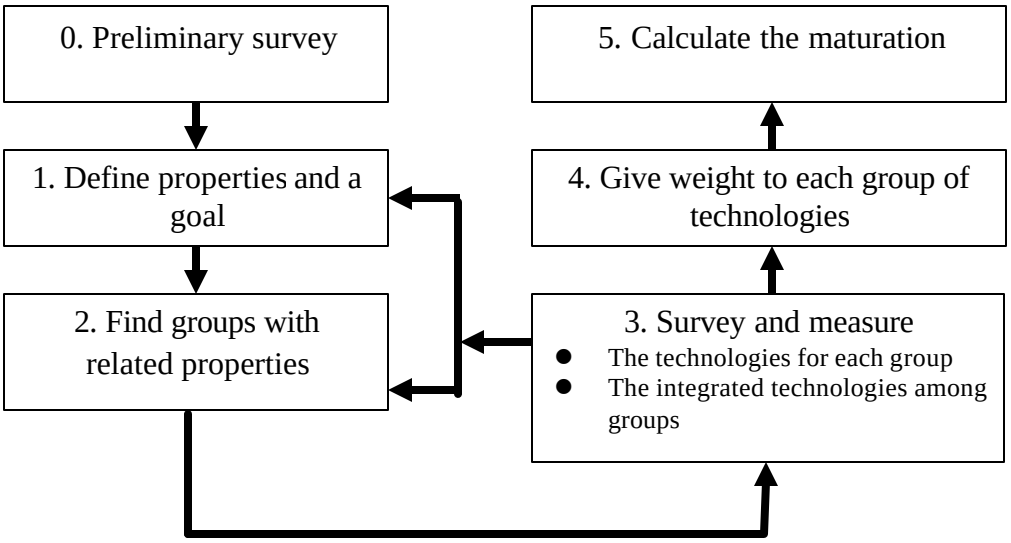

Figure 4. Procedure for technology maturity evaluation

Step 1. Define properties and a goal.

A *quality property* is a desired characteristic of a technology. Any complex technology has properties to be satisfied with the technology. For example, if our purpose is to make an inexpensive and powerful notebook, it should satisfy several properties such as low-power consumption, high resolution monitor, fast processing with light-weight materials and compatibility with other hardware interfaces. Initial properties can be found from a preliminary survey. After that, more properties can be found through iterative application of this procedure. A *goal* is defined as satisfying all of these properties at the same time.

Step 2. Find groups with related properties.

Some of the properties are related with each other very closely so that they can be solved by the same sub-technology. There are four kinds of relationships among properties.

- Strong-help relationship: A technology to satisfy a property can help to satisfy another property. These properties are in a helping relationship. If they are related closely, usually they are solved by the same technology. Therefore, these properties should be included in the same group.
- Strong-trade-off relationship: A technology to satisfy a property can be difficult to satisfy another property. These properties are in a trade-off relationship. Usually they are solved by different technologies respectively. Each technology can be integrated by an integration technology. This relationship requires the most difficult integration technology.
- Weak-help relationship: These properties can be solved by different technologies because their relationship is weak. Each technology can be integrated by an integration technology. However, the integration technology is not more difficult than that of strong-trade-off relationship, because the properties are in the helping relationship.
- Weak-trade-off relationship: These properties can be solved by different technologies because their relationship is weak. Each technology can be integrated by an integration technology. However, the integration technology is more difficult than that of weak-help relationships, because the properties are in the trade-off relationship.

To provide an easy explanation, some terminologies need to be defined here. *A unit technology* is a technology to satisfy the properties in a group. *An integrated technology* is an integration of unit technologies from different property groups. *A*

sub-technology is used to represent a unit technology or an integrated technology for the purpose of simplicity. *A goal technology* is an integrated technology which satisfies all the properties. Figure 5 represents the relationship among these technologies.

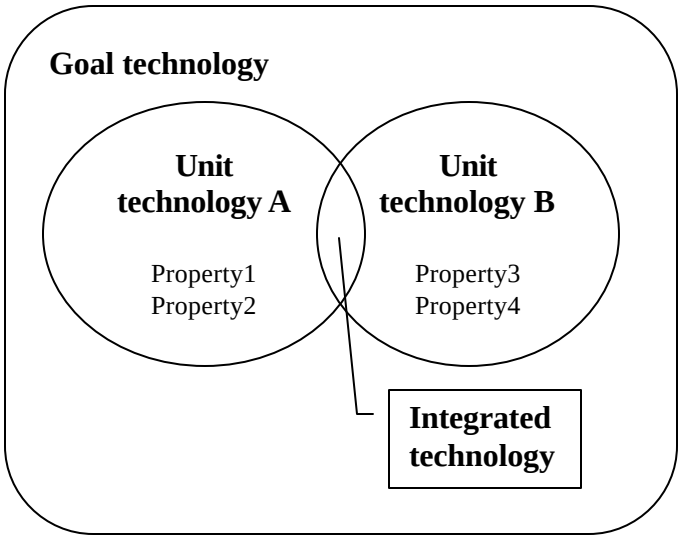

Figure 5. Category of technologies

Step 3. Survey and measure the maturity for each group.

In order to measure the technology maturity, unit technologies are measured first by the simplified Redwine/Riddle's model. Then, integrated technologies are measured. In the simplified Redwine/Riddle's model, the basic research level is combined into the concept formulation level, and the development and extension level is combined into the internal exploration level, since it does not have much effect on the purpose of this study and it makes the formula to calculate the maturity level simpler. Figure 6 represents the four levels of the simplified Redwine/Riddle's software maturity model.

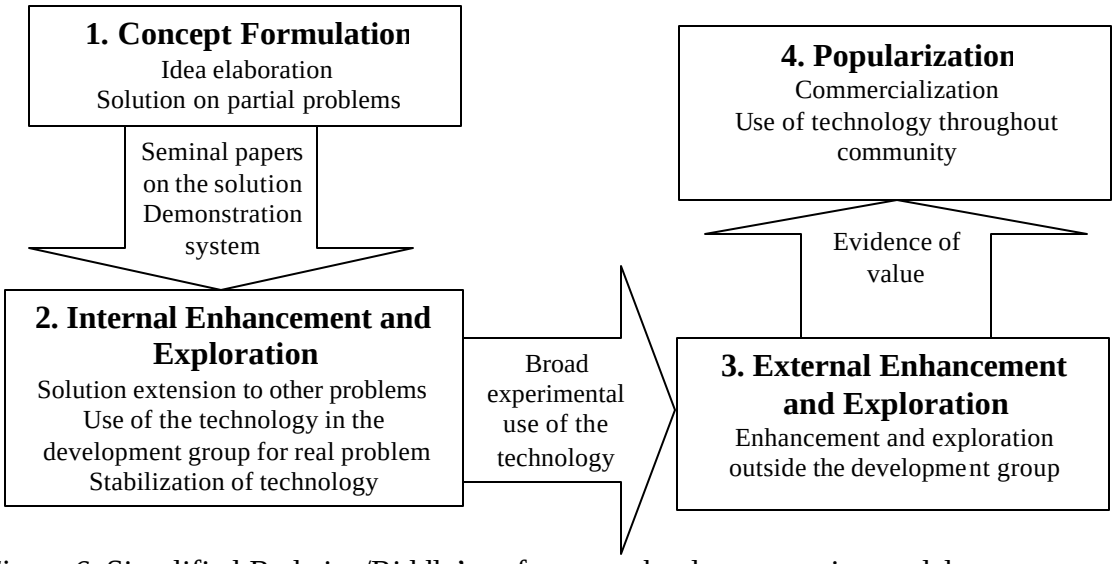

Figure 6. Simplified Redwine/Riddle's software technology maturity model

Step 4. Give weight to each group of technologies.

Since all the technologies are not important in the same degree, different weights should be assigned to different sub-technologies according to their importance. The weights range from 1 to 10. The importance of sub-technologies can be determined based on the information from Step 3.

If two property groups are in a strong-trade-off relationship, their integrated technology is very important and the weight is 8 to 10. If property groups are in a weak-trade-off relationship, their integrated technology is less important and the weight is 4 to 7 depending on the weights of their unit technologies. If their unit technologies have heavy weights, the integrated technology also has a heavy weight. If two unit technologies are in a weak-help relationship, their integrated technology is much less important and the weight is 1 to 3 depending on the weights of their unit technologies.

This is because their integration is relatively easy when they are in a helping relationship. Table 1 shows the criteria for assigning weights.

However, the current method of assigning weights needs to be improved, because the importance of a technology is decided quite subjectively. A more systematic and objective method is necessary for this.

Table 1. Criteria for assigning weights

Category of technology	**Importance of technology**	**Weight**
Unit technology	Very important	8-10
Unit technology	Medium important	4-7
Unit technology	Less important	1-3
Integrated technology	Strong-trade-off relationship	8-10
Integrated technology	Weak-trade-off relationship	4-7
Integrated technology	Weak-help relationship	1-3

Step 5. Calculate the maturity.

The overall maturity is the average of the weighted maturity of all the sub-technologies. To calculate the overall maturity, the following formula is used.

$$\text{Goal-based maturity level} = \frac{\sum_{i=1}^{n} MaturationLevel(i) * Weight(i)}{\sum_{i=1}^{n} 4 * Weight(i)} * 4$$

In this formula, *n* is the number of sub-technologies and 4 indicates the number of maturity levels.

Figure 7 shows an example of a technology maturity evaluation using this formula.

Weight 4 is given for the unit technology A in the popularization level. Weight 8 is given for the other unit technology B in the external exploration level. The integrated technology for both A and B has weight 4 and it is in the internal exploration level. By applying the above formula, the overall maturity of the goal technology is in the usable level. This result indicates that even though the unit technology A is in the popularization level, much more efforts are necessary to improve both the unit technology B and the integrated technology to achieve the final goal.

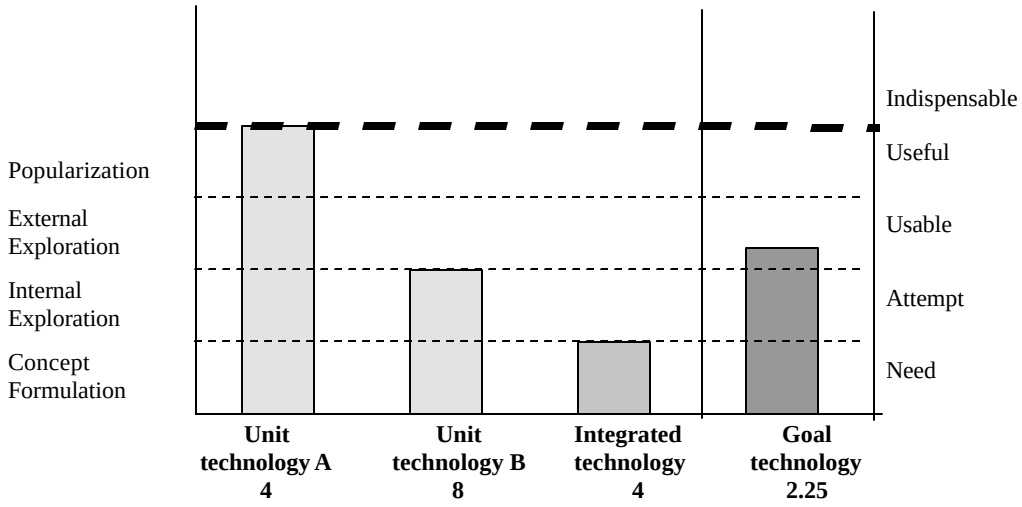

Figure 7. An example of goal-based technology maturity evaluation

3.3 Benefit of the New Model

For the comparison of the proposed model with the existing maturity model, the benefits of maturity evaluation and the purpose of Redwine/Riddle's model is discussed first.

The benefits of maturity evaluation are as follows:

- Current status of a technology can be shared with the research community, especially among the beginners in the technology.
- It helps to find the future research area; if a technology is really necessary and it is in the immature status, more research efforts on the immature technology are necessary.

The Redwine/Riddle's model is aimed at

- Identifying how long it usually takes for a technology to mature. Therefore, knowing the history of a technology from the basic research level is important.
- Identifying what the leverage points to speed up widespread usage.

On the contrary, the benefits of the proposed maturity evaluation approach include:

- Giving a simple and integrated perspective on the status of a technology which consists of many quality properties to achieve a goal.
- Leading to a more exact evaluation in a systematic way. During the application of Step 0 through Step 3, new quality properties or property groups can be found. In this case, by repeating the process, more exact evaluation results can be drawn.

CHAPTER IV

A CASE STUDY: APPLYING THE MODEL TO REQUIREMENTS SPECIFICATION

In this chapter, the proposed model in Chapter III is applied to the technology of requirements specification. After describing the motivation of this case study, the goal-based maturity evaluation model is applied to the requirements specification technology according to the proposed procedure in Chapter III.

4.1 Requirements Specification

The importance of requirements engineering has been emphasized for a long time. If the requirements are not correct, all the remaining processes including design, implementation and test are meaningless. According to Boehm, the correction of errors in the later stages of development can cost up to two hundred times as much as the correction in requirements analysis level (Boehm, 1981). Figure 8 represents the process of requirements engineering. Each activity can be accomplished iteratively as it is needed. The arrows in Figure 8 represent only the most important relationships between activities.

Among the requirements engineering activities, requirements specification is especially for describing requirements in a correct and easily understandable form. There have been many surveys on requirements engineering (Nuseibeh and Easterbrook, 2000,

Lamsweerde, 2000, Pohl, 1993, Zave, 1997). However, few of them are focused on requirements specification. Even though there are some surveys on requirements specification, most of them focus on a special aspect of requirements specification such as formal methods. There is no comprehensive survey or evaluation on requirements specification which includes all the aspects of formal, semi-formal and informal methods. Therefore, I chose requirements specification as a case study.

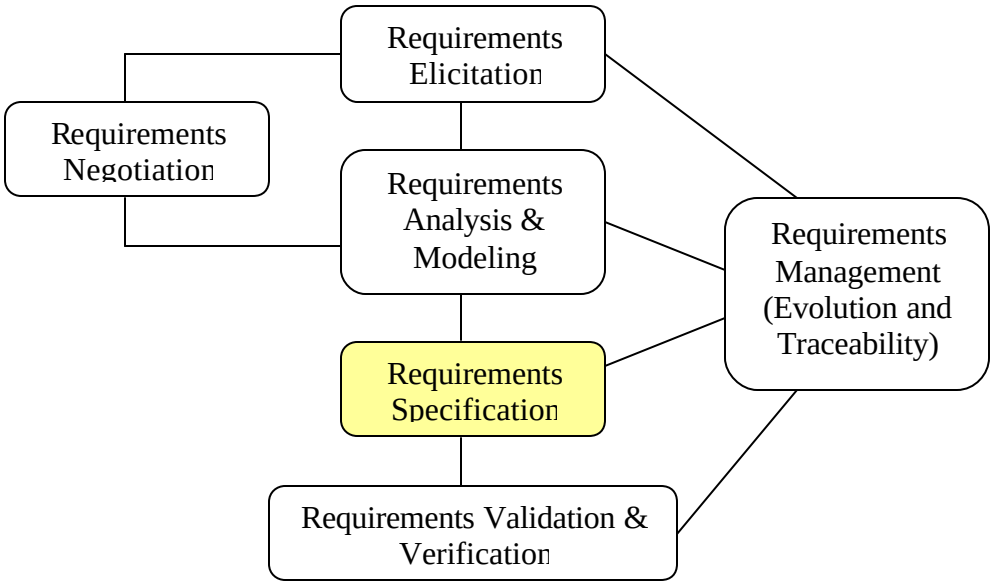

Figure 8. Process of requirements engineering

In most cases, natural languages are used for requirements specification. However, natural languages are prone to cause error. Even though there are rising concerns on formal specification methods, their application domains are narrow, and their notations are too difficult and too diverse. This means there is huge improving potential in

requirements specification technology. By analyzing the current technology status, this case study tried to find out the potential improvements.

4.2 Applying the Model

In order to evaluate the technology maturity in requirements specification, the proposed approach in Chapter III was used. The following steps are the result of applying the procedure after the preliminary survey.

Step 1. Define properties and a goal.

IEEE 830-1998 Recommended Practice for Software Requirements Specification defines good characteristics of requirements specification as correctness, unambiguousness, completeness, consistency, ranking for importance or stability, verifiability, modifiability and traceability. These can be properties of requirements specification. However, IEEE 830-1998 overlooked the importance of understandability. As formal methods become more popular, understandability becomes more important because formal notations are usually very difficult to read and understand. Therefore, in this study, understandability was added to the properties of requirements specification.

The following is a brief explanation of each property.

- Correctness: A requirement specification should describe what the software should meet according to the users' needs.
- Unambiguousness: A requirements specification should not allow multiple interpretations.

- Completeness: A requirements specification should include all significant requirements and all the outputs for all the inputs.
- Consistency: There should be no conflicts between requirements.
- Ranking of importance and stability: Requirements should be described with the indication of importance and stability.
- Verifiability: There should be finite feasible steps of process to check if a requirements specification satisfies all the desired quality properties.
- Modifiability: A requirements specification should be well structured and should not be redundant so that modified requirements do not result in any inconsistency.
- Traceability: Requirements should be traceable from or to documents of other requirements engineering activities.
- Understandability: Requirements specification should help easy communication between stakeholders.

The goal of requirements specification technology is to satisfy all these quality properties

Step 2. Find groups with related properties.

Even though the technologies to satisfy these properties can not be separated strictly, some related properties have high probability that can be achieved by the same technology. Therefore, the quality properties of requirements specification can be divided into three groups. Among these properties, correctness, unambiguousness, completeness, consistency and verifiability can be achieved by formal methods. Ranking importance and stability, modifiability and traceability can be achieved by requirements

management methods. Figure 9 represents the groups of quality properties in requirements specification technology.

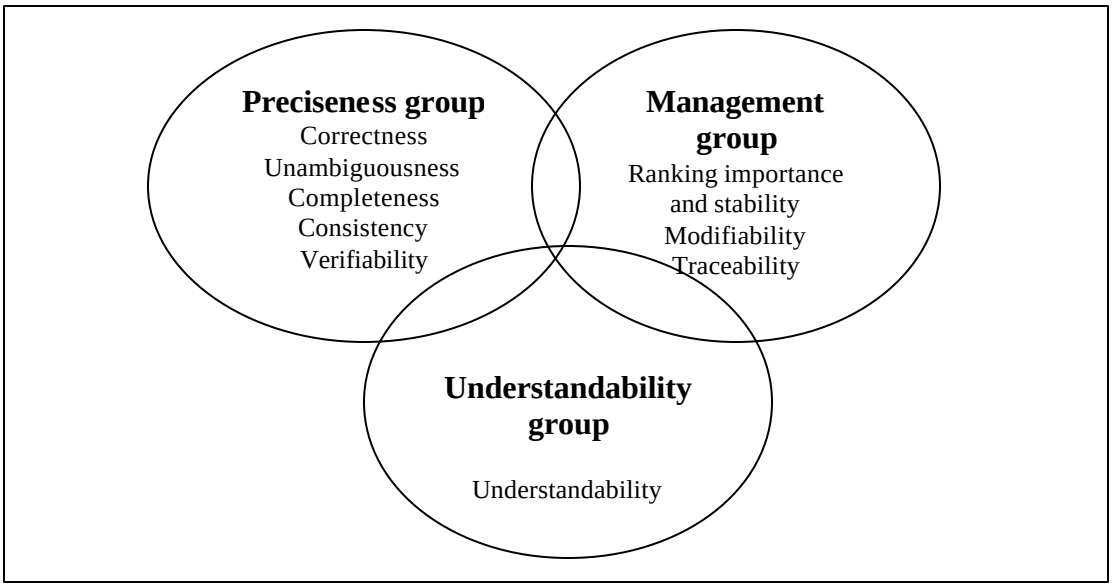

Figure 9. Initial groups of properties

Among these groups, it is important to note that preciseness and understandability are in a strong-trade-off relationship. The more precise, the less understandable, because preciseness is usually ensured by formal methods which are difficult for non-exports to understand. Therefore, it is important to include the integrated technology for both of these properties in the evaluation. This is because even though the two technologies are in the mature level separately, their combination could be immature. Figure 10 represents the trade-off relationship between preciseness and understandability.

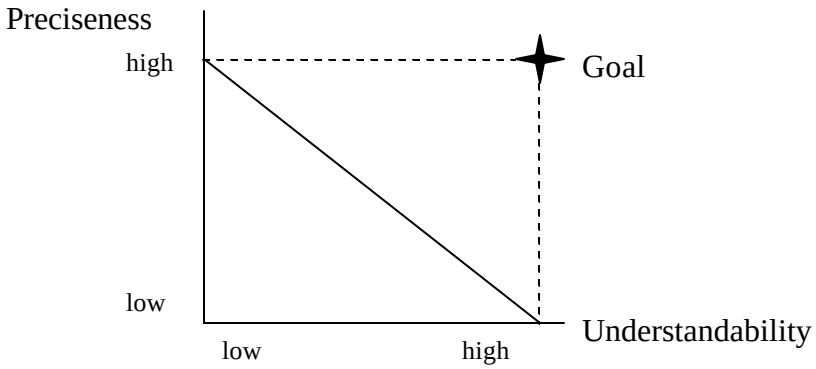

Figure 10. The relationship between preciseness and understandability

After the first iteration of the procedure, the prominent difference of maturity in the preciseness group was found. Verifiability technology was in the external exploration level and the other technologies for preciseness were in the popularization level. Therefore, the preciseness group can be divided into two groups: verifiability group and preciseness group. Due to this observation, another integrated technology between the verifiability group and the preciseness group was found. In addition, the survey result showed there are not much close relationships between the technologies for management and understandability or between the technologies for management and preciseness. Therefore, management group can be separated from the understandability group and the preciseness group. Figure 11 is the modified group diagram.

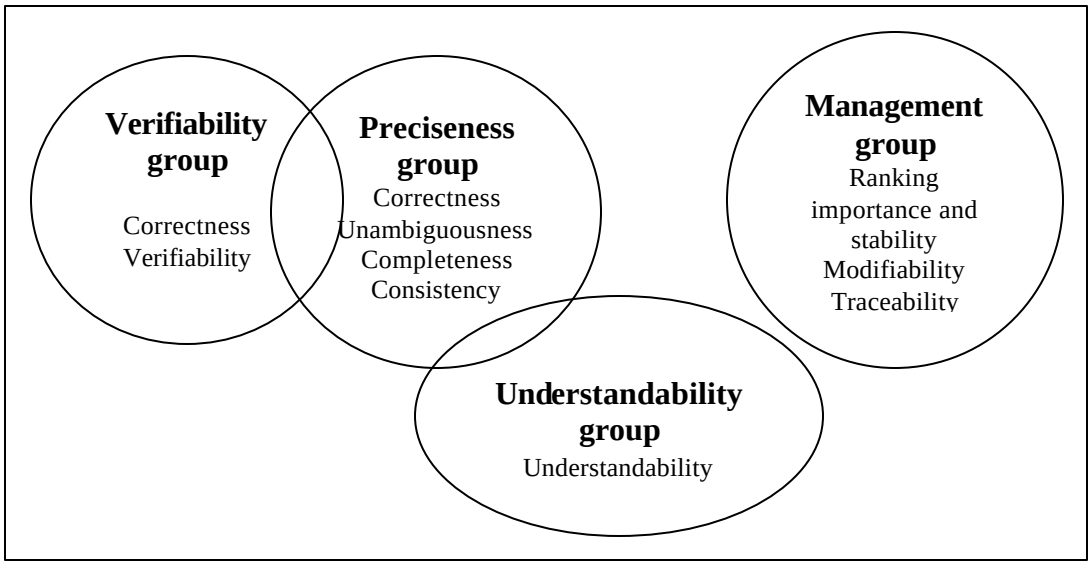

Figure 11. Modified groups of properties

*Step 3. Survey and measure the maturity for each group***.**

The results of maturity evaluation for sub-technologies are the following. Categories in each table are the categories of sub-technologies defined in Appendix A. Table 2 represents the result of technology maturity evaluation for preciseness technology.

Formal notations and supporting analysis tools are in the popularization level. However, verification technology is in the external exploration level. Therefore, the preciseness group was divided into the preciseness group and the verification group as described in Step 2.

Table 2. Evaluation of preciseness technology

Levels	**Category**	**Year**	**Description**
Concept Formulation			N/A (Not Available)
Internal Exploration	A.1.1	1970s	Z, VDM and SCR formal languages were developed.
	A.3.1	1980	The first model checker EMC was introduced. SPIN model checker began to be developed.
	A.3.2	1990	PVS theorem prover was introduced.
	A.1.2	2002	There was an approach to translate from UML to B formal specification.
External Exploration	A.1.1	1993	ISO draft of VDM was released.
	A.1.1	1997	SCR* Toolset was developed and used in industry.
	A.3.2	1993-present	PVS has been used in many industry and academy as a prototype.
	A.3.1	1990-present	After free releasing of SPIN model checker in 1991, it has been used widely. There is an international workshop for SPIN from 1995.
Popularization	A.1.1	2002-present	Z was standardized as ISO/IEC 13568 and it is supported by many commercial tools.
	A.1.1	1996-present	VDM was standardized as ISO/IEC 13817-1 and it is supported by many commercial tools.
	A.1.1	1984-present	SDL was standardized from 1984 and updated continuously until 1999 by ITU (International Telecommunication Union). It is supported by many commercial tools.

Table 3 represents the result of technology maturity evaluation for understandability technology. It shows that the technology for understandability is in the popularization level.

Table 3. Evaluation of understandability technology

Levels	**Category**	**Year**	**Description**
Concept Formulation			N/A
Internal Exploration	A.2.1	1994	UML began to be developed.
	A.2.1	1996	Attempto Controlled language was developed.
	A.2.2	1996	REVIEW system was developed to generate English text specification from DFD.
	A.2.2	1999	There was an attempt to visualize Z specification by using UML and other diagrams.
External Exploration			N/A
Popularization	A.2.1	1997-present	UML 1.0 was released and many commercial tools are available.
	A.2.1	N/A-present	DFD and E-R diagram are widely used.
	A.2.1	N/A-present	Natural language has already been the most widely used specification method.

Table 4 represents the result of technology maturity evaluation for management technology. It shows that the technology for management is in the popularization level.

Table 4. Evaluation of management technology

Levels	**Category**	**Year**	**Description**
Concept Formulation	A.4.1	1994	Identified pre-RS requirements problem.
Internal Exploration	A.4.1	1999	Developed a reference model for requirements traceability.
External Exploration			N/A
Popularization	A.4.1	N/A-present	There are many customized tools to support requirements traceability.

Table 5 represents technology maturity evaluation result of the integrated technology for preciseness and understandability. It shows that it is in the internal exploration level.

Table 5. Evaluation of the integrated technology for preciseness and understandability

Levels	**Category**	**Year**	**Description**
Concept Formulation	A.5.2	1998	Significantly many papers related with formal approach of UML were published.
Internal Exploration	A.5.2		Some projects using OCL (Object Constraint Language), which allows more formal semantic to UML, began.
	A.5.1	1994	ViewPoint framework was developed.
	A.5.1	1998	TRADE framework was developed.
	A.5.2	1990s-present	Active research on integration of UML and formal methods or on adding formal semantics to UML are being accomplished.
External Exploration			
Popularization			

Table 6 represents the technology maturity evaluation result of the integrated technology for preciseness and verifiability. It shows that it is in the internal exploration level.

Table 6. Evaluation of the integrated technology for preciseness and verifiability

Levels	**Category**	**Year**	**Description**
Concept Formulation			N/A
Internal Exploration	A.6.1	1996-present	There are many experimental translation technologies from a non-verifiable formal specification to the verifiable specification by using model checking or theorem proving
External Exploration			
Popularization			

Step 4. Give weight to each group of technologies.

Table 7 represents the initial weight assignment for the unit technologies and the integrated technologies according to their importance.

Table 7. Initial weight of technology groups

Technology group	**Weight**
Preciseness	10
Understandability	10
Management	5
Preciseness + Understandability	10
Preciseness + Management	3
Management + Understandability	3
Preciseness + Management + Understandability	10

As mentioned above, preciseness and understandability are in a strong-trade-off relationship. Therefore, the weight is 10. Management is important in the view of requirements engineering. However, it is less important than preciseness and understandability in the view of requirements specification. Management and preciseness are in a weak-help relationship because traceability improves preciseness and understandability. Management and understandability are also in a weak-help relationship. Therefore, their integrated technologies have lower weights than their unit technologies because it is relatively easy to integrate the technologies in a helping relationship.

After the first iteration of the procedure, a new unit technology and an integrated

technology were found as explained in Step 2. Table 8 is the modified weights for the modified groups.

Table 8. Modified weights of technology groups

Technology group	Weight
Preciseness	10
Understandability	10
Management	5
Verifiability	8
Preciseness + Understandability	10
Preciseness + Verifiability	8

However, the weight decision is so subject that others except the author can assign different weights. Therefore, more detailed criteria to assign weights are necessary in the future.

Step 5. Calculate the maturity.

Table 9 is a summary of the evaluation for the unit technologies and the integrated technologies.

Table 9. Summary of the evaluation

Unit technologies	Maturity levels	Weight
Preciseness	4	10
Understandability	4	10
Management	4	5
Verifiability	3	8
Preciseness + Understandability	2	10
Preciseness + Verification	2	8

Based on this result, the overall maturity can be calculated according to the formula defined in Section 3.2.

$$\text{Goal-based maturity level} = \frac{\sum_{i=1}^{n} MaturationLevel(i) * Weight(i)}{\sum_{i=1}^{n} 4 * Weight(i)} * 4$$

$$= \frac{(4*10)+(4*10)+(4*5)+(3*8)+(2*10)+(2*8)}{(4*10)+(4*10)+(4*5)+(4*8)+(4*10)+(4*8)} * 4 = 3.13$$

Therefore, the overall requirements specification technology is in the useful level. It indicates that even though technologies for preciseness, understandability and traceability are in the mature status, more efforts should be focused on improving their integrated technologies to achieve the goal.

4.3 Evaluation of the Model

In the Section 3.3, the claimed benefits of the proposed approach were the following:

- The model gives a simple and integrated perspective on technology status.
- The evaluation procedure leads to a more exact evaluation in a systematic way.

From the result of the case study in Chapter IV, this claim was proved to be correct. At first, requirements specification technology has nine quality properties and the sub-technologies to satisfy those quality properties are in the different maturity levels respectively. In this case, a method to summarize the overall maturity status is necessary. By applying the goal-based technology maturity evaluation method, the overall maturity

status was calculated and the result was that the requirements specification technology was in the useful level.

Secondly, during the evaluation process in the case study, verification technology was identified as a new important unit technology. The integrated technology between verification and the other quality properties was also identified. By measuring these technologies separately from other sub-technologies, it was possible to calculate the overall maturity more exactly because those technologies were in the different maturity levels.

CHAPTER V

CONCLUSION

This study has developed a goal-based, software technology evaluation model based on Redwine/Riddle's model, and has applied it to the requirements specification technology as a case study. The goal concept promotes a simple and integrated perspective on the technology maturity status. The repeated application of the proposed procedure leads to an improved analytic measurement and helps to obtain a more exact result. By the case study of requirements specification, the usefulness of the proposed model has been proved.

The contribution of this study can be summarized as follows. At first, this study suggests a new technology maturity model to promote an integrated view and exact evaluation. Secondly, the case study suggests the future research focus of requirements specification. Requirements specification technology is in the useful level. In order for it to be mature, the future research needs to focus on the three technologies: the verification technology, the integrated technology of preciseness and understandability, and the integrated technology of preciseness and verification.

Even though this study shows the practicability of the approach, an enhancement is needed in the future. The current model uses a rather subjective method to decide the weights of technologies. For this purpose, an enhanced systematic approach is required. For example, formal methods are not always necessary, because they are only helpful in

complex systems or safety critical application domains. Therefore, the percentage of the coverage of a technology could be used in order to precisely estimate the weights in Step 4.

REFERENCES

Agerholm, S. 1996. Translating specifications in VDM-SL to PVS. In *Proceedings of the 9th International Conference On Theorem Proving in Higher Order Logics*, Turku, Finland, August 1996, Published as *Lecture Notes in Computer Science*, vol. 1690, Springer-Verlag, 1999, pp.1-16.

The American Heritage®. 2000. Dictionary of the English Language, Houghton Mifflin, Boston, Massachusetts.

Bharadwaj, R. and Sims, S. 2000. Salsa: Combining constraint solvers with BDDs for automatic invariant checking. In *Proceedings of Tools and Algorithms for the Construction and Analysis of Systems* (TACAS 2000), Berlin, Germany. Published as *Lecture Notes in Computer Science*, vol. 1785, Springer, pp 378-394.

Boehm, B. W. 1981. *Software Engineering Economics*. Prentice-Hall, New York.

Bruel, J. M. and France, R.B. 1998. Transforming UML models to formal specifications. In *Proceedings of OOPSLA'98 Workshop on Formalizing UML. Why? How?*, Vancouver, BC, Canada, http://www.db.informatik.uni-bremen.de/umlbib/conf/OOPSLA98UML.html

Clarke, E. M. and Wing, J. M. 1996. Formal methods: State of the art and future directions. *ACM Computing Surveys*, 28(4):626-643.

Fantechi, A., Gnesi, S., Ristori, G., Carenini, M., Vanocchi, M., and Moreschini, P. 1994. Assisting requirement formalization by means of natural language translation. *Formal Methods in System Design*, 4(3):243-263.

France, R., Evans, A., Lano, K., and Rumpe, B. 1997. The UML as a formal modeling notation. In *Proceedings of OOPSLA'97 Workshop on Object-oriented Behavioral Semantics*, Atlanta, Georgia, USA., pp. 75-81.

Fuchs, N. E., Schwertel, U., and Torge, S. 1999. Controlled natural language can replace first-order logic. In *Proceedings of the 14th IEEE International Conference on Automated Software Engineering*, Cocoa Beach, Florida, USA, pp. 295-298.

Gotel, O. C. Z. and Finkelstein, A. C. W. 1994. An analysis of the requirements traceability problem. In *Proceedings of the First International Conference on Requirements Engineering* (ICRE), Colorado Springs, Colorado, USA, pp. 94-101.

Heimdahl, M.P.E. and Czerny, B.J. 1996. Using PVS to analyze hierarchical state-based requirements for completeness and consistency. In *IEEE High-Assurance Systems Engineering Workshop* (HASE '96), Niagara Falls, Canada, pp.252-262.

Heitmeyer, C. L., Jeffords, R. D., and Labaw, B. G. 1996. Automated consistency checking of requirements specifications. *ACM Transactions on Software Engineering and Methodology* (TOSEM), 5(3):231-261.

Heitmeyer, C., Bull, A., Gasarch, C., and Labaw, B. 1998. SCR*: A Toolset for specifying and analyzing requirements. In *Proceedings of the 10th Annual IEEE Conference on Computer Assurance* (COMPASS '95), Gaithersburg, Maryland, USA, pp. 109-122.

Kim, S. K. and Carrington, D. 1999. Visualization of formal specification. *Sixth Asia Pacific Software Engineeering Conference* (APSEC), Takamatsu, Japan, pp. 102-109.

Lamsweerde, A. V. 2000. Requirements engineering in the year 00: A research perspective. In *Proceedings of the 2000 International Conference on Software Engineering* (ICSE'2000), Limerick, Ireland , pp. 5-19.

Levy, N., Marcano, R., and Souquieres, J. 2002. From requirements to formal specification using UML and B. *International Conference on Computer Systems and Technologies*, CompSysTech'2002, Sofia, Bulgaria, http://www.prism.uvsq.fr/users/rafael/recherche.html.

Liu, S. 1992. A user-friendly formal requirements specification method. In *Proceedings of ACM 30th Annual Southeast Regional Conference*, Raleigh, North Carolina, USA, pp. 211-218.

Nuseibeh, B., Kramer, J, and Finkelstein, A. 1994. A framework for expressing the relationships between multiple views in requirements specification. *IEEE Transactions on Software Engineering*, 20(10): 760-773.

Nuseibeh, B. and Easterbrook, S. 2000. Requirements engineering: A roadmap. *The Future of Software Engineering, Special Issue 22nd International Conference on Software Engineering ACM-IEEE,* Limerick, Ireland, pp. 37-46.

Pohl, K. 1993. The three dimensions of requirements engineering. In *Proceedings of* the *Fifth International Conference on Advanced Information Systems Engineering*, Paris, France, pp.275-292.

Punshon, J. M., Tremblay, J. P., Sorenson, P. G., and Findeisen, P. S. 1997. From formal specifications to natural language: A case study. In *Proceedings of the 1997 International Conference on Automated Software Engineering (ASE '97)*, Lake

Tahoe, CA, USA, pp. 309-310.

Ramesh, B. and Jarke, M. 2001. Towards reference models for requirements traceability. *IEEE Transactions on Software Engineering*, 27(1):58-93.

Redwine, S. T. and Riddle, W. E. 1985. Software technology maturation. In *Proceedings of the 8th International Conference on Software Engineering*, London, UK, pp. 189-200.

Rolland, C. and Proix, C. 1992. A natural language approach for requirements engineering. In *Proceedings of the Fourth Advanced Information Systems Engineering*, Manchester, UK, Published as *Lecture Notes in Computer Science*, vol. 593, Springer-Verlag, New York, pp. 257-277.

Salek, A., Sorenson, P. G., Tremblay, J. P., and Punshon, J. M. 1994. The REVIEW system: From formal specifications to natural language. In *Proceedings of the First International Conference on Requirements Engineering*, Colorado Springs, Colorado, USA, pp. 220-229.

Schwitter, R. and Fuchs, N. E. 1996. Attempto - From specifications in controlled natural language towards executable specifications. In *Proceedings of the GIEMISA Workshop*, Tutzing, Germany, http://xxx.arxiv.cornell.edu/abs/cmp-lg/9603004.

Schwitter, R. 2002. English as a formal specification language. In *Proceedings of the 13th International Workshop on Database and Expert Systems Applications* (DEXA'02), Aix-en-Provence, France, pp. 193-197. Published as *Lecture Notes in Computer Science*, vol. 2453, Springer.

SDL (Specification and Description Language). 2003. *SDL Forum Society*, http://www.sdl-forum.org.

SE (Software Engineering) Tools Taxonomy – Requirements traceability tools. 2002. *International Council on Systems Engineering*, http://www.incose.org/tools/tooltax/reqtrace_tools.html.

VDM (The Vienna Development Method). 2000. *Center for software reliability*, University of Newcastle upon Tyne, UK, http://www.csr.ncl.ac.uk/vdm.

Wieringa, R. and Dubois, E. 1998. Integrating semi-formal and formal software specification techniques. *Information Systems*, 23(3/4): 159-178.

Zave, P. 1997. Classification of research efforts in requirements engineering. *ACM Computing Survey*, 29(4):315-321.

Z notation. 2003. *Z user group*, http://www.zuser.org/z.

APPENDIX A

SURVEY ON REQUIREMENTS SPECIFICATION TECHNOLOGY

Instead of listing all the technologies in detail, this appendix shows some representative technologies for each category.

A.1 Technology for Preciseness

As informal methods do not have enough semantics to verify the correctness in an automatic way, most efforts on preciseness technology are for formal methods.

A.1.1 Creating and improving specification languages

There are numerous formal specification languages. The following are representative ones. Each specification method has different strength and is used for different purpose. Some languages have their own verification methods. However, some languages rely on well known verification methods or tools by translating them to a proper language (Agerholm, 1996). These verification methods are introduced in Appendix A.3.

- Z: Z is a state-based specification language using schema notation. It was developed in 1970's and standardized as ISO/IEC 13568 in 2002. It is supported by many commercial tools (Z notation, 2003).

- VDM: VDM is a state-based specification language. It was developed in 1970's and standardized as ISO/IEC 13568 in 1996. It is supported by many commercial tools (VDM, 2000).
- SDL: SDL is a specification language for real-time systems. It was originally used for telecommunication area, but now it is used for much wider application domains. It was introduced in 1980 and standardized by ITU (International Telecommunication Union) in 1988. The latest standard version was released in 2000. It is supported by many commercial tools (SDL, 2003).
- SCR or SAL (SCR Abstract Language): SCR is a state-based specification language using tabular notation. It was introduced in 1970s. Its GUI toolset, SCR* has tools such as editor, consistency checker and model checker. Model checking is achieved by automatic translation of SCR into Promela which is a specification language for SPIN model checker (Heitmeyer et al., 1996, Heitmeyer et al., 1998). It has been used experimentally in industry and academy. SCR*'s improved version, Salsa provides consistency checking and also provides the combined verification ability of model checking and theorem proving (Bharadwaj and Sims, 2000).

A.1.2 Translating or paraphrasing an informal specification to a formal specification

- There are approaches to translate UML, a well-known informal modeling language into formal specification such as B (Levy et al., 2002).

A.1.3 Integrating formal and informal languages

This approach will be considered in Appendix A.5.

A.2 Technology for Understandability

As informal methods such as natural languages or diagrams usually give high understandability, most efforts on understandability technology are for formal methods.

A.2.1 Creating and improving specification languages

- Controlled natural language: This approach tries to improve understandability by using natural language while preserving preciseness by using limited grammar and vocabulary to give a formal reasoning ability. Attempto (Schwitter and Fuchs, 1996, Fuchs et al., 1999) and PENG (Schwitter, 2002) are the examples of controlled natural languages. Even though they use natural languages, they can be translated to other formal specifications for verification because of its well-formedness. However, controlled natural language editors usually require users' interaction to resolve ambiguous grammar.
- Visual notations: Some specification languages or modeling languages use visual notations such as diagrams, tables or graphs. These include DFD, E-R

diagram and UML (Unified Modeling Language).

A.2.2 Translating or paraphrasing a formal specification to an informal specification

- There were several approaches to paraphrase formal specification into natural language. REVIEW system generates an English text specification from a DFD specification or an object model (Salek et al., 1994, Punshon et al., 1997). Similar systems are GIST, ARIES and GETS system (Salek et al., 1994). There was research on translating or paraphrasing from formal proofs or relational calculus expressions to natural languages since 1970s (Rolland and Proix, 1992).
- Kim and Carrington (1999) tried to visualize Z specification by using diagrams. They used UML to represent static aspects of a system and used contract box to represent dynamic aspects of it.

A.2.3 Integrating formal and informal languages

This approach will be considered in Appendix A.5

A.3 Technology for Verification

A.3.1 Creating and improving model checking methods

- Model checking: Model checking is a method to check automatically whether a

model satisfies the desired properties of a system or not. It can be used to check the correctness of requirements specification. A model is described in a finite-state machine and the properties are described in logic formula. The first model checker was introduced in 1981. SVM and SPIN are well-known model checking tools (Clarke and Wing, 1996).

A.3.2 Creating and improving theorem proving methods

- Automatic theorem proving: Automatic theorem proving is a method to prove automatically whether a system satisfies the desired properties of a system or not. It can be used to check the correctness of requirements specification. Both of the system and properties are described in logic formula. PVS and STeP are well- known theorem proving tools.

A.4 Technology for Management

A.4.1 Improving traceability

- Gotel et al. identified the problem of pre-RS traceability (Gotel and Finkelstein, 1994). This concept influenced much to the later development of tools supporting traceability. Pre-RS traceability is the traceability from a requirements origin to a requirements specification. Post-RS traceability is the traceability from a requirements specification to the documents of later process

which reference the requirements specification.

- Reference model for traceability: There are a lot of technologies and tools to support traceability. However, their concerns and scope are different. Ramesh and Jarke (2001) proposed a reference model for requirements traceability which deals with comprehensive concerns and scopes in a well-defined framework.
- There are many commercialized tools for requirements traceability (SE Tools Taxonomy, 2002).

A.5 Integrated Technology for Preciseness and Understandability

A.5.1 Integrating several notations or methods into a framework

There are technologies to integrate several notations or methods into a framework. This approach enables developers to use their familiar specification notations by supporting multiple notations in a framework.

- TRADE (Toolkit for Requirements and Design Engineering) (Wieringa and Dubois, 1998): TRADE is a framework which includes several semi-formal specification techniques and formal methods. In this framework, most specifications are described in informal and semi-formal languages. Formal notations are used only for the most complex part of a system in which requirements cannot be reasoned without them. Specifications are traceable by

traceability links between informal and formal specifications.

- ViewPoint: ViewPoint is a framework which supports multiple specification methods and tools (Nuseibeh et al., 1994).

A.5.2 Adding formal semantics to existing informal notations

- pUML(Precise UML) (France et al., 1997, Bruel and France, 1998): UML is a well-known semi-formal specification language. Because of its wide use and easiness, many researches are trying to overcome the difficulty in understanding formal specifications by adding formal semantics to UML. Even though UML is widely used, it has a possibility to result in different understanding of specification among different stakeholders because of its lack of precise semantics. In addition, tools can check only syntax instead of semantics. To overcome this weakness, Precise UML project is trying to develop a formal reference manual to give precise description of core UML components and to provide inference rules for analyzing properties.
- Liu (1992) tried to combine DeMarco data flow diagram with VDM(Vienna Development Method). With the use of diagrams, the comprehensibility of a specification is improved and with the use of formal method, three kinds of consistency analysis were possible. The three consistency analyses include structural consistency, condition consistency and semantic consistency.

A.6 Integrated Technology for Preciseness and Verifiability

A.6.1 Translating a non-verifiable specification to a verifiable specification

- Attempto controlled language can be translated into first-order predicate logic for verification (Schwitter and Fuchs, 1996, Fuchs et al., 1999). They can be verified by theorem provers for first-order predicate logic. NL2ACTL is a system to translate from controlled natural language into ACTL (Action Computation Tree Logic) (Fantechi et al., 1994).
- Heimdahl and Czerny experimented to use PVS theorem prover for RSML specification. For this, they made a tool to generate theory and proof obligations from RSML specification (Heimdahl and Czerny, 1996).
- Agerholm experimented to translate VDM-SL to PVS (Agerholm, 1996).

A.7 Summary of Technology Categories

- Technology for preciseness

  A.1.1 Creating and improving specification languages

  A.1.2 Translating an informal specification to a formal specification

  A.1.3 Integrating formal and informal languages
- Technology for understandability

  A.2.1 Creating and improving specification languages

  A.2.2 Translating or paraphrasing a formal specification to an informal

specification

A.2.3 Integrating formal and informal languages

- Technology for verifiability

A.3.1 Creating and improving model checking methods

A.3.2 Creating and improving theorem proving methods

- Technology for management

A.4.1 Improving traceability

- Integrated technology for preciseness and understandability

A.5.1 Integrating several notations or methods into one framework

A.5.2 Adding formal semantics to existing informal notations

- Integrated technology for preciseness and verifiability

A.6.1 Translating a non-verifiable specification to a verifiable specification

VITA

Yonghee Shin was born in Korea in November, 1968. After receiving a Bachelor of Science degree in computer science at Sookmyung Women's University in March 1992, she worked for Daewoo Telecommunication Ltd. and Samsung SDS Ltd. in Korea for eight years. Following her desire for higher education, she pursued a Master of Science degree in computer science at Texas A&M University from August 2001 to August 2003, specializing in software engineering.

Her permanent address is 32-412, Siyoung Apartment, Sungsan-dong, Mapo-gu, Seoul, Korea.