Master Thesis
Software Engineering
Thesis No: MSE-2005:02
January 2005

An Empirical Study On Requirements Engineering Core Practices

Uday B. Goud
Kiran P

School of Engineering
Blekinge Institute of Technology
PO Box 520
SE-372 25 Ronneby
SWEDEN

This thesis is submitted to Department of Software Engineering and Computer Science at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Software Engineering. This thesis is equivalent to 20 weeks of full time studies for two students.

Contact Information

Authors:

Uday B. Goud
E-Mail: uday_goud2000@yahoo.com

Kiran P
E-Mail: kiranp_77@yahoo.co.uk

University Advisor:

Dr. Mikael Svahnberg
School of Engineering

Blekinge Institute of Technology
PO Box 520
SE 271 25 Ronneby
SWEDEN

Internet: www.tek.bth.se
Phone : +46 457 38 50 00
Fax : +46 457 271 25

ACKNOWLEDGEMENT

Firstly, we would like to thank our advisor Dr. Mikael Svahnberg, School of Engineering, Blekinge Institute of Technology. His constructive suggestions, directions and advice during the course of this thesis is very much appreciated.

We would like to express our appreciation to the companies that participated in this study. Without whose support, this thesis would not be feasible. We are thankful to all the interviewees for having invested their valuable time and effort for this study. We owe acknowledgment, to the fellow students who have rendered their help in contacting officials from these organizations.

Finally, we would like to express gratitude to our parents for their constant support and motivation.

Without the support from all these people this thesis would not be of the quality it is today. Thank you!

ABSTRACT

Requirements engineering (RE) is the primary task (process) that is done when agreed upon to develop a software product. The success of the software product is gauged on its ability to meet the intended needs of the stakeholders. There is abundant literature emphasizing the significance of RE and its influence on the entire software project, apart from its importance as the first step for a successful development endeavor. There are several established methodologies that are acknowledged to support the RE process and assist in creating a reliable structure of creating software. Despite the availability of such techniques and solutions, it was observed that umpteen number of software product failures are attributed to unsatisfactory RE practices.

In this thesis, we have conducted a study with six organizations to emphasize the gap between the *state of the art* and the state of the practice, and consequently identify the factors that hinder the industrial community to implement state of the art RE. As a result of this empirical research we have found that to a great extent, state of the art practices are unpopular, more specifically in small organizations. Interestingly the majority of the problems associated with RE are associated to non technical issues.

Keywords: Requirements Engineering, Requirements Elicitation, Requirements management, Core Practices

CONTENTS

Chapter 1: Introduction

Chapter 2: RE Core Practices

Chapter 3: Related Research

Chapter 4: Research Methodology

Chapter 5: Analysis of the Study

Chapter 6: Conclusion and further research

"RE is at the borderline between the informal and the formal"

- Dr. Manfred Broy

Chapter 1

INTRODUCTION

Introduction

Creating software is inherently a complex procedure (Duggan & Thachenkary, 2003), and the requirements specification phase is deemed to be the most complicated and important stage in this process (Reubenstein & Waters, 1991). It is acknowledged that the errors caused in the Requirements engineering (RE) phase if left unearthed will eventually lead to expensive effort in the later stages of software development (Reubenstein & Waters, 1991) (Sawyer & Kotonya., 1999) (Wiegers, 1999:260). The primary measure of success of a software application is the extent to which it meets the expectations of the user and its intended purpose (Nuseibeh & Easterbrook, 2000**)**. Software Requirements Engineering is the process of identifying what (specifications) is to be implemented into the software system, and analytically evaluating and refining those specifications. The goal of this practice is to discover the needs of the users, and then establish a conceptual basis for the process of building a software system (Herlea, 1996) by understanding the specific problem in its appropriate context. At this juncture most of the project goals and objectives are established (Humphrey, 2002).

RE as a whole is more of an administrative and communication driven effort than a technical effort, where efficient user - developer interaction (Wiegers, 1999:31) (Duggan & Thachenkary, 2003) is considered to be a vital aspect contributing to success. Individual, social and organizational factors have a prominent influence on the set of these RE activities (Kotonya & Sommerville, 1998). The consequences of bad requirements range from delayed delivery, over budget to poor quality software.

Frederick Brooks (1987) has illustrated a persuasive statement regarding the significance of requirements in one of his articles, "No silver bullet: Essence and accidents of software engineering":

"The hardest single part of building a software system is deciding what to build. No other part of the work so cripples the resulting system if done wrong. No other part is more difficult to rectify later."

- (Brooks, 1987):

1.1 Motivation for the Thesis

The discipline of software engineering holds existence for over four decades now (Mahoney, 2004). In this contemporary world, software application are being used in almost every segment of the humanity to facilitate efficient processing. Software artifacts being logically malleable have created boundless possibilities, and have revolutionized the way things were done in the past.

" IT is the closest thing we have to a universal tool. You can do almost anything with it."

- (Rogerson , 1998)

However, imperfection in software systems are found to be on a rise and research has also reveled that a majority of the users (or *stakeholders)* find that the software system is not up to their expectations. Furthermore it has been acknowledged by many authors that deficient requirements could be the major cause for project failure (Herlea, 2000) (Hofmann & Lehner, 2001). A Software System evolves from the requirements of the customers /stakeholders. The Requirements specification forms a foundation for the forthcoming phases of the software development; the requirements describe what the customer expects from the system, and how the system is expected to function. Therefore inconsistency and ambiguity in the system requirements hold a major accountability when the software products turn out to be unsuccessful.

In spite of the constant research being done in this area and various strategies that have been prescribed to address inadequacy in the requirements engineering process, even today the contemporary software engineers face the same situation. The issue, "m*ost of the software product failures are attributed to unsatisfactory requirements engineering process*" has being echoed over and again in various articles, publications and text books, since more than a decade now (Hofmann & Lehner, 2001) (Duggan & Thachenkary, 2003).

1.2 Objectives of the Thesis

The aim of this study is to investigate the extent to which the requirements engineering practices in the industry map with the existing generic requirements engineering principles/theory from the literature. In this thesis we shall attempt to give answers to these research questions:

1. To what extent do the software engineers in the industry implement the principles of requirements engineering in contrast to the theoretical practices?

2. What are the constraints the software engineers encounter when it comes to implementing the RE practices?

1.3 Contribution and Outline of the Thesis

The main contribution of this thesis is to empirically evaluate the gap between the state of the art RE and the contemporary state of the practice in the software industry. In this thesis we have investigated RE practices of six organizations and presented an analysis on the current state of practice. We have also highlighted the constraints faced by the engineers in implementing certain established methodologies.

The first part of the thesis i.e. chapter 2 introduces the reader to core RE practices, chapter 3 summaries other research works which are similar to this thesis. The research methodology implemented in this thesis is introduced in the chapter 4, which is followed by an analysis of this study in chapter 5. Finally in chapter 6 we have presented our conclusions along with suggestions for future research.

"I shall reconsider human knowledge by starting from the fact that we can know more than we can tell"

- Michael Polanyi

Chapter **2**

REQUIREMENTS ENGINEERING PROCESS

The goal of this chapter is to introduce Requirements Engineering (RE) practices which are further studied in the forthcoming chapters in this thesis. Hence, we give a brief introduction pertaining to the requirements engineering practices.

2.1 Introduction

One of the first phases of any software development process is RE phase. It is a complex problem solving process involving customers and many decisions (Aurum & Wohlin, 2003). There are various phases in the RE process (Fig 1), each phase consisting of several activities. Management of such processes demands appropriate procedures and tools. The quality of the software product relies largely on the quality of the development process employed in it (Aurum & Wohlin, 2003). In this chapter we have given short illustrations of all the activities involved in each phase of the RE process. Later in chapter 5, we map these practices to that of the practices in industry based on the conducted survey.

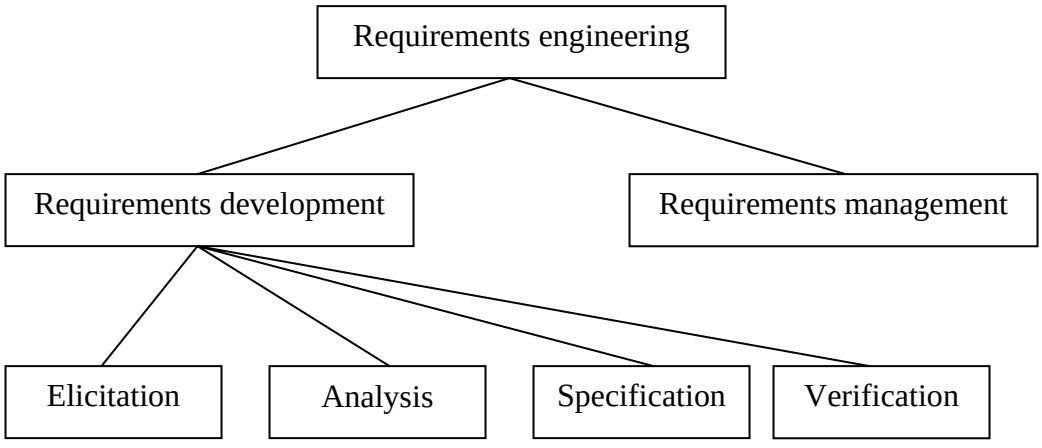


Fig 1: The above figure illustrates the course of activities encircling the phases of RE, adopted from (Wiegers, 2000).

The below are phases in the RE process, which are discussed from section 2.3 to 2.10.

- Requirements elicitation.
- Requirements analysis and negotiation.
- Requirements specification.
- Requirements validation.
- Requirements management.

2.2 Users, Customers or Stakeholders - Defined

Throughout the thesis while discussing various phases and their following discussions, the terms users, customers and stakeholders are used interchangeably. Hence it is important to introduce these terms, before getting further.

Stakeholders are the people "*party*" who have interest or "*stake*" in the product and shall be affected by the project, but are not directly involved with the development. System stakeholders might be an assortment of people with diverse knowledge bases, ranging from technical engineers to simple librarians. Internal Stakeholder might be managers, employees or directors of the organization. External stakeholders are end–users or customers.

Kotonya and Sommerville (1998) have illustrated an eloquent definition:

"System stakeholders are the people or organizations who will be affected by the system and who have a direct or indirect influence on the system requirements."

2.3 Requirements Elicitation

Requirements elicitation (also called as requirements acquisition) is the process of identifying / discovering the needs of the customers. Typically even if the users have a lucid thought of how they want their system to be, they usually fail to articulate their requirements in practical terms, thus leading to unrealistic or impractical demands (Sommerville & Sawyer, 1997:9) (Kotonya & Sommerville, 1998). Therefore it is up to the requirements analyst to work in close proximity with end-users and then derive and rollout a strategy to envision their requirements. It requires thorough understanding of the domain knowledge, organizational awareness as well as detailed problem comprehension (Sommerville & Sawyer, 1997). There are various methods known to be effective for Elicitation such as, conducting interviews, scenarios, soft system methods, developing software prototypes and presenting questionnaires (Kotonya & Sommerville, 1998). However, a requirements analyst is not restricted to a particular approach, organizational processes, application type, available resources and individuals choice play a role in selecting a specific method (Kassel & Malloy, 2003).

2.3.1 Conducting Interviews and Presenting Questionnaires

Interviewing is the most conventional mode of requirements elicitation. During the course of this thesis, this approach was found to be practiced by a majority of organizations. The process involves intense communication between the system developers and the end users. The goal of these interview sessions is to extract the customer's intensions or requirements for having to require a computerized system. These interviews could be conducted through a closed questionnaire or open interviews which are more like a face-to-face discussion session (Kotonya & Sommerville, 1998).

2.3.2 Scenarios

Scenarios can be thought of as detailed portrayal of usage context so as to facilitate design decisions, or as a model of existing product which can be used to anchor discussion regarding various design aspects (Kotonya & Sommerville, 1998:64). They can be considered as notions used to illustrate a specific situation or a setup. Scenarios help to illustrate various interactions between the end-user and the system. They also facilitate in identifying various possible system interactions and the facilities which might be required (Kotonya & Sommerville, 1998). A graphical representation of the scenario may be developed, which would simulate a more detail representation of the inputs and outputs, apart from depicting a lucid flow of the interaction sessions. Scenario based elicitation techniques do not require significantly more effort, but would assist contextual understanding to the end-users as well as the requirements analysts.

2.3.3 Use-cases

Use cases have become a standard technique of requirements elicitation for many of the present-day software organizations (Woolridge, 1999). The objective of implementing use cases is to mimic all the tasks that the user shall achieve with the system (Wiegers, 2003:133). Use-Cases illustrate a series of interactions between a system and an external actor. An actor might be a software application, a hardware device or a person that communicates with the system to yield a functional goal (Wiegers, 2003:133). Use cases provide a detailed structure for problem solving, and assist in experimenting to explore multiple solutions for a specific problem. Use case is a very practical modeling and analysis tool which can be used to identify the business processes and the requirement specifications necessary to support theses processes (Woolridge, 1999). According to specifications of UML (UML version 1.4, 2001), a use case is, "*the specification of a sequence of actions, including variants that a system (or other entity) can perform, interacting with actors of the system*".

2.3.4 Prototyping

Prototyping is a partial implementation of the final system, created with the intension of studying the problem and solution to the problem (Davis, 1992). Prototyping is a well

known method in the software engineering community as a reliable model for developing software (Carter et al, 2001). Prototyping is used to aid elicitation and validation of the system requirements (Kotonya & Sommerville, 1998). The rising cost of software and the software failure rate has driven the software organizations towards prototyping, in the attempt to create software that satisfies the customer's expectations (Davis, 1992). Two types of software prototyping are discussed in the literature, they are Throw-away prototyping and Evolutionary prototyping. (Davis, 1992) (Kotonya & Sommerville, 1998:73). Prototyping is known to improve quality of specifications apart from being cost effective (Andriole, 1994). During the course of our survey, we found that organizations realize the importance of prototyping. However, some did not put into practice.

"Throw-away" prototypes are usually discarded once the final system is developed. The aim of such a system is to facilitate the understanding of the system that is going to be built. This type of prototyping is used experimentally to recognize and reflect requirements that are ambiguous and those whose necessity is not well defined (Davis, 1992). The system analysts then generate the requirement specification based on the learnings from the prototype (Davis, 1992). This type of prototyping is well suited to evaluate small parts of complex problems, and is found to be cost-effective and improves specifications (Andriole, 1994).

Evolutionary prototyping is an approach where an initial prototype is developed, and then it is refined until the final stages of implementation. In this approach, well understood requirements are implemented, after a detailed analysis the developer redesigns the prototype based on what was learnt from the previous version (Davis, 1992). The process is repeated and tested until a distinct set of specifications are evolved. Features are added after each evolutionary iteration, and then transformed into an efficient implementation. This type of prototyping suits well when majority of the critical functions are well understood (Davis, 1992).

Pros and Cons of prototyping

Prototyping serves as resourceful grounds for evaluating systems requirements and its constraints, apart from facilitating constructive communication regarding the system (Carter et al, 2001). It also helps to uncover previously unknown or missing requirements (Carter et al, 2001). By designing prototypes developers have the opportunity to act upon and consider the lessons learned during the development (Carter et al, 2001). Finally the obvious benefit is that the misinterpretations between users and developers are clarified (Kotonya & Sommerville, 1998).

On the other hand, if the organization has no experience in implementing prototyping for development, the organization has to invest in training costs (Kotonya & Sommerville, 1998). Furthermore it might be possible that developing a prototype might require considerable amounts of time, ultimately causing delayed delivery of the product. However, if the delivered product is well appreciated, this might not be a problem (Kotonya & Sommerville, 1998).

2.3.5 Joint Application Development (JAD)

JAD was initially developed by IBM. The objective of JAD was to produce higher quality requirements specifications in lesser time (Jackson & Embley, 1996). The method addresses the short comings of conventional interviewing technique, and is known to consume less time (Rottmann). The method is carried out by a trained facilitator who shall be responsible to stimulate both the users and the developers in generating ideas about the system by way of effective communication, thus motivating to excavate all possible features to be implemented. JAD is also a recommended practice for nurturing user commitment, apart from accelerating the process of decision making (Davidson, 1999).

2.4 Understanding the Application Domain Knowledge

The modern day software applications are constantly increasing in size and complexity, (Sommerville & Sawyer, 1997) consequently increasing the density of in-house domain knowledge. It is very important that the requirements analysts should acquire this application domain knowledge, so that they can derive the tacit dimension of requirements (Sawyer & Kotonya, 1999). Users generally find it intricate to realize and communicate their requirements. It is often the task of the requirements analysts to discover the needs of the end-users, i.e. their vision for and expectations on the software application. The objective of gaining application domain knowledge is to perceive critical unstated assumptions and grasp implicit requirements (Wiegers, 2003:70).

Domain knowledge can be obtained by interacting with knowledgeable users at the stakeholders' organizations. On the other hand technical literature and manuals are always reliable sources of acquiring domain knowledge (Kotonya & Sommerville, 1998). Exploring organizational and business issues facilitates in understanding the end-users perception towards the problem and its constraints (Kotonya & Sommerville, 1998) (Sawyer & Kotonya., 1999). Larger organizations gain domain knowledge from their past experiences with similar projects.

2.5 User Involvement

Requirements elicitation is a mutual decision making activity comprising users, developers, and customers (Herlea, 1996). The most commonly referred factor on challenged projects is "lack of user input" (Standish Group, 1994). Wiegers suggests that the only way to avoid the expectation gap between the product that the customers expects to receive and the product that is built by the developers is to identify "user representatives" and involve them throughout the project (Wiegers, 2003:101). Efficient user-developer interaction has proven to be a crucial success factor (Wiegers, 2000). This strategy helps both the users and developers in communicating views, identifying and resolving conflicts, apart from sharing information that is necessary to efficiently complete the project (McKeen et al, 1994).

2.6 Requirements Analysis and Negotiation

Once the requirements for the Software are discovered, it is very important that they are organized in a coherent manner and assessed for conflicts and inconsistencies (Sommerville & Sawyer, 1997). The objective of this phase is categorizing the collected requirements into related subsets, prioritizing and refining them so that the stakeholders have an unambiguous understanding of the systems specifications (Wiegers, 2003:50). The process also encompasses transforming informal requirements (that are gathered from the stakeholders) to semiformal or formal requirements. Though some authors have illustrated analysis and negotiation as separate activities, in practice some amount of negotiation of requirements with the stakeholders is inevitably carried out during elicitation and analysis phases. This is due to the fact that obvious inconsistencies usually surface out during elicitation and analysis phase which are instantaneously negotiated (Kotonya & Sommerville, 1998). The result of analysis and negotiation is an agreed set of requirements (Sommerville & Sawyer, 1997).

2.6.1 Prioritizing the Requirements

As discussed before RE is a complex process, by and large it not feasible for the developers to implement every requirement requested by users into the software system. Most often software projects run on strict terms of budget and inevitably restricted resources (Wiegers, 1999a). The need for requirements prioritization is well acknowledged in various literatures (Karlsson & Ryan, 1996) (Kotonya & Sommerville, 1998).

Not all requirements are equally important (Firesmith, 2004). Prioritization is done to determine the degree of importance of each requirement. The project manager has to balance the requirements against the constraints of budget, schedule and available resources (Wiegers, 1999c). The strategy is to drop or differentiate lower priority requirements against the ones with higher priority, based on analysis of cost vs. value for implementation. Hence it is a way to deal with building software to provide the highest value at the best price (Wiegers, 2003:248). It is also done by characterizing the requirements on the scale of importance and urgency (Wiegers, 2003:250). This is usually derived from the outcome of careful analysis of opinions from developers and the stakeholders. Another method called “Pair wise importance technique” is used where requirements are compared in pairs to determine their relative importance (Karlsson, 1996).

Prioritization aids as an obvious means of choosing an optimized set of requirements to be implemented, apart from helping the developers to gauge the level of user satisfaction before the system is released (Karlsson, 1996).

2.7 Requirements Specifications

The objective of requirements specification is to present an overview of the applications ability as well as its proposed structure. A requirements specification includes functional requirements, data requirements, quality requirements as well as constraints on the

system. It is important that the requirements specifications are cross-referenced based on interdependencies with other requirements (i.e. traceability) (Gotel & Finkelstein, 1994). According to (Kotonya & Sommerville, 1998), formal methods are known to be more effective for specifying requirements, since natural language specifications often lead to ambiguity and misinterpretation. Nevertheless natural language specifications remains the most popular and practical approach towards requirements specification, basing on the simplicity of usage and understandability by all readers (Wiegers, 2003:165). However, no specific view of requirements provides a comprehensive understanding, therefore a mixture of both textual and visual representation provides a more complete picture of the required system (Wiegers, 2003:193). According to Wiegers, pictures communicate information more efficiently than text, and helps linking the gap between language and vocabulary. In this context requirement modeling serves an able instrument in specifying requirements. Wiegers also suggest the practice of use-case approach for specifying requirements (Wiegers, 2003).

2.7.1 Requirements Modeling

Requirements modeling is a logical representation for depicting and visualizing the complex nature of the multi-faceted interactions between subsets of requirements specification (Barker, 2000). It is a process of illustrating the complexly fused specifications into a coherent and less abstract form.

"*The written word is a wonderful vehicle for communication, but it isn't necessarily the best way to represent the requirements for computer software"*

- (Pressman, 2003)

Modeling requirements can unearth inconsistencies and omitted requirements, apart from assisting in eliminating superfluous requirements (Wiegers, 2003:51). Data flow models, object models, flow charts, state diagrams or entity-relationship models are some of the techniques for modeling requirements (Wiegers, 2003:51) (Kotonya & Sommerville, 1998). Interpreting specifications from colossal text documents is usually complicated, modeling cuts down the cycle time required to build complex systems, by simpler and more explicit notations. Consequently reducing the costs caused of rework due to specification errors (Barker, 2000). Modeling facilitates re-engineering by giving a clear picture of the entire systems functionality in an easily understood form (Barker, 2000).

2.8 Requirements Validation

Requirements validation is the process of evaluating system requirements for correctness, completeness and accuracy. Usually validation is the final phase of the RE process (Kotonya & Sommerville, 1998). The objective of the process is to confirm that the developers have a satisfactory set of descriptions to continue with the development (Kotonya & Sommerville, 1998). Below is a short description of validation techniques

The input to (Fig.2) the requirements validation phase is a complete and unambiguous set of requirements document, which conforms to the organizational standards (Kotonya & Sommerville, 1998:89). The implicit knowledge such as terminology and practices of the

organization are important since the requirements usually are closely related to the organizational culture and structure. The process output (Fig.2) is a list of problems associated with the requirements document, and agreed set of actions discussed to resolve those problems (Kotonya & Sommerville, 1998).

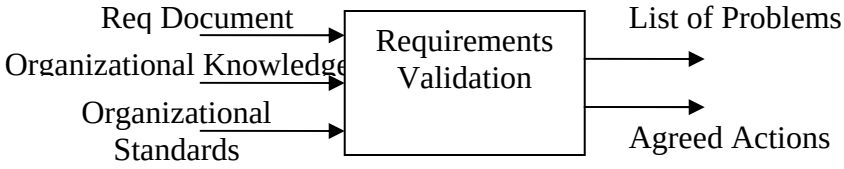


Fig 2: Requirements Validation (Kotonya & Sommerville, 1998: 89)

2.8.1 Reviewing system Requirements

Reviewing is a method for identifying ambiguous and unverifiable requirements. They can be conducted informally, by way of inviting colleagues to examine the deliverables, or the author draws comments or suggestions from the colleagues based on the description of the work product. Informal reviews are ad-hoc in nature where the feedback is usually unstructured. Formal reviews are usually carried by a review team who are responsible to judge the specification, based on whose evaluation a summary of critical issues and defects are identified. Inspection is acknowledged as the best approach for formal reviewing, yielding highest leverage on software quality (Wiegers, 2003:133).

2.8.2 Requirements Testing

The objective of requirements testing is to evaluate if the system meets the required functionality based on the requirements specification. Test plans that contain multiple test cases are written for the functional requirements based on the user requirements to reproduce the expected system behavior (Wiegers, 2003:273). Every requirement is comprehensively tested at least once, and some requirements are tested several times since some requirements represent multiple states in varying scenarios (Rosenberg et al, 1998). Wiegers suggests that writing functional black box test cases helps in understanding of how the system would behave under specific conditions (Wiegers, 2003:274).

2.9 Requirements Management

Requirements management is often controlling the evolution of complex requirement specifications, which due to their complexity are often subjected to many, sometimes conflicting changes (Leffingwell & Widrig, 2003). Such changes are particularly frequent at the requirement level entities. However, in practice the impact of changes in requirements is often severely underestimated**.** The foremost goals of the requirements management are (Wiegers, 1999:133):

- Managing changes to the requirements baseline.
- Ensuring that the project plans is updated reflecting the system requirements.
- Version control of individual requirements and the requirements documents.
- Organizing the relationships between requirements, and managing dependencies and interdependencies between individual requirements within various project counterparts (traceability of requirements).
- Tracking the status of requirements with respect to the baseline.

Consequently there is a need for a method to handle these activities throughout the software development; a process such as software configuration management supports requirements management to a great extent. "Effective change management demands a process for proposing changes and evaluating potential cost and impact on the project" (Wiegers, 2003:55). Tool support for requirements management has been discussed in section 2.9.3 in this chapter.

2.9.1 Configuration Management (CM)

It is crucial that the requirements are consistent and should reflect exactly what the customer wants (Kovitz, 1999). It is a typical phenomenon that requirements creep into the project even after the project requirements are baselined. This is due to the fact that change is almost certain in business processes, technologies and other factors as the project evolves (Kotonya & Sommerville, 1998). Consequently there is a need to maintain evolution of the requirements specification during the entire development life cycle. Accordingly, CM supports to aid maintain integrity of such changes to requirements throughout development, apart from maintaining the history of the development process. A range of commercial CM tools are available to assist automated configuration management.

2.9.2 Requirements Traceability

Several stakeholders such as project managers, analysts, designers, project sponsors and end-users are involved in development cycle. Consequently the needs of each of these stakeholders vary based on their goals and priorities, which ultimately becomes the root cause of problems concerning traceability of requirements (Ramesh & Jarke, 2001). CM tools assists in identifying the sources of requirements. Traceability facilitates understanding the evolution of software artifacts, apart from establishing the impact of changes in requirements specification (Kotonya & Sommerville, 1998:128). It also helps in understanding the underlying details of a specific design and implementation aspects of the software system (Kotonya & Sommerville, 1998:128).

"*The requirements traceability is the ability to describe and follow the life of a requirement, in both a forward and backward direction, i.e. from its origins, through its development and specification, to its subsequent deployment and use, and through periods of ongoing refinement and iteration in any of these phases.*" (Gotel & Finkelstein, 1994).

2.9.3 Tool Support

Managing the evolution and changes in requirements is too big a task to be handled manually, a tool can significantly lighten this task by managing many of the change management activities (Leffingwell & Widrig, 2003). A requirements management tool serves as an automated assistance as the development progresses (Wiegers, 1999b). Typical activities that are supported by a tool are (Wiegers, 1999b):

- Managing versions of individual requirements and documenting change history,
- Storing the attributes of individual requirements,
- Tracing the dependency of specific requirements on other system elements,
- Status tracking of each requirement during development,
- Control the accesses of information pertaining to requirements between individuals or groups of users,
- Identify related subset of requirements with similar attributes,
- Facilitates communication with stakeholders, by way of discussing and notifying requirements issues electronically.

However, without a well established process in place, any tool would turn out to be futile.

"*Requirements management tools can't do your requirements management for you, but the right one will support and simplify your existing process*". (Wiegers, 1999b).

The tools that support the Requirements Engineering Process can be divided in two groups (Kotonya and Sommerville, 1998):

- Tools that are used for modeling and validating system requirements. Such tools allow creating graphical or textual models of requirements and then checking these models for consistency and completeness.
- Tools that are used for managing requirements. These tools provide such facilities as requirements storage and retrieval, tracing of requirements and requirements change control and configuration management.

Summary

In this chapter we illustrate important and fundamental RE practices, i.e. Requirements Elicitation, Analysis and Negotiation, Specification, Validation and Management. The chapter covers brief accounts of activities encompassing these core routines. Based on these core practices we map (in chapter 5) the knowledge we have gained from experienced software professionals from the industry.

"If I have seen further it is by standing on the shoulders of giants"

- Sir Isaac Newton

Chapter **3**

RELATED RESEARCH

In the previous chapter we have seen the overview of the core Requirements Engineering practices. The goal of this chapter is to look a step further into research work specifically those similar to our current work. The chapter ends with a conclusive summary.

3.1 Review of Related Work

The related works are discussed in chronological order.

3.1.1 Earlier Works

One of the early works published in the field of survey on RE is Curtis et al, (1988). This is a field study conducted on multiple software development projects by interviewing the personnel involved in it. The research was much more focused on the early phases of the development process i.e., on the requirements and design phases. Initially through literature survey they concluded that Requirements and design decisions have greater influence on software productivity, quality and cost. So they laid their emphasis and conducted studies on how these decisions were made and treated in the development process and their impact on the downstream of the development process. The analysis of their interviews revealed three major factors – thin spread of application domain knowledge; fluctuating & conflicting requirements and communication and coordination breakdowns that adversely affect the project success. Finally they conclude that software development is not only a technical task but also should be viewed as a knowledge sharing (learning), communication and negotiation process. These processes should be given utmost importance in software development stream to build successful projects. However, the findings of this study are bound to large systems development.

In 1992 (Lubars et al, 1993) conducted a field study of ten organizations. The focus of their study was on how requirements issues (defining, interpreting, etc) are carried on in different projects, the projects were categorized into customer driven and market driven projects. The analysis of their survey revealed that requirements statements are lengthy and are defined inconsistently in the customer specific projects. How ever in these projects domain experts aid in interpreting the requirements. Customers were the major sources of initiating changes. On the other hand in the market driven projects the requirement statements are shorter. Due to less documentation and lack of domain

knowledge, interpreting requirements was a challenge. Here economic and market factors influence changes to the requirements. Meetings with the customer were still a dominant method for defining and clarifying requirements. Requirements were prototyped only in one third of the studied projects. In both the customer and market driven projects changes to requirements were handled by formal procedures and tools.

El Emam and Madhavji (1995) conducted a field study which is quite similar to (Lubars et al, 1993). The main purpose of their study was to make recommendations on improving RE processes by studying then current RE practices and impediments on them in various cases (information systems related). They studied 60 cases, a high number of cases when compared to above two studies. They found that technical and non-technical issues are equally important in RE, but non technical issues posed a greater threat on RE which is inline with findings of (Curtis et al, 1988). The study also revealed seven key issues that were problematic in RE practices. Much of these problems were related to decision making (planning), roles of personnel involved in RE and user involvement.

3.1.2 Recent Works

Another study by Nikula et al, (2000) focused on RE practices in small and medium sized companies. Their aim was to examine the level of technology transfer from research community to industry. The result of the survey disclosed very low level of technology transfer and also most of the companies they surveyed did not follow a defined RE process which indicates low level of formality in the RE practices. None of the companies participated in their survey used a requirements management tool but 10 out of 12 companies they studied use a configuration management tool.

Unlike the above studies which are multiple case studies Zowghi et al, (2001) conducted a single case study. Their study was focused on RE practices where stakeholders are geographically distributed. The analysis of the RE practices in the case study has shown problem areas such as communication, planning, lack of prototyping and tool support which are much similar to that found in (Curtis et al , 1988) and (Emam & Madhavji, 1995). In this particular case of geographical distributed stakeholders, communication channels between them are limited to electronic meetings (emails and video conferences) which are much less effective when compared to face-to-face meetings in typical software development. Geographical and cultural differences laid limitations on planning. Improper communication channels have slowed down the validation process.

A recent survey by Juristo et al, (2002) on European industries has shown immature RE practices still persist. Their results are much similar to that found in above surveys *viz.*, lack of proper documentation, improper tool usage etc. The authors say even though many problems have been highlighted and addressed by the research community (such as above studies) there is still a gap between practical work and research. However, awareness of available tools for RE has increased but many companies have not yet adapted to multiple tool usage. This is due to lack of tool integration facilities. Most of them were using only a word processor to aid RE process. The result of the survey laid stress on communication between researchers and practitioners (a similar thing, gap between research and practitioner community was explored by Nikula et al, (2000)).

The latest survey was reported by Neill & Laplante in the year 2003. This is a web based survey based on two other similar web based surveys one hosted by Macquarie University in Sydney, Australia, and the other by the University of Calgary (McPhee & Eberlein, 2002). The results of earlier one by Macquarie University are still not available and the later one was published in 2002. The survey by (McPhee & Eberlein, 2002) has shown that profound RE techniques are still not familiar in the industry and good communication channels are essential for effective RE. Most of the respondents in this survey feel that they do not allocate enough time to RE. The survey by (Neill & Laplante (2003) revealed that 50% of the respondents use scenarios and use cases for requirements elicitation. And only 30% used OO techniques for modeling requirements which contrasts with the use of use cases. Prototyping was done (60% of the respondents) despite the fact that most of the responses were using water fall model which does not involve prototyping. One of the interesting things that were found in this survey is that in spite of insufficient RE practices most of the respondents (70%) said that the customer was satisfied with the end product.

Authors	Year of Publishing	Research Method
Curtis, B. Krasner, H.& Iscoe, N.	1988	Case Studies
Lubars, M. Pot s, C. & Richter, C.	1993	Case Studies
El Emam, K. & Madhavji, N.H.,	1995	Case Studies
Nikula, U. Sajeniemi, J. & Kalvianen, H.,	2000	Industrial Survey
Zowghi, D. Damian, D. & Offen, R.	2001	Case Study
McPhee, C. & Eberlein, A.	2002	Web Survey
Natalia J, Ana M. Moreno, & Andrés Silva	2002	Industrial Survey
Neill, C.J. & Laplante, P.A.	2003	Web Survey

Table1: Tabular representation of related works

3.2 Current Work

This research is much similar to the above works in the view of methods that we have adapted. In the current work face-to-face interviews were conducted, much similar to most of the above works except for McPhee, C. & Eberlein, A. (2002) and Neill, C.J. & Laplante, P.A. (2003) which are web surveys. We selected one representative (respondent) from each organization which contrasts with most of the above related works where some of the respondents are involved in the same project (and organization). The selection of respondent was based on his/her involvement (experience) in the software development activities which is inline with above works. However, we focused on exploring the gaps between state of the theory and state of the practice which is quite similar to the objective of the study done by Nikula et al, (2000).

Apart from focusing on RE core practices Curtis et al, (1988) and Lubars et al, (1993) also discuss domain knowledge issues (knowledge sharing). Nikula et al, (2000) and

Juristo et al, (2002) discuss tool related issues. But none of the above works address the user involvement during RE. We integrated these three issues together with the core practices in order to increase the scope of results coupled with these issues.

Summary

The survey by (Curtis et al, 1988) is the first one in the field of RE and also mostly referenced in similar work, but this study not only focused on RE but also on other software engineering aspects. This study is different from other studies discussed above in the way that the study was coupled with human behavioral process. Some of the findings that are common in the all the above studies are:

- Improper communication channels
- Improper tool usage
- Importance of domain expertise
- Formal methods still not familiar
- Lack of coordination between research and practice

By and large, it is evident from the above studies that the RE as practiced in industry is still immature.

"Between thought and expression
there lies a life time"

-Bob Dylan

Chapter 4

RESEARCH METHOD

The objective of this chapter is to introduce the reader to the empirical research methodology implemented in conducting this thesis, after which selection of the specific method is discussed. This chapter also illustrates the design and objective of the questionnaire implemented in conducting our study. The chapter concludes with a discussion supporting the validity of this thesis.

4.1 Introduction

According to (Brilliant & knight, 1998) empirical research in software engineering is an observation of software development activities in an experimental sense. Empirical research could include a range of experiments, qualitative studies, surveys or archival analysis (Basili et al, 1999) and case-studies. The objective is to find out unknown information out of existing hypothesis, by way of gathering information and conducting qualitative and quantitative analysis.

4.2 Research Methodology Selection

Based on the classification of empirical research methodologies, we have chosen survey as the appropriate method for our study. In view of our intension to evaluate the core RE practices from an Industrial perspective to the academic perspective , we perceive that a questionnaire based survey opens us an opportunity to interact and thus gain knowledge from experienced professionals, ultimately giving us the first hand insight. Further more number of questions can be asked on a specific topic, considering flexibility on analysis (Robson, 2002:230). In survey method, the researcher chooses a sample of respondents from a large population and administers a standardized questionnaire (Robson, 2002:230). Representative samples of small population are taken, based on whose analysis the larger population is generalized. Research by survey method is very flexible, facilitating access to several variables in the field of study.

Other forms of empirical research methods like an experiment or a case-study might not be appropriate for our study. Experiments are a form of study, where the researcher needs to control the study environment; it is most suitable when the state of some independent variable of study have to be changed in order to direct the study (Basili et al, 1999). On the other hand a case-study is implemented to explore and understand a theory, a process

or a tool. It suits well in an environment where one needs to study "How" and "Why" questions, in this scenario researchers do not have control over the state variables in the environment of study (Perry et al, 2004).

Our modus- operandi also relates partially to a case-study, since we are going to study the RE process of various organization, considering each of them as an independent entity, consequently collecting information for the conclusive analysis.

4.3 Objective of the Survey

- To identify how the requirements engineering principles are practiced in the industry.
- To understand the awareness and importance associated with RE from the industry perspective.
- Identify the constraints concerned with implementing principle of RE in the real world.

4.4 Design of the Questionnaire

The questionnaire is available in the appendix

In designing the questionnaire, firstly we have identified the core requirements engineering practices primarily based on the illustrations of authors (Kotonya and Sommerville, 1998) and (Wiegers, 2003). The questionnaire is classified into four parts with 19 questions in total – Introduction with general questions, Elicitation, Analysis-Negotiation, Validation and finally Management. The objective of the questionnaire is to exclusively gather knowledge relevant to fundamental practices of all distinct phases of RE, hence we have not focused into specific details. However, sub-questions were posed wherever necessary. We have attempted to keep the questionnaire as straightforward as possible, in view of professionals from diverse backgrounds.

The objective of Questions 2 to 9 is to extract information related to Requirements Elicitation, Analysis and Negotiation, covering the following activities:

- Software Development Model.
- Personnel Involved in Elicitation.
- Method of eliciting requirements.
- Background knowledge acquisition.
- Extent of user Involvement.
- Requirements Prioritization.
- Requirements Storage.
- Requirements Modeling.

The objective of Questions 10, 11, 12 is to extract information related to Requirements Validation, covering the following activities:

- Requirements Consistency and integrity.

- Requirements Evaluation.
- Inspections.
- Inspection techniques.

The objective of Questions 13 – 17 is to extract information related to Requirements Management, covering the following activities:

- Method of Specifying Requirements.
- Traceability.
- Configuration Management.
- Requirements Management tools.

Finally, the questionnaire also consists of an open ended question (18), where we have asked the interviewee to give us brief account of problems encountered that are explicitly related to RE during their tenure.

4.5 Survey Procedure

After having designed the questionnaire, our strategy was to reach out to six organizations, within which three are to be large organizations i.e. companies which have significant experience in developing software solutions, and have a mature outlook in the industry, the other three will be small or medium sized organizations with comparatively lesser experience in producing software. Based on the feasibility issues such as the effort and time allotted for this thesis, we have restricted ourselves with this specific strategy for choosing organization for our study.

For this study we have identified one individual from each organization, who is involved in extensive development, more specifically working with requirements specification. We sent in an overview of the thesis to all the six interviewees, so that they will have a clear understanding of our motive. The interviewees were assured of their company's confidentiality.

The questionnaire was administered by way of conventional face to face meetings, which lasted up to 40-55 minutes. Though the questionnaire had closed ended questions, we have attempted to have short discussion for almost all the questions to grasp a lucid picture of each companies RE routines.

As the last part of the interview, we have once again explained the purpose of the study and its objectives, after which we asked them their intuition on the questionnaire, if we have missed out any vital aspects concerning RE from the industry perspective.

4.6 Analysis Procedure

Chapter 6 introduces to the analysis of this research. The organizations will be referred from A to F respectively, all the RE practices of each organization will be compared in a tabular form i.e. one table per practice, wherever feasible. Preceding each table is a short discussion based on the practice and the information gathered during the interview.

However, tables were not used where the information could not be appropriately represented.

Finally, views from the analysis are discussed in the form of answers to the research question quoted in chapter 1. Chapter 6 will have the general conclusions based on the study, apart from directions for future research.

4.7 Validity of Research

The research should be carried out in an unbiased (impartial) way so that it yields valid and reliable results Robson (2002). Robson (2002) mainly discusses three threats that potentially affect the validity *viz.*, reactivity, respondent bias and researcher bias. Reactivity in the sense that the presence of researcher may influence the setting of the study in particular the subject of study i.e., the threat of alteration in response from the respondent due to the interference of researcher. Respondent bias, respondent may some times hinder or withhold information (information might be confidential or classified) or else provide answers in the way they think that the researcher wants. This is sometimes due to lack of assurance of confidentiality (anonymity) from the researcher side. On the other hand researcher bias is the assumptions and preconceptions made by the researcher i.e., selection of interviewees, interview questions, selection of data for analysis which may have affect on the research setting.

Robson (2002:174,175) has listed some strategies in order to overcome these threats. We have adopted some of the strategies that we felt are relevant to our research method and are possible with in the frame of our study (time bounds). Below we discuss the strategies that we used to mitigate the threats:

Peer debriefing: support and inputs from the peer group who are not connected to the study can help in reducing researcher bias (Robson, 2002). During the course of the study the researchers had a couple of brief sessions (informal) with the co students contrasting each others views of the study. The inputs of the other members of the peer group were taken whenever necessary through direct email and mailing lists. These external inputs were specifically useful in refining our study through multiple views and thus reducing researcher bias.

Member checking: After every interview a summary of the interview transcript was prepared highlighting the researcher's interpretations of the session and a copy of it was sent to the respondent. This is to make sure that the interpretations are correct (reducing researcher bias and reactivity). However, there are some problems in doing so such as the respondent wants to suppress some information from the coded data (Robson, 2002). But these problems (respondent bias) were discussed with the respondent at the time of interview assuring him full confidentiality and anonymity (these can be checked by the respondent with the final version of the research report which will be e-mailed to each respondent).

Observer triangulation: Participation of more than one observer, in this case interviewer in the data collection. For the purpose of data collection, participation of

more than one observer is required. During this study half of the interviews were conducted by one researcher and the other half by another researcher to reduce reactivity and researcher bias.

Other threats: According to Robson (2002:231) if the questions used in the survey are ambiguous or complex to understand then there is a threat to *internal validity* of the research. In such a case it will not aid to yield valid results. But this is a major problem when using postal or web survey where respondent cannot get any instant assistance regarding the interpretation of question (there are higher chances that the respondents might not bother about the actual interpretation). In face-to-face interviews one can always elaborate the questions. However, to overcome this problem in the early phase itself the questionnaire was developed and reviewed several times to eliminate any ambiguity or complexity in the questions. A simple and straightforward questionnaire was then set up such that unanimous interpretation of the questions was possible by the respondent group (questionnaire is provided in the appendix). Furthermore all the respondents (interviewees) were posed the same set of questions (standard questionnaire was used) in a similar fashion to yield reliable response as suggested by Robson (2002).

Summary

This chapter introduces the reader to research methodology implemented in this thesis, along with description on the choice of the method. The design and categorization of the survey questionnaire is discussed, along with objectives of the survey. The last section 4.6, discusses the validity aspects of this study along with associated threats.

"If you study to remember, you will forget,
but if you study to understand,
you will remember"

- Anonymous

Chapter 5

ANALYSIS AND RESULTS OF SURVEY

In the previous chapter we have discussed the research methodology implemented in conducting this study. In this chapter we illustrate detailed analysis of the study based on the results of our investigation.

5.1 Introduction

We have conducted personal interviews with six organizations, with the primary objective of identifying the process of core RE practices of each of these organizations. During the study we have also identified the major constraints the requirements analysts encounter with regard to implementing RE practices that will discussed in section 5.4. In this chapter we present the results of our study and in the second half we will answer the research questions based on the findings from this study. The questionnaire implemented in the study is available in the appendix.

5.2 General perspective

Before going any further, it is essential to present a general perspective drawn from this study. Our opinion i.e. in reality established RE methodologies might not map with the industry is probably true. Nevertheless in this context, it was clearly evident that majority of the organizations have a modest approach towards requirements engineering process. Organizations tend to believe that their current maturity level in RE is good enough, and do not usually consider it essential for the professionals to be explicitly trained. However, none of the organization we have studied had significant complaints about their existing technical methodologies, except for minor process improvement issues. On the other hand analysts realize that their existing process could be improved to a great extent, in order to write better requirements and produce quality software. The majority of the problems expressed are typically associated to social, communication and organizational factors.

5.3 Results of the Study

The organizations whose RE process we have studied will henceforth be referred as A, B, C, D, E and F respectively. The first three companies (A, B and C) are fairly smaller companies with less than 15 employees when compared with the other three (D, E and F)

having more than 50 employees. Companies D, E and F have significant experience in developing software, and have a mature outlook in the industry. The other three A, B and C are small or medium sized organizations with comparatively lesser experience in producing software

To start with, it is motivating to identify the specific development model implemented in each of these organizations. Every development model has pros and cons; accordingly by and large the choice of the model depends on the project (Sughosh, 2003). For instance, a waterfall model is a widespread model most organizations use. However, a prototyping development model is known to produce better software when compared to the waterfall model (Wilson, 92). Likewise stakeholders cannot be really sure about their requirements until they see a working model; However, waterfall model does not accommodate changes in the later stages (Sughosh, 2003).

In the study it was interesting to note that all the companies were using hybrid development models (a combination of development models). Much of the models are inclined towards waterfall model or a blend of waterfall and other evolutionary models (Table 1). Furthermore it was evident that organizations often tend to adhere to dependable practices rather than a typical development model.

Table 1 - Software Development Model

Company	Waterfall	Prototyping	Incremental	Evolutionary	Other(Hybrid)
A	X			X	
B					X
C					X
D	X			X	X
E	X			X	X
F					X

In our study we realized that only two large organizations i.e. D and E (Table 2) have personnel dedicated to eliciting requirements. RE literature contains ample recommendations to involve dedicated personnel for this role (Wiegers, 2003) (Sommerville & Sawyer, 1997). However, though the other organizations recognize the significance of the role, they consider that it is way too expensive to have a personal dedicated for this task which is usually the initial phase of project. This assumption diverts from the fact that RE is a persistent process throughout the development. Furthermore, a majority of the organizations fail to recognize RE as a multifaceted process characterized by complex routines, thus not assigning the required magnitude.

Table 2 - Total no of Employees & Personnel responsible for RE

Company	Total No of Employees	RE personnel
A	<15	Senior Developers
B	<15	Senior Developers
C	<15	Senior Developers
D	>50	Strategic Project Manager
E	>50	Strategic Team Leaders
F	>50	Project Manager

5.3.1 Requirements Elicitation

The elicitation phase is characterized by close interactions between the developers, stakeholders and the end-users. Most organizations follow the approach of interviewing and informal discussion with the stakeholders to elicit requirements (Table 3). Other methods of elicitation illustrated in the literature such as scenarios, joint application development (JAD), Observation – social analysis or soft system methods were not practiced or unknown to any of the organizations. (Table 3) Two of the larger organizations (D,E) we have studied seem to work mostly with similar kinds projects hence implement a "re-use" approach for a large amount of the requirements specification, usually based upon previous projects and the development experiences from those projects. Company B relies on prototyping approach to a major extent, and company A implements prototyping occasionally.

During the process of elicitation the discussion is primarily centered on the functionality of the application, engineers often do not realize the importance to explore the quality and performance attributes of requirements. Analysts express that most of elicitation related problems are due to stake holder's inability to express their tacit knowledge.

Table 3 - Requirements collection

Company	A	B	C	D	E	F
Interviews	**X**	**X**	**X**	**X**	**X**	**X**
Prototyping	**X**	**X**				
Scenarios						
Use Case			**X**			
Soft system method						
JAD						
Re-Use				**X**	**X**	

5.3.2 Understanding the Application Domain Knowledge

From the study we found that relatively all the organizations have the similar approach towards gathering application domain knowledge, typically through preliminary discussions, technological presentation and technical literature. Two of the larger organization i.e. D and E, who considerably work on similar projects and specific clientele, have domain experts who have explicit proficiency in the domain (Strategic

Project Managers and the team leaders). They have in-depth knowledge regarding the problem domain as well as the business domain. These domain experts are responsible for educating the developers and the project leaders on the constraints and implications of the proposed system, thus acting as a central hub for communication throughout the development.

Apart from the informal interactions with the client, developers visit the customer's base whenever essential. These activities appear to be the most prevalent methods to gain domain knowledge. None have expressed any obstacles caused by virtue of inadequate domain knowledge understanding.

5.3.3 User Involvement

It was observed that user involvement is considered crucial specifically during the initial phases of the RE, typically in ruling out the issues on cost vs. functionality. Table 4 shows that majority of the companies involves the stakeholder during the elicitation, analysis and negotiation phases. RE personnel believe that the greater parts of the functional conflicts are usually resolved in these preliminary phases, and the need for users' involvement throughout the project would not arise. Moreover engineers from companies A and D presume that when users are involved throughout they tend to request additional functionality, eventually digressing from the initially agreed requirements. However, company B has ascertained that user involvement is usually not required once the requirements are negotiated and prioritized. Most often users are convinced by the preliminary application prototype.

On the contrary, companies C, E and F (Table 4) believe that it is very important that the users are not only involved throughout development but also gradually educate them about the system, so that their perception towards the developing system is enriched, as a result users can also assist them by delivering valuable suggestions and improvements. In particular, all the companies expressed the need of informal discussion session as and when necessary.

Table 4 - User Involvement

Company	Only Elicitation	Elicitation, Analysis & Negotiation	Prioritization	Throughout the Project
A		**X**		
B		**X**	**X**	
C				**X**
D		**X**		
E				**X**
F		**X**	**X**	**X**

Looking into the post elicitation activities all the six companies prioritize the requirements but most of the interviewees said that the final decision regarding the prioritization is done by the client. Based on the inputs on cost and risk ratings from the technical leads, requirements are prioritized in negotiation with the stakeholders. However, it seemed as if the requirements are classified only into "must haves" and

"provisional", and lack theoretical granularity of prioritization such as requirements per release, cost per implementation and other distinct considerations for prioritization.

Smaller organizations i.e. A, B and C have said that prioritization is usually based on their understanding of the goals of the system, nevertheless valid inputs from the end user are also considered. Prioritization to a large extent also depends on the system being developed. Company B which strictly follows prototyping does prioritization only if the project is large.

5.3.4 Requirements Specifications

We discuss in chapter 2 , that though formal methods are known to be effective in specifying requirements, natural language specifications remains to be the most popular and practical approach for writing requirements specification (Table 5). Interestingly this supposition holds strong in the real world of requirements. Requirements analysts are accustomed to natural language specification based on its ease of use and flexibility. Moreover developers believe that it is much easer for them and their clients to understand the natural language specification without additional training. However, Use Cases are used to decompose the complexity of typical requirements, more specifically when accurate semantics are required to describe a specification.

Table 5 - Requirements Specification

Company	Natural Language	Formal Methods	Other
A	X		
B	X		UseCase
C		X	UML
D	X		
E	X		
F			Use Case

Only one organization i.e. C, has been extensively working with formal modeling techniques such as UML, ER models and other structured design methods for requirements specification. Interviewees were of the intuition that implementing formal modeling techniques might increase the cost of development, apart from delaying the course of development. On the other hand they also have mentioned the situations where natural language notations have led to ambiguity, misinterpretation and lacking design detail.

Table 6 - Requirements Modeling

Company	OO Model	SAD	ER Model	No model	Other
A				X	
B				X	
C					UML
D					UML
E					Prototyping
F					UML

It was evident that most developers have modest knowledge on requirements modeling. However, except for companies A and B all other companies do requirements modeling (Table 6). Among the modeling techniques UML is the popular technique used (being used in companies C, D and F as seen on table 6). As far as company D is concerned modeling is done based on complexity of a specific requirement to decrease its level of abstraction. It was obvious that not many persons in the organizations have the required proficiency in modeling methods.

5.3.5 Requirements Validation

All companies have considerable level of client/user involvement in the validation phase but only company A has said that only the client is held responsible for validating the requirements and the final set of requirements is then developed based on the clients acceptance after validation. However, formal requirements inspections are also performed at companies C, D, E and F (Table 7) where checklist based inspections are most popular. But we found that only at company E inspections are aided by tools. Whereas in company A, much emphasis is laid on validation (agreement of requirements with the client) and followed by an informal inspection of requirement. In company B no inspection is done instead the focus is laid on prototyping.

Table 7 - Requirements Inspection

Company	Yes	No
A	**X** (Informal)	
B		**X**
C	**X**	
D	**X**	
E	**X**	
F	**X**	

5.3.6 Requirements Management

We have discussed in Chapter 2 that a requirement management tool serves as an automated assistance all through the development. It was surprising to note that only two of the organizations D and E (Table 8) were making use of a requirements management tool. The rest of them seem to be content with spreadsheets or word document templates (Table 8). Another noteworthy point is that these organizations did not recognize the several possibilities of requirements management tool and its added benefit. Most of the version control and tractability functions are managed by making manual notations.

Table 8 - Storage of Requirements

Company	Spread Sheet	Word processor	Database	Tool Support
A	**X**			
B		**X**		
C		**X**		
D				**X**
E				**X**
F	**X**			

On the other hand, we have learnt that to a great extent the environment in which the project is developed is also a factor to be considered. For instance, organizations involved with fairly smaller projects and comparatively less number of persons might not need the support of automated assistance. Consequently the expenditure on a commercial tool is saved.

Change Management

Except for companies D and E, the other organizations have a very informal approach towards change management. Companies D and E have a structured coordination for conducting change impact assessment and change requests through a change control board , they also have a proficient configuration management system. Company F has expressed that their change management process is not up to the mark, and often runs into chaos. However, they have already considered improving the existing system to assist in making informed decisions. The project manager is generally responsible for assessing the impacts of change requests, but the final verdict is issued by the top management.

Alternatively in the smaller organizations i.e. A, B and C, the entire change management task is managed by a group of senior developers and the project manager They are responsible to evaluate the impact of change requests and then take crucial decisions. Later the people influenced by the changes are notified through documents and discussion sessions. It was interesting to note that in this organizational setting a complex requirements management tool or a configuration management tool would require additional resources, which could be avoided. Nevertheless the interviewees made an important note that a greater part of the choice of management scheme would to a large extent depend on the size of the project, accordingly the number of requirements to be managed. Moreover when fewer people are involved coordination is not usually complex when compared to larger projects with more number of people involved.

5.4 Research Questions

Based on the analysis drawn from the previous section, in this chapter our views will be presented in the form of answers to the research questions posed in chapter 1.

1) To what extent do the software engineers in the industry implement the principles of Requirements engineering in contrast to the theoretical practices?

In the previous section, we discuss how the core RE practices are implemented in the industry; the discussion also introduces incompatibility of certain practices in the real world scenario. In this section we discuss some noteworthy aspects concerning realistic practices from a general perspective.

In the real world of requirements; requirements analysts strongly believe that every software project is different. This is apparent from the usage of development model in the surveyed companies; all of them were using a blend of development models tailored according to the project needs. Consequently a typical method or a routine used in a specific project most often cannot be implemented in other projects. This conception holds strong typically on Market driven vs. Customer driven projects.

Observing the initial phases of the RE process it is quite evident that even though there are many techniques available for elicitation, interviews and discussions proved to be predominant. This is due to the fact that these methods are simple when compared to other techniques and require no training and fewer resources. This also holds true for specification techniques where natural language is almost used by all the companies and the formal methods still lack popularity. Likewise larger companies with more resources and technical workforce were able to implement methods such as modeling to a greater extent, unlike the smaller companies. Companies tend to follow ad-hoc or informal approaches, in order to fit into their existing system and the organizational setting

Though the literature has ample recommendations for extensive user involvement for producing successful software products, it was apparent from the study that this aspect is largely influenced by social and communication factors apart from users being unable to gauge its importance.

Despite the availability of numerous tools for requirements management, still word processors and spread sheets are being used. In our study only two companies were using explicit tools for requirements management. The factor is largely influenced by resources, extent of the project and the companies recognizing benefits of a tool.

At this juncture, there are several efficient methodologies and tools to support the existing requirements engineering process to create a quality structure for developing software. However, from this study it was evident that not many people in the organization are motivated to induce change into the existing process.

2) What are the constraints the software engineers encounter when it comes to implementing the RE practices?

In this section, we discuss most compelling constraints, which we found to be interesting during this study.

Consequence of the present day situation

The very first impression we could draw after all the interviews is - almost all the organization are reasonably conscious of the established RE methodologies. The major cause for the methodologies not being able to put into consistent practice is, due to the soaring competition on winning a project and delivering the product within strict deadlines, often software engineers are forced to override the known best practices and process standards. Ultimately relying on their experiences on previous successful projects. However, the interviewees ascertained some degree of risk with such an approach which is usually inevitable.

Motivating Users

User Involvement is known to be a critical factor. However, analysts find it difficult in identifying the right user representatives, and then motivate them to involve and assist analysts in exchanging views and negotiate requirement specifications apart from discussing feasibility concerns. Analysts lament that most often the end-user

representative either do not find enough time or fail to realize the importance of their involvement. On the other hand, some users believe that it is solely a part of the analysts' strategy to extract their requirements.

User Representatives
The smaller organizations expressed that due to the complex hierarchy at the clients' organization, it becomes cumbersome for the analysts and the clients to agree and negotiate on an unambiguous set of requirements specification, ultimately causing untimely creep of requirements. In some projects, analysts gather requirements from the marketing team, and then ultimately realize that relying on surrogates does not lead to needs of real users.

Tacit Knowledge
Analysts express that most of the requirements related problems are due to stakeholders' inability to express their tacit knowledge on the desired requirements. More so, at the right point of time during development.

Untimely Creeps
Most often stakeholders realize or learn more about their requirements after the project commences, thereby burdening the developing teams with extra-overhead by change requests and additional functionality. As a consequence altering the baselined requirements. Eventually the budget and release dates are misjudged.

Vague Requirements
They are numerous instances where developers realize that some requirement specification fall short of detail description. Eventually this vague specification leads to ambiguity, misinterpretation and difficulty in implementation.

One of the most common problems is that stakeholders tend to ask for too much i.e., more than what can be produced in available time and budget.

Another noteworthy viewpoint on the process of requirements engineering is that time and resources are vital entities which influence the established methodologies from the realistic perspective.

5.5 Large Vs Small Organizations

As initially anticipated two of the large companies i.e. (D, E) have a well defined and elaborate process of RE, which they often repeat and iterate for improving quality. The task is often managed by dedicated Strategic project managers. Company F has no specific methodology, even then their practices are well managed by experienced business analysts and project managers. Analysts from these organizations expressed their success rate to be commendable till date.

On the other hand, smaller organization (A,B,C) where number of employees are less than 15 did not have well defined roles, Most of the RE is managed by senior developers and project leader who have formal education in software engineering. However, even

with such ad hoc organizational practices and simple hierarchy, these organization still seem to manage to get successful results. However, they have faced circumstances where stakeholders were not satisfied with the end-product, which lead to unanticipated effort.

In line to this, when compared to larger organization, small organizations are engaged in comparatively smaller projects with lesser requirements and cater mostly to customer-driven projects.

From the study it was evident that mature organization might need a well structured and detailed process to achieve their goals unlike the fairly smaller organization who can manage with an informal approach bearing on the plainness of their organizational structure and simple hierarchy

5.6 Limitations of the Study

As discussed in this thesis RE is a multifaceted process consisting of various phases, hence usually a group of analysts and senior technical managers are involved in the process. In this study we have contacted one person per organization who is well conversant on their process. Evidently it is improbable to extract in-depth information on the constraints on each of the technical process methodology. However, we could grasp a comprehensible overview from the questionnaire and through the discussion session.

In this chapter we present our interpretation from the results of the study. However, we are conscious that others could have interpreted the transcripts in a different manner, or could have drawn more or different conclusions from the study's transcripts.

We have chosen organizations of different size and maturity; we do not propose to generalize the results of this study to a larger population. The objective is to illuminate some interesting findings within this sample of population.

Summary

In this chapter, we have presented the analysis of the study, based on the interviews conducted in six organizations. The chapters primary focus is to compare the RE practices of these organization to the established methodologies. Later the research questions posed in chapter one are answered based on the conclusions drawn from this study. The next chapter we have presents the conclusions for the thesis.

"The road to success is always under construction"

- Arnold Palmer

Chapter **6**

CONCLUSIONS AND FURTHER RESEARCH

In this chapter we present our conclusions based on the analysis drawn from the study conducted in six software organizations. Directions for future research work are also discussed in the later half of the chapter.

Conclusions

There is a lot of emphasis associated to requirements engineering in various literature, However, form this study it was obvious that industry does not assign so much of magnitude to RE.

There is no universal approach or a set methodology that could be implemented to address issues related to RE in any organization. Every project is different and differs on various factors such as flexibility, budget, and resources - to some extent the organizational setting also influences the RE process.

Perhaps it is unrealistic to presume that informal or ad-hoc development practices influence the quality of the software product. We base our impression on the fact that knowledge resides in the intellect of the individuals. Companies believe that it is important to discover the existing knowledge, accordingly a team of knowledgeable developers can deliver an outstanding product irrespective of their development practices.

It is important to note that non technical issues such as effective communication between users-developers, organizational skills for the requirements analysts and social factors have a lot of influence on the RE practices.

Natural language notations are being used mainly in the industry, though individuals are conscious of the benefits by using formal methods, based on the plain factor of simplicity natural language remains to be popular among the industrial community.

It was interesting to note that all the problems faced during developing software, most often are solved implementing the knowledge within the organization. It is very unlikely, that software developers think in terms of learning a new methodology. It is difficult to

bring in a change into the existing process; people tend to adhere to known methods, more so based on "easy of use" and practicality.

Finally a point worth mentioning, one cannot become a good requirements analyst by way for formal training or exceptional educational qualifications. As Karl Wiegers (2003a) says, requirements analysts are grown with years of relevant experience.

Bottom-line

At this juncture, we are convinced that there is certainly a gap between the theoretical perspective of RE when observed from industrial view point. The most apparent concern is that best practices suggested in the literature most often seem to overlook aspects such as budget and resources within an organization. Where as from the industrial standpoint, money, time and resources is what matters, eventually influencing the entire development process.

Requirements exist in the intellect of stakeholders, end-users and developers – hence it is up to the analysts to discover and give them an appropriate composition. It is improbable that "*a specific process*" or a "*set methodology*" can be a perfect approach in addressing RE in a broad sense. From a realistic perspective in order to stay competitive and efficient, organizations have to improve on their existing RE practices and allow them to reign by implementing their own strategy that suits their work culture. Techniques, approaches and tools might help, but ultimately the crux for building successful software depends on the quality and insight of the personnel involved in its process. It is essential that the developers work as a team and opt for the best methodologies or techniques that keep the RE process practical and uncomplicated, besides establishing a collaborative environment with the stakeholders.

> "Requirements are not engineered …they are nurtured"
>
> - (Thomas & Hunt, 2004)

Further Research Work

In our research we have conducted surveys on different organizations and RE practices vary in them, this is due to the fact that every project was different and every organization (needs) was different from another and they have to adopt their own version of RE practice which is much like an instinctive task when compared to the state of the art. So it is worthy to conduct case studies on different organizations where there is a large scope for organizational and project needs taken into account. And further on different personnel from the same organization can be interviewed for multiple view points of the project or organizational needs. In this way the current work can be extended to generate a framework or recommendations for RE practice on an organizational level.

Even though there are many tools available to handle RE especially requirements management (which includes traceability) only two of the large companies were using

specific management tools and the remaining companies were mostly contended with word processors and spread sheets. Traceability issues caused much overhead in RE as they had limited resources to automate these issues. Therefore we suggest that in future research it would be worthy enough to address these management issues for improvement in the small companies (or companies with limited resources).

Since we have observed that non automated methods - specifically for requirements management are popular by majority, perhaps research emphasis should be laid on non automated methods to support decision making. Furthermore factors such as organizational settings should also be considered in developing RE solutions.

Another suggestion would be to research on the extent of client/user involvement in the RE process. Since requirements are change-prone and it is found that most of the changes are requested by clients/users.

Development of new RE methods and approaches that are more straightforward and easier to apply, so that it becomes more probable that industry adopts them.

REFERENCES

(Andriole, 1994) Andriole, S.J. (1994). *Fast, cheap requirements: prototype, or else!*. Journal: IEEE Software, Davis Alan, Royce Winston, vol: 11 issue: 2, IEEE. Pages: 85-88.

(Aurum & Wohlin, 2003) Aurum, A.; Wohlin, C. (2003). *The fundamental nature of requirements engineering activities as a decision-making process*, Journal: Information and Software Technology, vol: 45 issue: 14 pages: 945-954 publisher: Elsevier.

(Barker, 2000) Barker D. (2000). *Requirements modeling technology: a vision for better, faster, and cheaper systems*. Journal: VHDL International Users Forum Fall Workshop, 2000. Proceedings, IEEE Computer Society. Pages: 3-6

(Basili et al, 1999) Basili V.R. Shull F. Lanubile F. (1999). *Building knowledge through families of experiments*. Software Engineering, IEEE Transactions *vol:* 25 *issue:* 4 IEEE, pages: 456-473 .

(Brilliant & Knight, 1999) Susan S. Brilliant, John C. Knight. (1999). *Empirical research in software engineering: A workshop*. ACM SIGSOFT Software Engineering Notes, Volume 24 Issue 3

(Brooks, 1987) Frederick P. Brooks, Jr. (1987). *No silver bullet: Essence and Accidents of Software Engineering*. Volume 20, Issue 4, IEEE Computer Society Press. Pages 10-19.

(Carter et al, (2001) Carter, R.A., Anton, A.I., Dagnino, A., Williams, L. (2001). *Evolving beyond requirements creep: A risk-based evolutionary prototyping model*, Journal: Requirements Engineering, IN: Proceedings. Fifth IEEE International Symposium, IEEE. pages: 94-101.

(Curtis et al, 1988) Curtis, B. Krasner, H. and Iscoe, N. (1988), *A Field Study of the Software Design Process for Large Systems*, Comm. ACM, vol. 31, no. 11, pages: 1268–1287.

(Davis, 1992) Davis, A. M. (1992). *Operational Prototyping: A New Development Approach*, IEEE Software, September 1992, vol: 9, issue: 5, IEEE. Pages: 70-79

(Davidson, 1999) E.J. Davidson (1999) , *Joint application design (JAD) in practice Journal*, The Journal of Systems and Software, vol: 45 issue: 3 pages: 215-223 provider: Proquest

(Duggan & Thachenkary, 2003) Duggan Evan, W. and Thachenkary Cherian, S. (2003). *Higher Quality Requirements: Supporting Joint Application Development with the Nominal Group Technique*, Journal: Information Technology and Management, vol: 4 issue: 4, Proquest. Pages: 391-408.

(Emam & Madhavji, 1995) El Emam, K. and Madhavji, N.H. (1995), *A Field Study of Requirements Engineering Practices in Information Systems Development*, Proc. 2nd IEEE Int'l Symp. Requirements Eng., IEEE Press, pages: 68–80.

(Firesmith, 2004) Donald Firesmith. (2004). *Prioritizing Requirements*, Journal of Object Technology, vol. 3, no. 8, pp. 35-47. http://www.jot.fm/issues/issue_2004_09/column4> Last accessed 10th January 2005

(Graham, 1991) Graham I. (1991). *Structured prototyping for requirements specification in expert systems and conventional IT projects*, Computing & Control Engineering Journal, March 1991, vol: 2 issue: 2, IEE/IEEE. Pages: 82-89.

(Gotel & Finkelstein, 1994) Author: Gotel O.C.Z. ; Finkelstein C.W. (1994). *An analysis of the requirements traceability problem , Journal: Requirements Engineering*, Proceedings of the First International Conference, pages: 94-101 publisher: IEEE Computer Society.

(Herlea, 1996) Daniela Elena Herlea. (1996). *Users' involvement in the requirements engineering process*, Knowledge Acquisition Workshop, University of Calgary. <http://ksi.cpsc.ucalgary.ca/KAW/KAW96/herlea /FINAL.html>. Last accessed 22nd December 2004

(Herlea, 2000) Daniela E. Herlea Damian (2000), *Challenges in Requirements Engineering*, Department of Computer Science, University of Calgary, Canada. January 2000. <http://pharos.cpsc.ucalgary.ca/Dienst/Repository/2.0/ Body/ncstrl.ucalgary_cs/1999-645-08/pdf> Last accessed 10th January 2005

(Hofmann & Lehner, 2001) Hubert F. Hofmann and Franz Lehner (2001). *Requirements Engineering as a Success Factor in Software Projects*. Software IEEE, Volume: 18 , Pages:58 – 66.

(Humphrey, 2002) Watts S. Humphrey. (2002). *Managing the software Process*, SEI Series, Pearson Eduction Asia.

(Jackson & Embley ,1996) Jackson, R.B. ; Embley, D.W. (1996) , *Using joint application design to develop readable formal specifications*, Information and Software Technology, vol: 38 issue: 10 publisher: Elsevier

(Juristo et al, 2002) Natalia Juristo, Ana M. Moreno, and Andrés Silva (2002), *Is the European industry moving toward solving requirements engineering problems?*, Software, IEEE, Volume: 19, Issue: 6, Nov.-Dec. 2002, Pages: 70 – 77

(Kovitz, 1999) Benjamin L. Kovitz. (1999). *Practical Software Requirements A manual of Content & Style*, Manning Publications.

(Kotonya and Sommerville, 1998) Gerald Kotonya and Ian Sommerville, (1998). *Requirements Engineering, processes and techniques,* John Wiley & Sons, Chichester.

(Karlsson & Ryan K, 1996) Karlsson J and Ryan K, (1996). *Supporting the selection of software requirements,* Journal: Software Specification and Design, Proceedings of the 8th International Workshop, pages: 146-149, IEEE Computer Society.

(Lubars et al, 1993) Lubars, M., Pots, C., Richter, C. (1993), *A Review of the State of the Practice in Requirements Modelling*, Requirements Engineering, Proceedings of IEEE International Symposium on , 4-6 Jan. 1993, Pages:2 – 14

(Leffingwell & Widrig, 2003) Dean Leffingwell, Don Widrig. (2003), *Managing Software Requirements: A Use Case Approach*, Second Edition , Addison Wesley

(McPhee & Eberlein, 2002) McPhee, C.; Eberlein, A. (2002). *Requirements engineering for time-to-market projects; Engineering of Computer-Based Systems*, Proceedings. Ninth Annual IEEE International Conference and Workshop , 8-11 April 2002, Pages:17 – 24

(Neill & Laplante, 2003) Neill, C.J. & Laplante, P.A. (2003). *Requirements engineering: The state of the practice*; Software, IEEE, Volume: 20, Issue: 6, Nov.-Dec 2003, Pages: 40 – 45

(Nikula et al, 2000) Nikula, U. Sajeniemi, J. and Kalvianen, H. (2000), *A State-of the- Practice Survey on Requirements Eng. in Small- and Medium-Sized Enterprises*, tech. report, Telecom Business Research Ctr., Lappeenrata Univ. of Technology.

(Nuseibeh & Easterbrook, 2000) Bashar Nuseibeh and Steve Easterbrook (2000), *Requirements Engineering: A Roadmap*, International Conference on Software Engineering, Proceedings of the conference on The future of Software engineering, ACM Press, Pages: 35-46.

(Perry et al, 2004) Dewayne E. Perry , Susan Elliott Sim , Steve M. Easterbrook (2004), *Case Studies for Software Engineers*, Proceedings of the 26th International Conference on Software Engineering

(Pressman, 2003) Roger S. Pressman. (2003). Software *Engineering Resources* <http://www.rspa.com/spi/ analysismodeling.html#datamodel> Last accessed 10th January 2005

(Reubenstein & Waters, 1991) Reubenstein, H. & Waters, R. (1991). *The Requirements Apprentice: Automated Assistance for Requirements Acquisition*, IEEE Transactions on Software Engineering, 17(3), Pages: 226-240.

(Robson, 2002) Robson, Collin (2002). *Real world Research: A Resource for Social Scientists and Practitioner-Researchers.* 2nd Edition, Blackwell Publishing Ltd.

(Rosenberg, 1998) Linda H. Rosenberg, Theodore F. Hammer, Lenore L. Huffman. (1998). *Requirements, Testing, and Metrics*, at the 16th Pacific Northwest Software Quality Conference, UTAH. NASA – Software Assurance Technology Center.

(Rogerson , 1996) Simon Rogerson. (1996). *Computers and Human Values, Originally* published as ETHIcol in the IMIS Journal Volume 6 Issue 5. Available on (last visited 23.11.04) http://www.ccsr.cse.dmu.ac.uk/resources/general/ ethicol/Ecv6no5.pdf . Last accessed 10th January 2005

(Ramesh & Jarke, 2001) Ramesh Balasubramaniam, Jarke Matthias. (2001). *Toward Reference Models for Requirements Traceability*, Journal: IEEE Transactions on Software Engineering, vol: 27 issue: 1 pages: 58-94 publisher: IEEE provider: Ebsco

(Rottmann) Dave Rottmann, *Joint Application Development (JAD)*, <http://www.umsl.edu/~sauter/analysis/488_f01 _papers/rottman.htm> Last accessed 15th Dec 2004

(Sawyer and Kotonya, 2004) Pete Sawyer and Gerald Kotonya (2004). *Software Requirements Engineering Knowledge Area Description*. SWEBOK, Stoneman version 1.0, Available from www.swebok.org

(Standish Group, 1994) The Standish Group, (1994). "*Charting the Seas of Information Technology—Chaos*." West Yarmouth, MA: The Standish Group International.

(Sommerville & Sawyer, 1997) Ian Sommerville and Pete Sawyer. (1997). *Requirements Engineering : A good practice guide*, John Wiley & Sons Ltd, Chichester.

(Sughosh, 2003) Sughosh P K (2003). *Software development models and planning* (Peer Publishing), <http://cpp.ittoolbox.com/ documents/document.asp?i=2295> Last accessed 30th Dec 2004

(Thomas & Hunt, 2004) Thomas, D and Hunt, A (2004). *Nurturing Requirements*, IEEE Software. vol: 21 issue: 2 pages: 13-15 publisher: IEE/IEEE

(UML version 1.4, 2001) Glossary of Object Management Group specification for UML, version 1.4 (2001) available from www.uml.org

(Wilson, 1992) Diane Wilson, Thyra Rauch, and Joeann Paige (1992), *Prototyping and the Software Development Cycle*, article is drawn from a number of sources at the ACM CHI '92 conference. <http://www.firelily.com/opinions/cycle.html> Last accessed 30th Dec 2004

(Woolridge, 1999) Richard Woolridge. (1999). *An Introduction to Use Case Analysis*, [Internet], Castek (CBD-HQ). Available from <http://www.cbd-hq.com/articles/1999/ 991115rw_caseanalysis.asp>. Last accessed 30th Dec 2004

(Wiegers, 1999a) Karl E. Wiegers. (1999). *First Things First: Prioritizing Requirements*, Software Development, Process Impact,

September 1999. Available from <http://www.processimpact.com/articles/prioritizing.html> Last accessed 30th Dec 2004

(Wiegers, 1999b) Karl E. Wiegers. (1999). *Automating Requirements Management*, originally published in <http://www.sdbestpractices.com/> Last accessed 30th November 2004

(Wiegers, 2000) Karl E. Wiegers (2000), *When Telepathy Won't Do: Requirements Engineering Key Practices*, Originally published in Cutter IT Journal, May 2000, available on <http://www.processimpact.com/articles/telepathy.html> Last accessed 10th January 2005

(Wiegers, 2003) Karl E. Wiegers (2003). *Software Requirements, 2nd Edition*, USA, Microsoft Press.

(Wiegers, 2003a) Karl E. Wiegers (2003), So You Want to Be a Requirements Analyst? *Software Development*, vol. 11, no. 7 (July 2003)

(Zowghi et al, 2001) Zowghi, D. Damian, D. & Offen, R. (2001), *Field Studies of Requirements Engineering in a Multi-Site Software Development Organization: Research in Progress*, Proc. Australian Workshop on Requirements Eng., Univ. of New South Wales, available at <www.cs.uvic.ca/~danielad/AWRE/Zowghi_AWRE.pdf>

Appendix

Questionnaire used in the interview

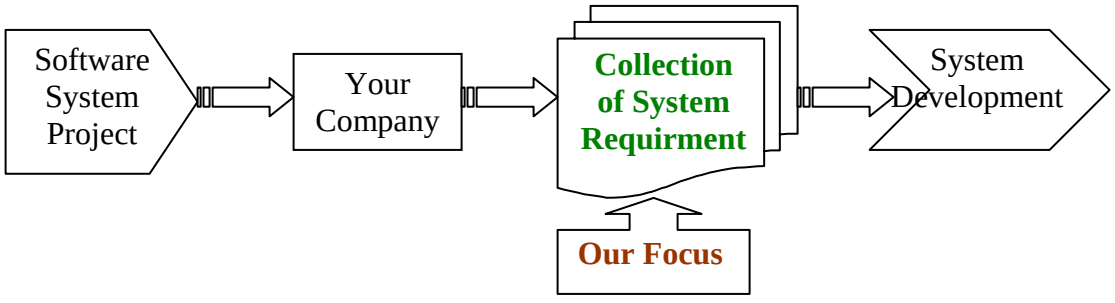
Introduction for the Interview

Who we are?
We are Software Engineering Masters Students from Blekinge Institute of technology, at the moment working on our Masters Thesis; we have chosen Requirements Engineering as our topic.

Why this Interview?
Our main focus of investigation is on "academic *perspective Vs the industrial perspective",* on the Requirements Engineering practices, hence we are interested in gathering information from big and small Software Organizations.

What is Requirements Engineering?
Requirements are defined during the early stages of a system development as a specification of what should be implemented. They are descriptions of how the system should behave, application domain information, constraints on the system's operation, or specifications of a system property or attribute.

What we expect from this interview?
A brief overview of your requirements collecting process



Questionnaire

Name of the Organization: _____

Name of the Interviewee: _____

1. Does your company have a pre-define specific Strategy to conduct the requirements elicitation process?

2. who are the people involved in Requirements elicitation process, are they trained specifically for this job?

3. What is the development model which you use or have used in your projects?
   a. Waterfall
   b. Prototyping
   c. Evolutionary
   d. Incremental
   e. Spiral

   Discussion:

4. What are the activities which best describe your requirements collection process?
   a. Interviews
   b. Prototyping (evolutionary, throw-away)
   c. Scenarios
   d. Data Flow Diagram or UML e.g. Use Cases (Semiformal Modeling)
   e. Soft System Methods
   f. Joint Application Development (JAD)
   g. Formal modeling

   Discussion:

5. during the initial phases of requirements gathering, do your engineers go to the customers to extract the domain knowledge?

   a. How do you gain domain knowledge
   b. How do you gain background knowledge of the system

   Discussion :

6. Can you briefly tell us, regarding the user involvement during the requirements elicitation process? At what levels are the users involved?
   a. Only during elicitation

b. Elicitation, analysis and negotiation
c. requirements prioritization
d. throughout the project

Discussion:

7. What is the customary procedure used for making notes of the collected requirements?

a. Database
b. paper files
c. tool
d. another ?

Discussion:

8. How do you classify, the requirements that are demanded by the stakeholders? Are the requirements prioritized?

9. If at all a specific technique is used for modeling requirements, which one would it, be?

a. OO model
b. structured analysis design
c. ER model
d. No specific model
e. Any other

Discussion:

10. What are the typical steps involved in the requirements evaluation/validation process?

a. Evaluating testability of requirements
b. prototyping ,
c. reviewing the requirements
d. any other methods.

Discussion:

11. Do/did you perform requirements inspections? Yes / No

12. If your answer is Yes, which techniques do/did you use?

a. Ad hoc walk-through
b. Formal walk-through
c. Automatic (using software tools)
d. Checklist
e. Scenario
f. Others

Discussion:

13. What is the method used for requirements specification (SRS)?
    a. Natural Language
    b. formal methods
    c. any other?

    Discussion :

14. How do you manage the requirements that are collected?
    a. database
    b. paper records
    c. tool
    d. any other method?

    14.1 does your system support traceability (yes / no)
    14.2 does the system record the source of requirements (yes / no)

Dicussion :

15. How does your company handle Requirements Change Management process?
    a. Follow a Change request process
    b. Follow a process for “Change impact assessment “.
    c. Do you make use of any configuration management tool?

    Discussion:

16. Do you use any Requirements Management tool?

17. Could you tell us about the most common/usual problems you encounter during the requirements collection phase?

18. In your opinion, does your company do enough requirements engineering? Yes/no. (a) What do you think is missing?

19. Do you think we have missed any important questions, relevant to our study?