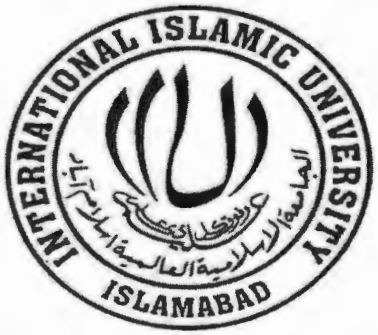
MS THESIS

EXTREME REQUIREMENTS ENGINEERING (XRE)



A THESIS PRESENTED TO

FACULTY OF BASIC & APPLIED SCIENCES

DEPARTMENT OF COMPUTER SCIENCE & SOFTWARE ENGINEERING

IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE

OF

MS IN SOFTWARE ENGINEERING

BY

Sonia Naz
355-FBAS/MSSE/F12

Supervised By
Dr. Naveed Ikram

Department of Computer Science & Software Engineering

Faculty of Basic and Applied Sciences

International Islamic University, H-10, Islamabad

MS
005.1
SOE

1. Requirement engineering
2. software engineering

Department of Computer Science & Software Engineering

International Islamic University Islamabad

Date: 17/12/2015

FINAL APPROVAL

This is to certify that we have read the thesis submitted by Sonia Naz, REG# 355-FBAS-MSSE-F12. It is our judgment that this thesis is of sufficient standard to warrant its acceptance by International Islamic University, Islamabad for the degree of MSSE.

COMMITTEE:

INTERNAL EXAMINER:

Muhammad Nasir

Lecturer

m.nasir@iiu.edu.pk

EXTERNAL EXAMINER:

Dr. Uzair Khan

Assistant Professor

uzair.khan@nu.edu.pk

INTERNAL SUPERVISOR:

Salma Imtiaz

Assistant Professor

Salma.imtiaz@iiu.edu.pk

SUPERVISOR:

Dr. Naveed Ikram

Associate Professor

Naveed.ikram@riu.edu.pk

DEDICATION

I would like to dedicate my research work to the
Most beneficent and the most merciful
ALMIGHTY ALLAH

HOLIEST man in the history of mankind
PROPHET MUHAMMAD (Peace Be Upon Him)
And

I also dedicate my work to my

PARENTS

Who have always been my nearest and enable me to accomplish this research work successfully.

Along with my hard working and respected teachers

A dissertation submitted to the

Department of Computer Science & Software Engineering,

Faculty of Basic and Applied Sciences,

International Islamic University, Islamabad,

as a partial fulfillment of the requirements

for the award of the degree of

MS in Software Engineering (MSSE)

DECLARATION

I hereby declare that this Thesis **"Extreme Requirements Engineering (XRE)"** neither as a whole nor as a part has been copied out from any source. It is further declared that I have written this thesis entirely on the basis of my personal efforts, made under the proficient guidance of my thesis supervisor, **Dr. Naveed Ikram.**

If any part of this research thesis proved to be copied or found to be a research of some other individual, I shall standby the consequences.

No portion of the research work presented in this thesis report has been submitted in support of any other degree or qualification of this or any other university or institute of learning.

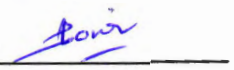
Sonia

Sonia Naz
355-FBAS/MSSE/F-12

Acknowledgement

First of all, I am thankful to Almighty Allah for his countless blessings. I am thankful to Dr. Naveed Ikram, my supervisor, for his continuous guidance and for being so helpful in many other ways. I am really privileged to work with such a competitive researcher, academician and a wonderful person. I also cannot thank enough my parents and family for their support and efforts.

Lastly, to all my friends who shared their support in one way or another, either morally or physically, thank you. I also express my deepest sense of gratitude for everyone who participated in the survey and helped me in successful completion of this thesis.



Sonia Naz
355-FBAS/MSSE/F-12

ABSTRACT

CONTEXT: Requirements engineering is a well-known and mature discipline. Requirements are properly identified, analyzed, specified and managed in this process. Still requirements engineering causes many problems in software projects. Requirements mistakes cause failure of projects in terms of cost or schedule overrun, failure in providing the specified functionality or producing software systems that do not have adequate quality. There is need to put the industrial best RE practices into practice to overcome these challenges.

OBJECTIVE: We have conducted an empirical study to identify RE practices, which are viewed as most valuable by RE practitioners worldwide. We have conducted a global survey with RE experts using an online questionnaire to find out RE valuable practices. Our target population was professionals who were involved in RE activities. The survey respondents were divided into multiple groups with respect to their characteristics in order to find out significant differences and commonalities between these groups. The results of the survey included 6 high value RE practices. These practices are related to "specification standards", "consultation with stakeholders", "prioritization of requirements", "use of diagrams", "having direct contact with customers", and "identification of requirements". Significant difference was observed between traditional and agile groups with respect to value of RE practices.

In the second step of our thesis we proposed an RE approach Extreme Requirements Engineering (XRE) based on the identified high value RE practices. This approach complemented the existing methods of agile with guidelines. XRE tells how the RE best practices can be used at their extreme in agile to get maximum benefits by putting minimum effort.

CONCLUSION: The highly valuable RE practices that have been identified through this study and the approach XRE will help future practitioners in balancing the cost and benefits of carrying out requirements engineering activities. It will be helpful in bringing agility to the requirements engineering process.

Contents

List of Figures

List of Tables

CHAPTER 01

INTRODUCTION

1. Introduction:

Software is part of our existence whether it is educational, professional or personal it made our life easy and accurate. Software is no longer a program to perform a task but it is the interaction of programs, data-structures and documentation and therefore is very complex to develop and maintain. Software is facing many challenges during different phases of development. These challenges might hinder a software development process and put the management in a situation which if not handled properly may lead to product overshooting budget and schedule as well as ending up in poor quality.

Requirements engineering process has the significant impact on all other phases of software development. Most common, time-consuming and expensive to repair errors are the consequence of inadequate requirements engineering process [1]. Previous studies [2] [3] show that most of the factors causing failure of software projects are related to requirements engineering so it has a critical impact on success or failure of software projects. If we do not perform the requirements process in a right way we may have a wrong product in our hands at the end of the project [4].

1.1. Requirements Engineering Valuable Practices

Different activities of RE (Requirements Engineering) such as gathering/elicitation, analysis, description, documentation and management are performed through requirements engineering practices. Some examples of good RE practices are using a standard document structure, have direct contact with customers to avoid unclear requirements and having a dedicated role for requirements engineering activities. 'A good requirements practice can either reduce cost of the development project or increases quality of the resulting product' [6]. Good requirements practices ensure that the product quality is under control and it helps in keeping project within the schedule [5] [7]. Project success can be achieved by applying these well-established practices.

There are different sources of good RE practices in literature. Sommerville et al [5] presented a set of 66 RE practices. By using these guidelines/practices practitioners can improve the software development process and can gain business benefits. These guidelines are divided into three major groups which are basic, intermediate and advanced guidelines. There are 36 basic

practices, 21 intermediate and 9 advanced practices. The basic practices are concerned with fundamental activities required to gain control of requirements engineering process, intermediate practices are concerned with the use of methodological approaches and tools and advanced practices are concerned with formal methods. These practices are divided into 8 categories which include documentation, elicitation, analysis and negotiation, describing requirements, modeling, validation, management and practices for critical systems. Similarly, other researchers like Davis [6], Wiegers [7], Young [8] and Robertson [9] presented some good RE practices.

1.2. Existing Surveys for Valuable RE Practices

Many researchers attempted to find out the RE practices that are most valuable according to the RE practitioners, and can benefit the software organizations. Primary research studies in this field include a comprehensive survey [10] conducted with practitioners of ten Australian software development organizations in 2008. The purpose of this survey was to identify relative perceived values of RE practices from seven key areas of requirements engineering. The results of this survey presented some high value RE practices. These are making a business case for the project, assessing the systems feasibility, defining system boundaries, specify requirements quantitatively, define standard templates for requirements, and use a data dictionary, change management process and propose test cases. Another empirical study [11] was conducted in 2012 to identify RE practices which are important for GSD projects. As compared to the previous study [10] this study included responses from the broad range of RE experts worldwide by conducting an online survey. This research study identified six high value practices. These practices mainly focus on GSD project stakeholders, scope, standards and requirements traceability management. Another study [12] surveyed outsourcing experts from 18 software development organizations. The aim of this study was to identify significant RE practices for outsourcing projects. The RE practices from six key areas of requirements engineering are included in this study. The results identified 43 RE practices significant for outsourcing projects.

Kassab et al [13] presented a research survey in 2014 with primary objective of identifying changes in requirement elicitation, analysis, modeling and verification practices over the last decade. The results of this survey indicated upsurge of agile methodologies over traditional

SDLCs in last decade. Decline in the trends of object orientation, scenarios and UML diagrams. Increased use of natural language to express requirements.

1.3. Inadequacy of Existing Research in the Field of RE Practices

Most of these primary studies [10] [11] [12] in the field of RE practices were conducted in the specific domains such as global software development or outsourcing. In addition most of these studies were conducted in only one part of the world with a low sample size. Similarly, almost all of the studies do not include all eight categories of RE practices. For example, the study conducted by K. Cox et al [10] interviewed practitioners from only ten software development organizations in Australia. The study conducted by Niazi et al [11] identified valuable RE practices in GSD context. The study conducted by Iqbal et al [12] included RE practices for outsourcing projects and the study conducted in 2014 by Kassab et al [13] included only four areas of RE.

Hence, the pertinent question is: What are the most valuable practices related to different areas of RE on which the RE practitioners from all over the world have a common consensus? If we find such RE practices we can generalize them and use them as the basis for the development of good RE processes. Our research is in line with the purpose of finding these valuable RE practices. We have conducted a global survey to find out valuable RE practices. Secondly we developed an RE approach based on these practices for agile projects. In the following paragraphs, we will give details of our research.

1.4. Motivation

There are two main factors behind the motivation for this research. In the following paragraphs, we will provide details of each factor.

1.4.1. The Cost Factor

The software development industry is characterized by rapid innovation and intense competition. To survive this competition, the software industry must develop high quality software on time and within pre-defined budget. It is a common untrue assumption that performing RE practices may increase the development cost. That is the reason why IT management hesitates to perform

2008 survey. Majority of the respondents reported the use of natural language to express requirements.

Small and very small software firms have some characteristics that discriminate them from medium and large size companies these include project size, staff quality, resources and project management etc. Hence the requirement engineering practices which are suitable for medium and large sized firms not necessarily suitable for small firms. Quispe et al conducted a study [13] to identify state of the RE practices in very small firms of Chile. The data was collected from twenty four experienced project managers through survey and focus group. The findings showed that these firms are facing challenges like lack of communication between customers and companies which results in imperfect requirement specification, dissatisfaction and scope creep.

Adam et al [148] conducted several case studies in small and medium sized enterprises (SMEs) of German software industry. These studies aimed at understanding current RE capabilities of the companies and to suggest improvements. The RaqMan approach was applied in the case companies to assess their RE process capability. RaqMan is a process improvement approach based on RE practices presented by the authors [147] previously. Some RE practices are identified by the participants as working well in industries these are model user behavior, view based documentation, model domain, prototyping, determine the scope, and determine the feasibility.

Daneva et al [149] conducted a focus group study to validate thirteen RE practices for ERP projects which they have presented in an earlier study [150]. The participants were ERP architects of organizations. The participants identified the practices which they have used or seen in their real life. As a result, twelve out of thirteen RE practices were selected by the participants as used by them in real life ERP projects.

2.1.1. Literature Analysis

Most of the previous surveys used either interviews or questionnaires or a mixture of both. Most of the studies [10] [13] [144] [145] [146] [148] are performed in only one part/country of the world. Hence, the findings cannot be widely generalized to practitioners from other countries.

SDLCs in last decade. Decline in the trends of object orientation, scenarios and UML diagrams. Increased use of natural language to express requirements.

1.3. Inadequacy of Existing Research in the Field of RE Practices

Most of these primary studies [10] [11] [12] in the field of RE practices were conducted in the specific domains such as global software development or outsourcing. In addition most of these studies were conducted in only one part of the world with a low sample size. Similarly, almost all of the studies do not include all eight categories of RE practices. For example, the study conducted by K. Cox et al [10] interviewed practitioners from only ten software development organizations in Australia. The study conducted by Niazi et al [11] identified valuable RE practices in GSD context. The study conducted by Iqbal et al [12] included RE practices for outsourcing projects and the study conducted in 2014 by Kassab et al [13] included only four areas of RE.

Hence, the pertinent question is: What are the most valuable practices related to different areas of RE on which the RE practitioners from all over the world have a common consensus? If we find such RE practices we can generalize them and use them as the basis for the development of good RE processes. Our research is in line with the purpose of finding these valuable RE practices. We have conducted a global survey to find out valuable RE practices. Secondly we developed an RE approach based on these practices for agile projects. In the following paragraphs, we will give details of our research.

1.4. Motivation

There are two main factors behind the motivation for this research. In the following paragraphs, we will provide details of each factor.

1.4.1. The Cost Factor

The software development industry is characterized by rapid innovation and intense competition. To survive this competition, the software industry must develop high quality software on time and within pre-defined budget. It is a common untrue assumption that performing RE practices may increase the development cost. That is the reason why IT management hesitates to perform

development [17]. With the rise of agile methodologies in the last decade, the software projects completed within budget and on time increased from 29% to 39%. Figure 2 shows the project resolution results by research conducted by the Standish group for years 2004-2012. According to CHAOS report agile is one of the success factors especially in small projects.

	2004	2006	2008	2010	2012
Successful	29%	35%	32%	37%	39%
Failed	18%	19%	24%	21%	18%
Challenged	53%	46%	44%	42%	43%

Figure 2: Project resolution results

(From CHAOS MANIFESTO 2013, http://www.versionone.com/assets/img/files/CHAOSManifesto2013.pdf)

However, CHAOS report also states that the rate of specified requirements completed within time dropped from 84% in 2010 to 69% in 2012. So there is a need to focus on requirements related issues. Secondly the main objective of agile is to do only most required activities which consume fewer resources and produce the maximum return. Hence, it will be appropriate if we introduce only most valuable and most required RE activities in agile.

Our research is also motivated by XP (eXtreme Programming). XP is a light weighted agile method. XP focuses on 12 core practices e.g. test driven development, pair programming, continues integration etc. XP emphasizes to use these valuable practices at extreme levels to get maximum benefits. For example, testing is a valuable practice so in XP testing is performed throughout the development process. Test cases are developed before writing the code, continues acceptance testing is performed through customer's feedback. Informal reviews are performed through pair programming where programming is always done by pairs of two programmers, one writes the code and the other observes. XP is very successful process that creates working software faster and with very few defects.

Similarly, we want to identify valuable practices in RE and want to develop an RE approach by using these practices at their extreme. This approach will also provide many benefits and overcome challenges of RE in agile.

1.5. Research Questions

We explored three questions to evaluate the value of RE practices, to check the differences and commonalities of the opinion of different groups of RE practitioners and to check the correlation between different categories of RE practices.

RQ1: *Which requirements engineering practices are perceived as having high values by requirements engineering practitioners?*

The question aims at identifying a set of RE practices that are perceived as having high value according to the RE experts worldwide according to a predefined criteria.

RQ2: *Does any difference exist between high value requirements practices with varied characteristics of participants (with respect to experience level, company type, company size, Region, and the development process type)?*

This question aims at assessing the similarities or differences among different groups of participants based on their varied characteristics. For example experience level is one grouping type here we want to check that the participants with different levels of experience (junior, intermediate and senior) have the same opinion about the perceived value of RE practices?

RQ3: *Is there any co-relation among different categories of RE practices?*

This question aims to evaluate the relationships among different categories (requirements documentation, elicitation, analysis and negotiation, describing requirements, system modelling, validation, management, and requirements for critical systems) of RE practices.

1.6. Expected Outcomes

- A list of RE practices which are perceived as most valuable by RE practitioners all over the world.
- Differences and commonalities among opinion of different groups of RE practitioners about perceived value of RE practices.
- Correlation of different categories of RE practices based on participants responses.
- An RE approach "XRE" based on guideline for RE practitioners in agile.

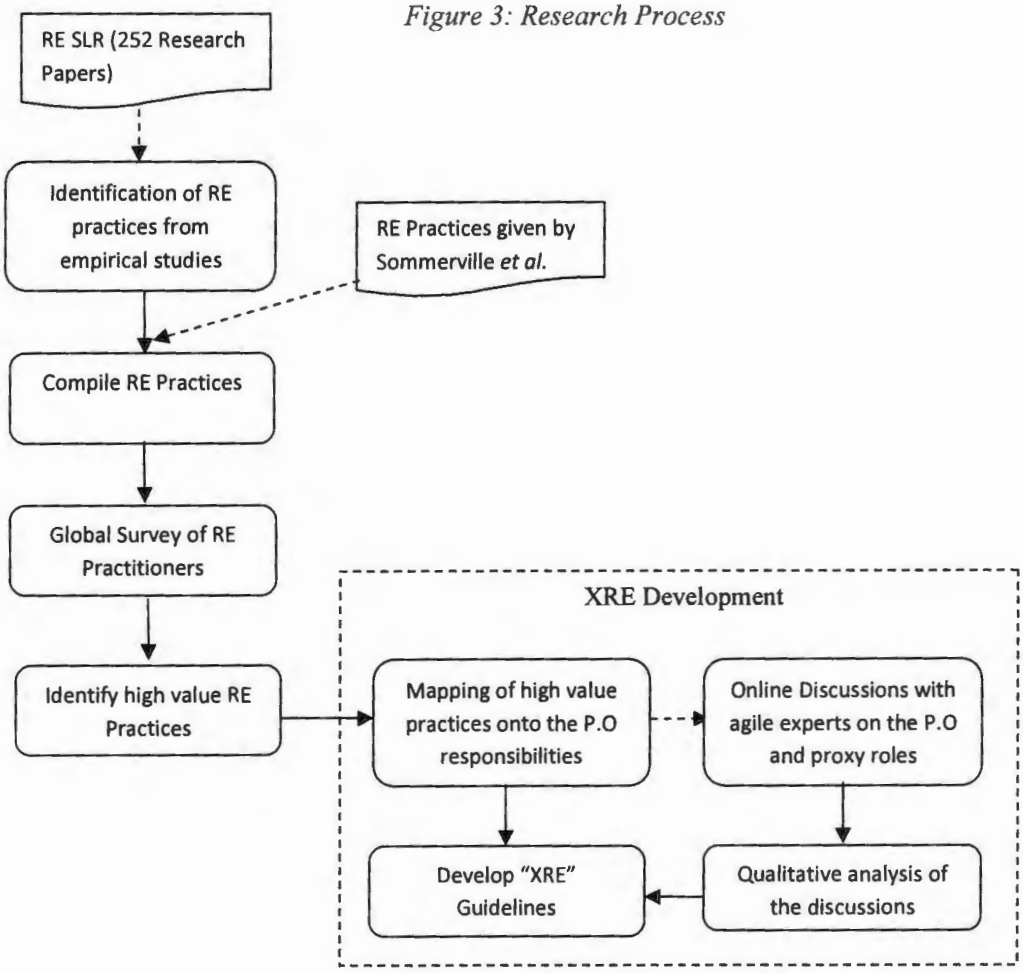
1.7. Proposed Solution

The proposed solution complement the agile methods Scrum and Extreme Programming (XP) with guidelines for the customer representative/product owner. The implementation of proposed guidelines ensure the extreme use of high value RE practices identified as a result of the survey. The solution is designed in two stages the first stage consists of reviewing the existing methods of Scrum and XP. Second stage consists of mapping the RE practices into customer representative/P.O responsibilities, providing the guidelines and providing their implementation information.

1.8. Research Process

In order to answer the research questions, the adopted research method consisted of following steps.

Figure 3: Research Process



- Literature review to understand the concepts of requirements engineering practices and to find up-to-date requirements engineering practices to be included in the questionnaire for the assessment.
- Designing of questionnaire and pilot testing to check its feasibility.
- Sampling by using different sampling techniques and inviting the target audience to participate in the survey.
- Survey execution
- Results analysis by applying different statistical techniques
- Development of RE approach XRE
- Concluding the research and propose future work

Following research methods were used to perform the tasks:

- **Research Type:** Investigative
- **Data collection method:** We used the survey method to collect data. Surveys have different methods like Structured Interviews or Questionnaires. We used an online survey questionnaire to elicit data.
- **Source of Data:** The data was collected from practitioners of software development organizations in different countries of the world. The type of the targeted population was in-house software development organization and the organizations that produce the software for external parties. The organizations were small, medium or large.
- **Sampling Method:** Convenience Sampling, Snowball Sampling
- **Data Analysis Method:** We used Statistical analysis techniques Relative frequencies, Chi-square (Linear by linear association) and correlation (Pearson) for data analysis.

The details of the research process are given in chapter 3. Research methodology.

1.9. Thesis outline

Remaining of the thesis was organized as follows:

Chapter#2: This chapter Contains literature review related to requirements engineering practices and the previous studies that were conducted in different parts of the world in this field. This chapter further gives brief introduction to different areas of RE and the research in those areas. It also provides in-sight into recent trends in the field and the progress in last decade. Second part covered the width of literature on agile processes and cultivates the background for the proposed

SDLCs in last decade. Decline in the trends of object orientation, scenarios and UML diagrams. Increased use of natural language to express requirements.

1.3. Inadequacy of Existing Research in the Field of RE Practices

Most of these primary studies [10] [11] [12] in the field of RE practices were conducted in the specific domains such as global software development or outsourcing. In addition most of these studies were conducted in only one part of the world with a low sample size. Similarly, almost all of the studies do not include all eight categories of RE practices. For example, the study conducted by K. Cox et al [10] interviewed practitioners from only ten software development organizations in Australia. The study conducted by Niazi et al [11] identified valuable RE practices in GSD context. The study conducted by Iqbal et al [12] included RE practices for outsourcing projects and the study conducted in 2014 by Kassab et al [13] included only four areas of RE.

Hence, the pertinent question is: What are the most valuable practices related to different areas of RE on which the RE practitioners from all over the world have a common consensus? If we find such RE practices we can generalize them and use them as the basis for the development of good RE processes. Our research is in line with the purpose of finding these valuable RE practices. We have conducted a global survey to find out valuable RE practices. Secondly we developed an RE approach based on these practices for agile projects. In the following paragraphs, we will give details of our research.

1.4. Motivation

There are two main factors behind the motivation for this research. In the following paragraphs, we will provide details of each factor.

1.4.1. The Cost Factor

The software development industry is characterized by rapid innovation and intense competition. To survive this competition, the software industry must develop high quality software on time and within pre-defined budget. It is a common untrue assumption that performing RE practices may increase the development cost. That is the reason why IT management hesitates to perform

RE activities. On the contrary, the literature says that by ignoring RE can increase the cost even up to two hundred (200) time [14] [15] [16].

Figure.1. shows ignoring errors in requirements stage increases the cost of development with each successive stage and if we do not correct them until the production stage it will double the development cost. To overcome this challenge, there is a need to focus on RE process. Hence identifying valuable RE practices can help to solve this problem. Because instead of wasting money on all RE practices the practitioners can adopt only some most valuable well-established practices, which will prevent requirements errors and increase the return on investment (ROI) on the project.

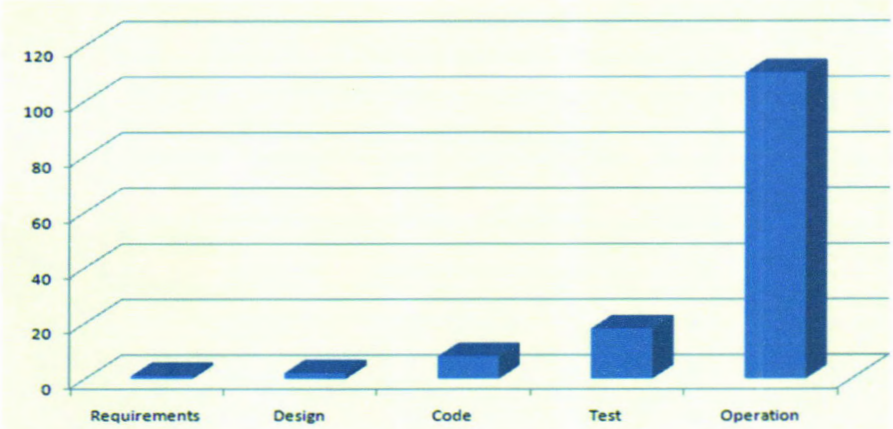


Figure 1: Cost of Correcting Requirement Errors

(From Motivation for Improving Software Development Practices, 2014. http://objectivesoftwaresolutions.com/content/motivation-improving-software-development-practices)

1.4.2. Upsurge of Agile and Extreme Programming

Agile software development has emerged in the last decade as new and different way of software development as compared to the traditional one. All agile methods including Scrum and eXtreme programming (XP) follow the core principles that are part of the agile manifesto. These principles include distinct move towards collaborative development, minimizing n-necessary work especially documentation, active participation of customers/stakeholders and iterative

development [17]. With the rise of agile methodologies in the last decade, the software projects completed within budget and on time increased from 29% to 39%. Figure 2 shows the project resolution results by research conducted by the Standish group for years 2004-2012. According to CHAOS report agile is one of the success factors especially in small projects.

	2004	2006	2008	2010	2012
Successful	29%	35%	32%	37%	39%
Failed	18%	19%	24%	21%	18%
Challenged	53%	46%	44%	42%	43%

Figure 2: Project resolution results

(From CHAOS MANIFESTO 2013, http://www.versionone.com/assets/img/files/CHAOSManifesto2013.pdf)

However, CHAOS report also states that the rate of specified requirements completed within time dropped from 84% in 2010 to 69% in 2012. So there is a need to focus on requirements related issues. Secondly the main objective of agile is to do only most required activities which consume fewer resources and produce the maximum return. Hence, it will be appropriate if we introduce only most valuable and most required RE activities in agile.

Our research is also motivated by XP (eXtreme Programming). XP is a light weighted agile method. XP focuses on 12 core practices e.g. test driven development, pair programming, continues integration etc. XP emphasizes to use these valuable practices at extreme levels to get maximum benefits. For example, testing is a valuable practice so in XP testing is performed throughout the development process. Test cases are developed before writing the code, continues acceptance testing is performed through customer's feedback. Informal reviews are performed through pair programming where programming is always done by pairs of two programmers, one writes the code and the other observes. XP is very successful process that creates working software faster and with very few defects.

Similarly, we want to identify valuable practices in RE and want to develop an RE approach by using these practices at their extreme. This approach will also provide many benefits and overcome challenges of RE in agile.

1.5. Research Questions

We explored three questions to evaluate the value of RE practices, to check the differences and commonalities of the opinion of different groups of RE practitioners and to check the correlation between different categories of RE practices.

RQ1: *Which requirements engineering practices are perceived as having high values by requirements engineering practitioners?*

The question aims at identifying a set of RE practices that are perceived as having high value according to the RE experts worldwide according to a predefined criteria.

RQ2: *Does any difference exist between high value requirements practices with varied characteristics of participants (with respect to experience level, company type, company size, Region, and the development process type)?*

This question aims at assessing the similarities or differences among different groups of participants based on their varied characteristics. For example experience level is one grouping type here we want to check that the participants with different levels of experience (junior, intermediate and senior) have the same opinion about the perceived value of RE practices?

RQ3: *Is there any co-relation among different categories of RE practices?*

This question aims to evaluate the relationships among different categories (requirements documentation, elicitation, analysis and negotiation, describing requirements, system modelling, validation, management, and requirements for critical systems) of RE practices.

1.6. Expected Outcomes

- A list of RE practices which are perceived as most valuable by RE practitioners all over the world.
- Differences and commonalities among opinion of different groups of RE practitioners about perceived value of RE practices.
- Correlation of different categories of RE practices based on participants responses.
- An RE approach "XRE" based on guideline for RE practitioners in agile.

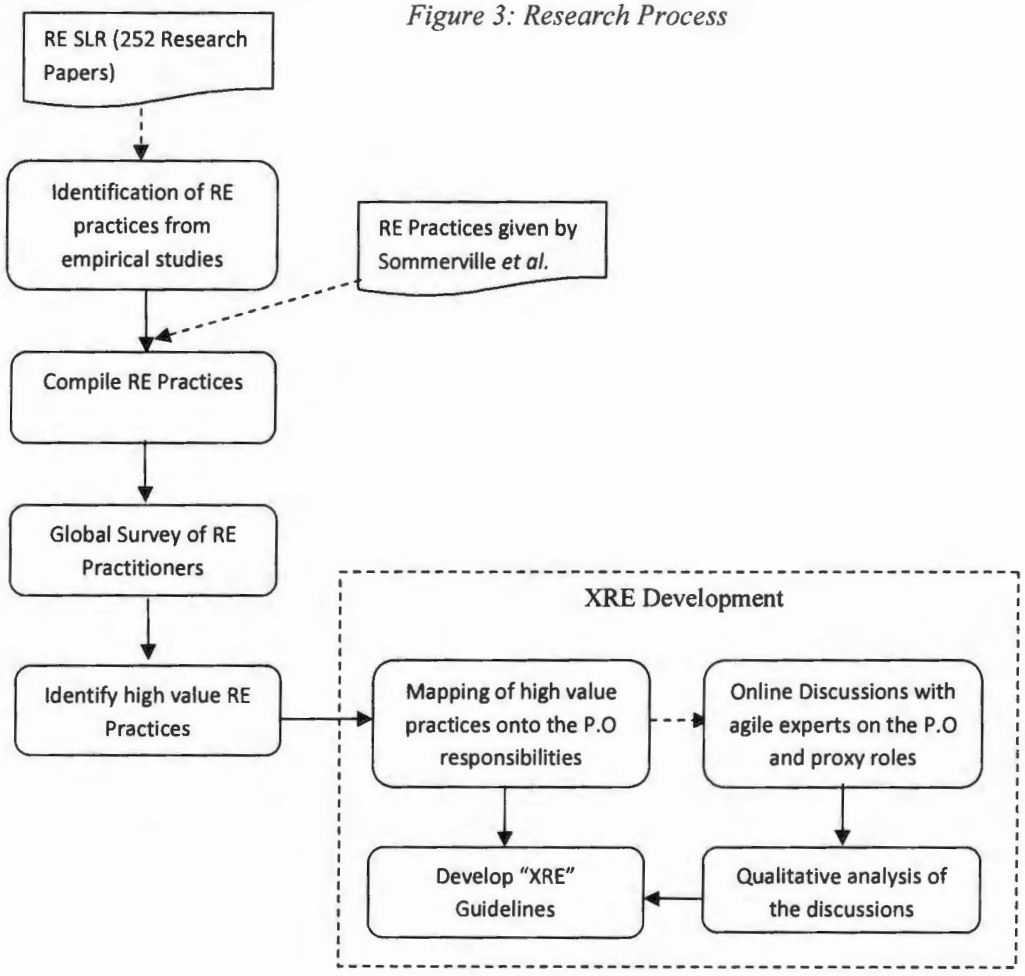
1.7. Proposed Solution

The proposed solution complement the agile methods Scrum and Extreme Programming (XP) with guidelines for the customer representative/product owner. The implementation of proposed guidelines ensure the extreme use of high value RE practices identified as a result of the survey. The solution is designed in two stages the first stage consists of reviewing the existing methods of Scrum and XP. Second stage consists of mapping the RE practices into customer representative/P.O responsibilities, providing the guidelines and providing their implementation information.

1.8. Research Process

In order to answer the research questions, the adopted research method consisted of following steps.

Figure 3: Research Process



- Literature review to understand the concepts of requirements engineering practices and to find up-to-date requirements engineering practices to be included in the questionnaire for the assessment.
- Designing of questionnaire and pilot testing to check its feasibility.
- Sampling by using different sampling techniques and inviting the target audience to participate in the survey.
- Survey execution
- Results analysis by applying different statistical techniques
- Development of RE approach XRE
- Concluding the research and propose future work

Following research methods were used to perform the tasks:

- **Research Type:** Investigative
- **Data collection method:** We used the survey method to collect data. Surveys have different methods like Structured Interviews or Questionnaires. We used an online survey questionnaire to elicit data.
- **Source of Data:** The data was collected from practitioners of software development organizations in different countries of the world. The type of the targeted population was in-house software development organization and the organizations that produce the software for external parties. The organizations were small, medium or large.
- **Sampling Method:** Convenience Sampling, Snowball Sampling
- **Data Analysis Method:** We used Statistical analysis techniques Relative frequencies, Chi-square (Linear by linear association) and correlation (Pearson) for data analysis.

The details of the research process are given in chapter 3. Research methodology.

1.9. Thesis outline

Remaining of the thesis was organized as follows:

Chapter#2: This chapter Contains literature review related to requirements engineering practices and the previous studies that were conducted in different parts of the world in this field. This chapter further gives brief introduction to different areas of RE and the research in those areas. It also provides in-sight into recent trends in the field and the progress in last decade. Second part covered the width of literature on agile processes and cultivates the background for the proposed

RE approach.

Chapter#3: This chapter describes the research methodology. It provides brief objective of the research and significance of the research. Its state the research questions of the thesis and gives their explanation. It provides the details about the sampling techniques like snowball sampling that are used to select target population.

It tells the readers how the questionnaire was designed how many categories of questions and which requirements engineering practices were included in the questionnaire. It reports the pilot testing of the Questionnaire and provides details of the techniques used for data analysis. It also gives description on the execution of the survey.

Chapter#4: This chapter contains data analysis and interpretation after conducting the survey.

Chapter#5: This chapter contains detailed description of the presented RE approach. And also provides details about how the approach can be implemented in industrial setting.

Chapter#6: In this chapter the conclusion of research has been provided. The contributions of this research work were discussed in an impartial way. This chapter provides summary, the research contribution and future work of the proposed research work.

1.10. Summary

In chapter 1 we have provided brief introduction of our dissertation. Section 1.1 explains the RE domain and valuable practices concept. Section 1.2 provides summary of existing research in this field. Section 1.3 explains the inadequacy of the existing research to find out valuable RE practices. The existing surveys though provide us several valuable RE practices however these practices are not generalizable because almost all of these survey are conducted in only one part of the world, with low sample size or conducted in specific domain. Section 1.4 provides motivation for the dissertation which consists of three factors first factor is related to cost of software development and its link with RE, second one is related to agile methods and third one is related to existing research and the need to conduct a survey for RE practices. Section 1.5 provides the research questions explored in the study. Section 1.6 demonstrates the expected outcomes of the study. Section 1.7 gave overview of the proposed solution. Section 1.8 describes the research process used in the dissertation. Section 1.9 contains overall structure of the dissertation and provides introduction of different chapter for the readers.

CHAPTER 02
LITERATURE REVIEW

2. Literature Review

This chapter presents the review of relevant literature in the field of RE practices. The purpose of review was to gather broader understanding of research in different areas of RE specially related to identification of valuable RE practices. Following three questions broadly outline scope of the literature review.

Q1: What are the primary studies performed to identify valuable RE practices and what are their limitations?

Q2: What are the up-to-date (1997-2011) RE practices?

Q3: What are the focus points in recent (2011-2014) research in different areas of RE?

Section 2.1 to 2.4 provide the analysis of questions 1-3 respectively.

2.1. Primary Research Studies to Identify Valuable RE Practices

Several research studies have been conducted in different countries to investigate the state of the RE practices. In this section, we are summarizing these primary research studies in the area of RE practices. Tahir et al [144] conducted an empirical study to identify most performed and least performed RE practices in Malaysian software industry. Twenty seven practitioners from software development organizations participated in the survey. They were asked to rank each RE practice according to its frequency of use. Results of the study indicated that most of the RE activities such as requirements prioritization, feasibility study, using software systems to manage requirements, having standard templates for requirements, and identifying nonfunctional requirements are mostly followed. While some other RE practices like conflict resolution, using interaction matrices, prototyping, organizing formal requirements, risk analysis, and including stakeholders in requirements validation are among fewer practiced activities.

Solemon et al [145] conducted a study to investigate the current state of RE problems and practices in 63 Malaysian based software firms. They included 82 practices from nine key areas of RE in the questionnaire. The respondents were asked to identify RE practices which they have taken part in recently. They discovered that most of the problems were requirements based problems rather than organizational problems. These problems include incomplete requirements, ambiguous requirements, and changing requirements.

Cox et al [10] conducted in-depth interviews with practitioners of ten Australian software development organizations. Based on their findings they identified several RE practices having high perceived values. These practices are making a business case for the project, assessing the systems feasibility, defining system boundaries, specify requirements quantitatively and define standard templates for requirements, use a data dictionary, change management process and propose test cases.

Talbot et al [146] looked at RE practices in software development industry of New Zealand. The research aim was to determine the factors that affect RE practice especially within the small and mediums sized companies. They collected the data from 30 practitioners through questionnaire and face-to-face interviews. Most of the results showed inconclusive findings however it also appeared that the issues and challenges faced by New Zealand industry are same as to other similar oversees companies.

Niazi et al [11] investigated relative perceived values of RE practices critical for GSD projects. The participants placed the degree of importance to each RE practice, based on their previous GSD projects experience. The study included responses from a wide range of RE experts worldwide by conducting an online survey. This research study identified eleven high value practices. These practices mostly focus on stakeholders, scope, standards and traceability of requirements in GSD projects.

Another study [12] surveyed outsourcing experts from 18 software development organizations to identify significant RE practices for outsourcing projects. The RE practices from six key areas of RE are included in this study. The results include 43 RE practices significant for outsourcing projects.

Kassab et al [13] presented a research survey about RE up-to-date practices. This survey tends to identify changes in requirement elicitation, analysis, modeling and verification practices over the last decade. They used an online questionnaire as data collection method. The results of this survey showed high trends in object oriented analysis and design while scenarios and UML are less common. Software quality assurance, prototyping and inspections showed no changes from

2008 survey. Majority of the respondents reported the use of natural language to express requirements.

Small and very small software firms have some characteristics that discriminate them from medium and large size companies these include project size, staff quality, resources and project management etc. Hence the requirement engineering practices which are suitable for medium and large sized firms not necessarily suitable for small firms. Quispe et al conducted a study [13] to identify state of the RE practices in very small firms of Chile. The data was collected from twenty four experienced project managers through survey and focus group. The findings showed that these firms are facing challenges like lack of communication between customers and companies which results in imperfect requirement specification, dissatisfaction and scope creep.

Adam et al [148] conducted several case studies in small and medium sized enterprises (SMEs) of German software industry. These studies aimed at understanding current RE capabilities of the companies and to suggest improvements. The RaqMan approach was applied in the case companies to assess their RE process capability. RaqMan is a process improvement approach based on RE practices presented by the authors [147] previously. Some RE practices are identified by the participants as working well in industries these are model user behavior, view based documentation, model domain, prototyping, determine the scope, and determine the feasibility.

Daneva et al [149] conducted a focus group study to validate thirteen RE practices for ERP projects which they have presented in an earlier study [150]. The participants were ERP architects of organizations. The participants identified the practices which they have used or seen in their real life. As a result, twelve out of thirteen RE practices were selected by the participants as used by them in real life ERP projects.

2.1.1. Literature Analysis

Most of the previous surveys used either interviews or questionnaires or a mixture of both. Most of the studies [10] [13] [144] [145] [146] [148] are performed in only one part/country of the world. Hence, the findings cannot be widely generalized to practitioners from other countries.

The number of respondents and the size of participating organizations also vary from study to study. The focus of research studies varies considerably. [10] focused on RE practices for GSD projects, [11] on outsourcing projects while [150] validated RE practices that are related to ERP projects. The key areas of RE that are included in studies also vary in number and type, for example [145] included 80 practices from nine areas of RE while [10] examined practices from 6 areas of RE. The findings also vary especially the findings relating to the number of resulting RE practices, as well as value of RE practices. Some of the findings were common such as some RE practices that are perceived high value in multiple studies. However the results are not generalizable because of variable factors as mentioned above.

2.2. SLR of Up-to-Date (1997-2012) RE Practices

The literature indicates that available sources for RE practices are mostly outdated for example, Sommerville and Sawyer presented RE good practices [5] in 1997. RE practices have changed significantly and new practices have been adopted in last decade. In this section, we presented up-to-date RE practices which are identified from current (1997-2012) evidence based literature. Evidence based software engineering (EBSE) improves decision making related to software development and maintenance by incorporating the current evidence from literature with practical experiences and human values [152]. EBSE tries to close the gap between theory and practice by giving emphasis on methodological rigor while focusing on relevance for practice [153]. We went through total two hundred and fifty two (252) research studies published during 1997-2012. We took this literature from the authors of an SLR "Evidence in Software Engineering: A Systematic Literature Review" [151] conducted in 2012. We reviewed these articles to identify RE practices. Table 2.1 shows the RE practices that we identified from these studies. From the identified 57 practices only 6 practices were cited in more than one studies. Detail is given in chapter 3.

NO.	Requirements Engineering Area	Requirements Engineering Practice	Research Studies
1.	Analysis and negotiation	Use goal-oriented requirement engineering (GORE) techniques for requirement elicitation and negotiation.	[18][19] [20][21] [22][23]
2.	Elicitation	Facilitate requirement elicitation with domain ontology.	[24][25] [26][27]
3.	Describing requirements	Have direct contact with customer to avoid unclear requirements.	[28][29] [30]
4.	Management	Have a dedicated role for requirement engineering activities	[28][29]
5.	Elicitation	Use different techniques for requirement gathering like interviews, document analysis, use cases, scenarios, prototyping, brainstorming and focus groups.	[28][31]
6.	Analysis and negotiation	Use automatic tools in requirement analysis process.	[32][33]
7.	Elicitation	Use Appreciative Inquiry in requirement gathering	[34]
8.	Management	Use formal, informal or semiformal representations for requirements management	[35]
9.	Elicitation	Do design-driven requirement elicitation for web-based system's development	[36]
10.	Specification	Use clone detection techniques to remove redundancy from software requirement specifications.	[37]
11.	Management	Use risk analysis techniques to deal with risks in safety critical systems.	[32]
12.	Management	Use Automated Similarity Analysis technique to identify requirements duplicates and `interdependencies.	[38]
13.	Specification	Explicit definition of requirement process and documentation.	[29]
14.	Management	Focus on the quality of software requirements to ensure success of software projects.	[39]

15.	Management	Make requirement engineers familiar with Requirement Error Taxonomy to reduce the likely hood of errors they commit while developing requirement documents.	[40]
16.	Management	Use sustainability taxonomy with goal oriented techniques.	[41]
17.	Modelling	Use goal-modelling for requirement modelling activities.	[42]
18.	Specification	Use task-oriented requirement engineering [TORE] for assuring more completeness and correctness of requirement specifications.	[43]
19.	Specification	Include UML diagrams in requirements specification document.	[44]
20.	Management	Develop context requirements	[45]
21.	Management	Develop context-product requirements	[45]
22.	Management	Manage requirements in context	[45]
23.	Management	Monitor and evolve customer requirements	[45]
24.	Management	Monitor and evolve context requirements.	[45]
25.	Management	Monitor product satisfaction of requirements	[45]
26.	Management	Manage architectural requirements	[45]
27.	Management	Specify product line requirements	[45]
28.	Management	Analyze support of evolutionary in architectural requirements.	[45]
29.	Management	Manage design complexity in requirements	[45]
30.	Management	Identify requirements that contribute to increased design complexity	[45]
31.	Analysis and negotiation	Analyze requirements to achieve balance between design complexity and customer satisfaction	[45]

32.	Elicitation	For requirement elicitation and negotiation use transition packages, promote training within organization or use outside consultants	[32]
33.	Specification	Improve quality of natural-language requirement documents by using style guides	[32]
34.	Management	Develop a stakeholder viewpoint	[40]
35.	Management	Include team member satisfaction in the project success factor	[40]
36.	Management	Build the team vision collaboratively	[40]
37.	Management	Use human facilitator in integrated, rich communication media during decision making	[40]
38.	Management	Build consensus on formal operating norms for meetings, deadlines and commitments	[40]
39.	Management	Facilitate communication session to allow everyone to speak	[40]
40.	Specification	Share requirement specification templates	[40]
41.	Management	Establish technology accessibility and compatibility for all teams	[40]
42.	Prioritization	Use distributed quality function deployment for requirement prioritization	[40]
43.	Specification	Create a proper project structure clearly showing the value of dependency of each activity and artefact	[40]
44.	Management	Adopt a standard way to work	[40]
45.	Management	Maintain and share a project artefacts repository	[40]
46.	Management	Establish requirements awareness system, outlining people's roles and responsibilities	[40]
47.	Management	Make sure to establish the requirement engineer as single intermediate between the customer and the development team	[46]

48.	Analysis and negotiation	Use automated requirement analysis techniques for early requirement analysis	[47]
49.	Management	Use automated tools to support the collaboration in requirement management process	[48]
50.	Modelling	Intensive use of models in RE activities	[49]
51.	Management	Define the real (actual) requirements	[44]
52.	Analysis and negotiation	Apply group recommendation technologies in the context of requirement negotiation	[50]
53.	Specification	Use context diagrams during high level design and specification of requirements	[21]
54.	Analysis and negotiation	Gradual and iterative detailing of requirements	[51]
55.	Analysis and negotiation	Identify risks related to each requirement in the form of risk profiles	[52]
56.	Prioritization	Prioritize requirement risks on the basis of values given by success critical stack holders	[52]
57.	Management	Identify success critical stakeholders	[52]

Table 2.1 Summary of RE practices identified from current literature

2.3. Focus Points in Recent (2011-2014) RE Research

In response to the third question, we reviewed recent articles and journals related to different areas of RE. The RE process has different phases like elicitation, modeling, prioritization, specification and management. We have reviewed the recent research papers related to each phase and highlighted the focus points about these areas. The purpose of doing this review is to gain knowledge about current practices in different areas of RE.

Focus Points in Requirements Elicitation Research: Selection of requirements elicitation techniques is one of the focus points in recent research. There are several research techniques but most often interviews technique is used. The interview topics must be selected in such a way that they ensure covering all the needed information. Burney et al [53] presented an elicitation topic

map (ETM) in 2014 that helps the practitioners in preparing interviews. An empirical study was conducted by Hadar et al [54] in 2014 to examine the perceived and actual effects of domain knowledge on elicitation through interviews. They found several positive and negative effects.

The use of domain ontologies in requirements elicitation is also evident in current literature [55] [56] [57]. The use of only interview technique for requirements elicitation is because of less methodological guidance about other techniques of elicitation. Carrizo et al [58] presented a framework for selection of elicitation methods in 2014. This framework provides a wide variety of elicitation techniques and information on their use to requirements engineers. Similarly, Wellsandt et al [59] compared eight elicitation techniques through a qualitative criteria related to embedded sensors in 2014.

Tiwari et al [60] presented a framework which uses project contextual knowledge to select elicitation techniques. Research is done on tools and techniques that can be used to automate elicitation process.

Meth et al [61] conducted a systematic literature review in 2013. The aim of this SLR was to capture the current state of automated requirements elicitation. This SLR included 36 research studies. They analyzed these studies according to tool categories, evolution approaches, and technological concepts. They also highlighted the future research needs in the area of automated requirements elicitation.

During the elicitation process, ways of communication between the development team and the customers are established. The inadequate customer involvement can cause problems in clarifying and gathering requirements, prioritizing requirements, getting feedback, loss of productivity and business loss [59]. A systematic literature review of stakeholder's identification (SI) methods in requirements elicitation was conducted by Pacheco et al [62] in 2012. 47 research studies were included in this SLR. It addressed three questions. First question was related to methods that are used to carry out the SI in requirements elicitation. The result indicated that the methods used for SI are few and not well structured. Second question was related to effective practices used for SI. The results identified three effective practices. The third

question was related to consequences of incorrect SI. The results indicated that incorrect SI can lead to requirements that do not correspond to real world needs.

Focus Points in Requirements Modeling Research: The focus of requirements modeling research in recent years is mainly on modeling approaches, methods, and techniques. In goal based modeling business objectives, associated tasks and resources are captured [63] [64]. Another approach is to model business processes using UML diagrams. Michel et al [67] aimed at providing evidence related to the effectiveness of UML modeling and proved that it is not a time consuming task but actual time is spend developing and communicating the design. Bider et al [65] presented a modeling technique in 2014 for business processes to elicit their requirements. Schneider et al [66] presented a modeling language URML. The purpose of this language is to capture danger modeling, feature modeling and goal modeling.

Lee et al [68] presented a goal driven feature modeling approach in 2013. This approach separates the feature space into the problem space and the solution space features and creates mapping between them. Wnuk et al [69] presented a modeling framework iMORE for requirements artifacts in large- scale, market driven requirements context. This framework distinguishes between internal and external information structures.

There is the need for empirical evaluation of these modeling approaches. Abrahao et al [70] presented a method for evaluating the quality of requirements modeling methods based on user perception. This method consists of two parts. First part is based on the theoretical model that explains different quality dimensions of modelling methods. The second part consist of a practical instrument to measure these dimensions. They also evaluated the model and found it suitable for assessing requirements modeling methods.

Similarly Goknil et al [71] presented a metamodeling approach in 2013 for reasoning about requirements and their relation to models expressed in different modeling approaches. Amokrane et al [72] provided analysis of a set of modeling languages, frameworks and methods for small and medium size organizations. They analyzed them according to their accessibility and verification techniques. Badreddin et al [73] explored the reasons behind limited use of

modeling practices in open sources software development community. They also highlighted the characteristics of modeling tools that will encourage their use.

Focus Points in Requirements Prioritization Research: Requirements prioritization is the process of identifying requirements and ordering them according to their importance [74]. Several requirement prioritization techniques have been discussed in recent literature. Some techniques that got focus of the recent research during 2011 to 2014 include analytical hierarchy process which helps in setting priorities and making best decisions [75-85]. Binary search tree is also used for requirements prioritization [76] [79] [82] [83] [86].

In cost value approach the cost of implementing each requirement and the relative value of each is calculated [67] [66] [79] [84] [87]. In numerical assignment approach each requirement is given a symbol according to its perceived value and these values are used to compare requirements [76] [78] [79] [82] [87]. Planning game is another most frequently used approach for requirements prioritization these days. This is a meeting that occurs once each iteration in XP where requirements are prioritized by customer representative [75] [78] [79] [81] [83] [84].

In cumulative voting approach multiple stakeholders are asked to prioritize requirements through a ratio scale [76-78] [82] [84] [86] [87]. The data obtained from this voting is useful in finding correlation between requirements and issues related to them.

Achimugu et al [88] conducted a systematic literature review of requirements prioritization research in 2014 and found that the prioritization techniques suffer from multiple limitations these include dependency issues, lack of scalability, coordination among stakeholders, and methods of dealing with rank updates during requirements evolution. Saranya et al [89] analyzed different techniques in 2014 that are used for prioritization of non-functional requirements.

They found NFR algorithm as most suitable methodology for prioritization of NFR. Khari et al [90] compared six prioritization techniques in 2013 by applying them a set of quality requirements and found value oriented prioritization (VOP) as best method to prioritize software requirements.

CHAPTER: 03

RESEARCH DESIGN & EXECUTION

3. Research Methodology

There is widespread interest in empirical approaches for software engineering research these days. The empirical approaches emphasize on what people do or can do in practice instead of what is possible, in theory [93]. Empirical methods are vital for software engineering because through these methods we can include human behavior into the research approach taken [94]. Empirical research has different methods like experiments, case studies, surveys and post-mortem analysis. These methods are actually not competing, on the contrary, different methods can be used together to obtain more sources of information [94]. The selection of these methods depends on the available resources, access to the subjects, opportunity to control the variables of interest and the skills of the researcher [95].

We used survey research method to collect data. "Survey research is used to identify the characteristics of broad population of individuals. It is mostly closely related to the use of questionnaires for data collection" [95]. Survey is an effective data collection method for software engineering projects especially when we talk about identifying current practices [96]. Through survey large amount of data can be collected in relatively short period of time, surveys are less expensive as compared to other data collection techniques and surveys are easy to administer [97]. Through survey we can send the questionnaire to a large number of people. We can cover a large population by taking a sample which is representative of the population. We analyze the data and draw conclusions. Then these conclusions are generalized to the population from which the sample was taken.

The major challenge faced by the survey research is to control the sampling bias [95]. Sampling bias causes problems in generalizing the survey results. For example, the respondents may not be the true representative of the actual population.

Low response rate also causes bias in surveys. According to Singer et al [98] the response rate for software engineering surveys is almost 5%. Another challenge faced by surveys is to ensure the questions are designed in a way to get useful and valid data. Another problem is that it is possible that what people say they do in survey response they may not actually do it because they do not introspect reliability on their work practices [95].

The core activities of the survey are discussed below.

3.1. Conception:

First activity of the survey is to formulate the research questions that we have to answer. In this thesis, we have to evaluate the following questions;

RQ1: *Which requirements engineering practices are perceived as having high values by requirements engineering practitioners?*

The question aims at identifying a set of requirements engineering practices that are perceived as having high value according to the requirements engineering experts worldwide.

RQ2: *Does any difference exists between high value requirements practices with varied characteristics of participants (with respect to experience level, company type, company size, Region, and the development process type)?*

This question aims at assessing the similarities or differences among different groups of participants based on their varied characteristics. For example experience level is one grouping type here we want to check that the participants with different levels of experience (junior, intermediate and senior) have the same opinion about the perceived value of RE practices?

RQ3: *Is there any co-relation among different categories of RE practices?*

This question aims to evaluate the relationships among different categories (requirements documentation, elicitation, analysis and negotiation, describing requirements, system modelling, validation, management, and requirements for critical systems) of RE practices.

3.2. Select Sample

We define our target population as practitioners from different software development organizations that are involved in requirements engineering activities. For example requirements engineers, requirements analysts, Project managers/Team lead or business analysts. The eligibility criteria for target population was,

1. Requirements Engineers working in the industry
2. Team Lead / Project Managers

3. Business Analysts
4. Requirements Analysts

We decided to use convenience sampling. We could not use representative list because for this we would have required a complete list of all RE professional from all over the world from which we could randomly select sample but no such list exists. We used two ways to gain access to our RE professionals around the globe.

3.2.1. Snowball Sampling

We used snowball sampling as one of the means to access participants. "Snowball sampling is a non-probability sampling technique that is used by researchers to identify potential subjects in studies where subjects are hard to locate". [99] In snowball sampling, we first identify the initial subject than we request the subject to help us identify people with similar properties. We sent emails to RE experts by using our personal contacts. We asked them to assist us in our research and forward the survey links to their contacts with having similar characteristics as required by our research. We included the survey link to the emails.

3.2.2. LinkedIn Groups

Secondly we sent requests through emails to members of LinkedIn groups. Table 4 presents the list and details of these groups. We posted the request to participate in our survey and survey link on these groups as well.

Sr. no	Group	Members
1.	Requirements Engineering Specialist Group (RESG)	1,361
2.	Computer & Software Engineering Professionals	72,445
3.	Software Engineering Professionals	11,383
4.	Requirements Engineering	18,551
5.	Reuse in Requirements Engineering	54
6.	IREB certified professionals for Requirements Engineering (CPRE)	1,179
7.	Software Development Professionals Group	19,963
8.	Software Engineering	16,591
9.	Software Professionals Network	1,540

10.	Software Professionals Group	4,413
11.	Requirement Engineering	1,773
12.	Direct Client Requirements in U.S.A	9,197
13.	Global Software Engineering	1,556

Table 3.1 Summary of requirement engineering related groups on LinkedIn

3.3. Determine the Research Method

We draw upon more established work on RE practices [14] and [16] to devise our research approach and choose appropriate research method. As our target population was dispersed around the globe we carried out an online questionnaire technique. We used a web based survey tool SurveyGizmo (http://www.surveygizmo.com).

3.4. Design the Questionnaire

All the questions included in the questionnaire are closed ended. However the respondents can add any RE practice which is not included in the questionnaire but about which they have the perception of being important. The questionnaire can be found in Appendix H.

3.4.1. Questions Categories

The questionnaire has three categories of questions. First category is related to the particulars of respondents e.g. what is the experience level of the participant or position in the company etc. This category included 5 questions. Second category deals with the demographics of the respondent's company e.g. location, primary business, size and scope etc. This category included 5 questions. The remaining questionnaire consists of RE practices for which the participants have to choose an importance level.

3.4.2. RE Practices Included in Questionnaire

Initially, we included only 66 practices presented by Sommerville and Sawyer in their book "Requirements Engineering: A Good Practice Guide" [5]. These practices are divided into three major groups. There are 36 basic practices which deal with fundamental activities to gain control of the RE process, 21 intermediate practices which deal with the use of methodological

approaches and tools, and 9 advanced practices which are related to methods such as formal specification. These practices were divided into 8 major categories.

1. Requirements documentation: practices relating to structuring and organizing the requirements documents.
2. Requirements elicitation: practices to help discover the requirements from stakeholders, application domain, and organizational environments.
3. Requirements analysis and negotiation: practices to help identify and resolve incompatibilities and missing information problems.
4. Describing requirements: practices for effectively writing requirements.
5. System modeling: practices for the development of models in order to better understand requirements.
6. Requirements validation: practices to help establish formal validation procedures relating to incompleteness, inconsistency or incompatibility problems.
7. Requirements management: practices for requirements management.
8. Requirements for critical systems: practices particularly useful for critical systems.

This book was published in 1997 and it does not include the practices which were introduced and adopted by the software development companies and professionals in last decade. Therefore, we decided to go through the evidence based literature published from 1997 to 2012 to identify some more up to date practices. From this literature, we identified 57 new RE practices. Table.2.1 shows these practices. The practices which were referred in more than one research papers were considered to be added to the survey. These practices were incorporated into 8 existing categories [5]. We included these 6 practices in the survey so our survey included total 72 RE practices.

3.4.3. Perceived Value

We used the concept of perceived value to assess the relative importance of practices. "Perceived value is the extent to which a practice is used in the organization because it is perceived by practitioners to bring benefits to the organization" [10] [11]. A four point scale is used to assess the relative perceived values of RE practices. If majority of the participants (>=50%) consider a practice as having high value it is considered as a high value RE practice.

Types of assessments are given below.

- **High Perceived Benefits (H):** The practice that have a standard document structure and is always followed as a part of the software development process
- **Medium Perceived Benefits (M):** The practice is widely followed in the organization but it is not mandatory
- **Low Perceived Benefits (L):** Some project managers introduced this practice only for particular projects
- **Zero Perceived Benefits (Z):** This practice is rarely or never used in organization.

3.5. Pilot Testing of Questionnaire

In a quantitative study like ours the survey instrument (questionnaire in our case) needs to be validated to check its effectiveness and to check whether the questions are appropriate to elicit the right information which can answer the research questions. We conducted a pilot study with two RE experts. After the pilot, study we finalized the questionnaire.

3.6. Data analysis Techniques and Tools

Data analysis is the process of systematically applying statistical techniques on data with the purpose of describe, illustrate and evaluate data [100]. There are many statistical data analysis tools and techniques available. The selection of these tools and techniques depends on the type of data we are going to analyze. We used statistical analysis technique for data analysis.

3.6.1. Relative Frequencies

RQ1: Which requirements engineering practices are perceived as having high values by requirements engineering practitioners?

In order to find out perceived value of a practice, the occurrence of perceived benefit is counted from the responses. The count of individuals giving a particular value to a practice is called frequency of the value of that practice. For example, if 20 participants gave high value to practice X we will say the high value of practice X has a frequency=20. Secondly we calculated the relative frequencies. The proportion of individuals having the quality is called the relative

frequency or proportional frequency [101]. For example, if we have total 60 participants than the relative frequency of high value of practice X is 33.3. The practices which are defined as having high perceived value by the majority of participants (≥ 50%) are considered as most valuable practices.

3.6.2. Chi-Square (linear by linear association)

RQ2: Does any difference exist between high value requirements practices with varied characteristics of participants (with respect to experience level, company type, company size, Region, and the development process type)?

We used linear-by-linear chi square test to identify the difference between different groups of participants as done by similar previous studies [11]. This test is most suitable to our data where sample size is large and data is in the form of cross tabulation.
"The chi-square test is a statistical test that can be used to determine whether observed frequencies are significantly different from expected frequencies" [102].

Through chi-square tests, we can compare observed and expected frequencies quantitatively as it is not possible to decide just by looking at them that there is enough difference between them or not. Statistical significance, in this case, implies that the differences are not due to chance alone, but they also represent other processes at work. In chi-square test we begin by stating a null hypothesis. For example in our study, we stated the following hypothesis for the group based on experience level,

H_0: There is no significant difference between the perceived value of practice X between junior, intermediate and senior level participants

Here X can be any practice from 72 practices included in the questionnaire. And the alternative hypothesis is;

H1: There is the significant difference

Based on the result of the chi-square test we either reject the null hypothesis or fail to reject it. We used SPSS to perform the chi-square test. "We did not used Pearson chi-square instead we

used linear-by-linear association because our data is in ordinal form (where order of data maters).

3.6.3. Correlation

RQ3: Is there any co-relation among different categories of RE practices?

Co-relation is the statistical measure of how the value of two variables move in relation to each other. We needed to know whether participants' responses about one category of RE practices are related to their responses to other categories e.g. the participants who gave high values to documentation practices they also gave high values to elicitation practices or not? . To check this we applied Correlation test. There are several Correlation Coefficients we applied Pearson Correlation Coefficient as it is most commonly used.

The correlation between two variables is determined by the correlation coefficient. The correlation coefficient can be between -1 to +1. Perfect positive correlation (+1) means that if the value of one variable goes up or down the value of the other variable will go in lockstep in same direction. Perfect negative correlation (-1) means if the value of one variable goes up or down the value of other variable will move in the opposite direction. If the value of the coefficient is equal to 0 it means there is no relationship between the values of these two variables and they are completely random. We used SPSS to calculate the correlation coefficients.

3.7. Execution of the Survey

The survey was launched on 16 April 2014 and it remained open till 16 June 2014. In total, we got sixty eight (68) responses from RE professionals worldwide. From these participants fifty four (54) fully completed the survey while Fourteen (14) gave partial responses with only 11-15% data about perceived value of RE practices. We included the fifty four (54) complete responses in data analysis. The detailed results and analysis of data is given in chaper.4.

3.8. Summary

Chapter 3 reports the research methodology of the thesis. Section 3.1 provides brief objective of the research and section 3.2 explains the significance of the research. Section 3.3 explains in

detail the research methodology of the thesis. Its state the research questions of the thesis that are RQ1: Which requirements engineering practices are perceived as having high values by requirements engineering practitioners?, RQ2: Does any difference exist between high value requirements practices with varied characteristics of participants (with respect to experience level, company type, company size, Region, development process type)? And RQ3: Is there any co-relation among different categories of RE practices?

It provides the details about the sampling techniques like snowball sampling that are used to select target population. It tells the readers how the questionnaire was designed how many categories of questions and which requirements engineering practices were included in the questionnaire. It reports the pilot testing of the Questionnaire and provides details of the techniques used for data analysis. These statistical techniques include calculation of relative frequencies, chi-square test (linear by linear association) and co-relation test. Section 3.4 gives the brief description of the execution of the survey.

CHAPTER 4

DATA ANALYSIS AND INTERPRETATION

Section2 of survey deals with companies demographics. The results of this section with respect to primary business function of participant's companies' show that 53.7% companies are engaged in in-house software development while 38.9% are engaged in outsourcing development. With respect to number of employees, we divided participant's companies into three groups as done by Cox et al [11]. These groups are small with 0 to >= 19 employees, medium with 20 to >=199 employees and large with 200plus employees. According to this grouping 51.9% survey participants belong to large companies, 24.1% from medium and 18.5% from small companies. While 57.4% of these companies are multinational and 42.6% are national. Most of these companies are concerned with the development of data processing, real-time and embedded systems while some are concerned with safety critical systems.

Agile approaches have become very popular from the last decade. Agile is a software methodology which is initiated from practice to encourage collaboration between users and developers, to control rapid development cycles and to meet the changing business needs [103]. The effective requirement engineering process ensures fewer typical iterations to complete an agile software project [104]. The quality of the process can be gained through disciplined practices.

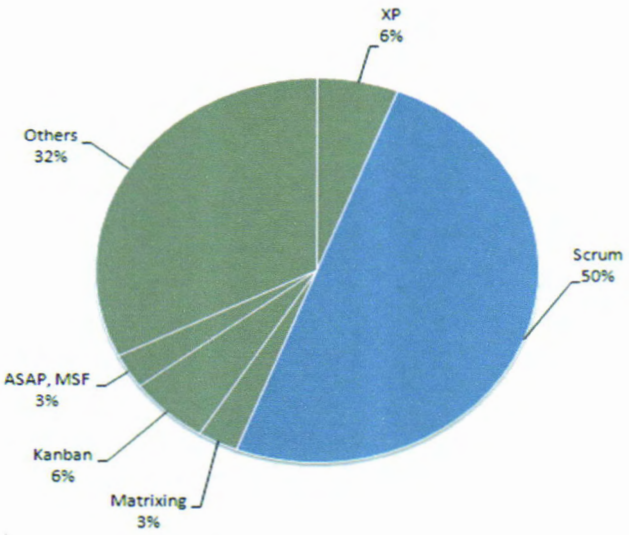


Figure 5: SDLCs employed by participants from agile community

With respect to the reported use of agile methodologies, survey results indicate that 57.4% of companies are following one or more agile methodologies while remaining 35.1% are following

traditional software development processes. Figure 5 shows software development life cycles employed by participants from the agile community. We can see that currently there is a high trend of agile methodology Scrum. It validates the findings of another empirical study [13] related to current trends of usage of agile methodologies. According to some participants agility is simply less formality so they use and trim different agile methodologies with respect to their contexts and project types. Hence, we can see a great ratio (32%) in others section.

4.1.1. Overall Analysis of High Value (>=50%) RE Practices

The analysis included responses from 54 professional that participated in our survey. These responses related to requirements engineering practices are presented in Appendix A. As we described the practices which are defined as having high perceived value by majority of participants (≥ 50%) are considered as most valuable practices. We identified six (6) high value practices. These practices are RD1 (Define a standard document structure), RE3 (Identify and consult system stakeholders), RA5 (Prioritize requirements), DR3 (Use diagrams appropriately), DR6 (Have direct contact with customers to avoid unclear requirements) and RM1 (Uniquely identify each requirement). Table 4.1 shows details of these practices.

ID	Practice (high benefits)	Frequency n=54	RE Category
RD1	Define a standard document structure	28	Documentation
RE3	Identify and consult system stakeholders	37	Elicitation
RA5	Prioritise requirements	31	Analysis & Negotiation
DR3	Use diagrams appropriately	28	Description
DR6	Have direct contact with customers to avoid unclear requirements	37	
RM1	Uniquely identify each requirement	33	Management

Table 4.1 high value (>=50%) requirements engineering practices

Two requirements engineering practices RE3 (Identify and consult system stakeholders) and DR6 (Have direct contact with customers to avoid unclear requirements) are most highly cited (68.5%) high value practices. According to Saiedian et al [105] relationship between developers and customers decide how well the development team can understand and meet needs of the users. The indirect link between customer and developers means that they do not communicate

directly, but there is some intermediate between them. The intermediates may not fully understand the customer's requirements and may alter them unintentionally. Additionally the customers are future maintainers of the system so if they participate directly in the development process they can understand the internal product issues and can make early change decision about requirements. Hence, the direct contact between customers and developers is necessary.

51.9% of our survey participants marked requirements documentation practice RD1 (Define a standard document structure) as having high value. It is important to note that the majority of our participants (57.4%) are from the agile community. Typically it is perceived that in agile requirements documentation is neglected. The reality is that in agile multiple small length documents are maintained. These include user stories, product backlog, sprint backlog and a set of roadmap documents like domain models, physical architectures, and logical architectures etc. To make these documents self-explanatory, it is very important to define a standard document structure.

Sixty one percent (61%) participants marked RM1 (Uniquely identify each requirement) as high value practice. According to Sommerville *et al* [5] a unique identifier should be assigned to each requirement. With this identifier, we can refer to this requirement in other parts of the document easily.

RA5 (prioritize requirements) is another highly cited (57.4%) practice. Requirements prioritization is very important in software projects because in most of the projects the timeline is short, resources are limited and the user's expectations are high. So it is necessary to ensure that the essential features are included in the product.

4.1.2. High Value RE Practices According to Participant's Experience Levels (Junior, Intermediate, and Senior)

As mentioned above we have divided the participants into three groups according to their experience levels. This categorization includes juniors (>5 years), intermediates (5-10 years) and seniors (>10 years). Figure 6 shows the summary of high value (>=50%) practices identified by these groups. The detailed analysis is given in appendix B.

RA5 (Prioritize requirements), DR6 (Have direct contact with customer to avoid unclear requirements) and RM1 (Uniquely identify each requirement) are perceived as high benefit (>50)

practices by junior, intermediate and senior experts. RE3 (Identify and consult system stakeholders) is perceived as high benefits practices by Intermediate and senior level experts.

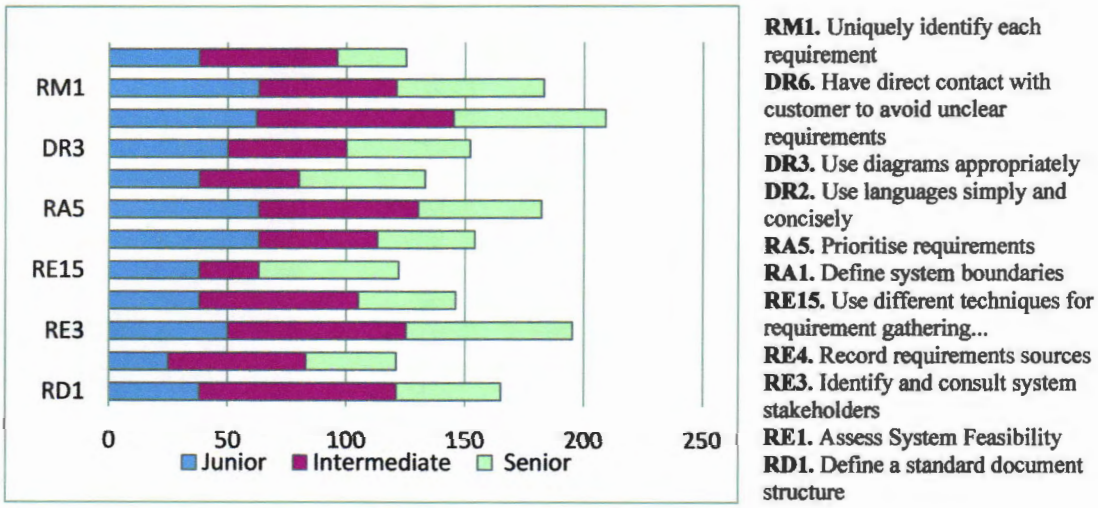


Figure 6: RE valuable practices based on practitioners' experience

RE3 (Identify and consult system stakeholders) is perceived as high benefits practices by Intermediate and senior level experts. RA1 (Define system boundaries) is perceived as high value practices by junior level participants. While RD1 (Define a standard document structure), RE1 (Assess System Feasibility), RE4 (Record requirements sources) and RM10 (Have a dedicated role for requirement engineering activities) are mostly cited as high value practices by intermediate level participants. RE15 (Use different techniques for requirement gathering), DR2 (Use languages simply and concisely) and DR3 (Use diagrams appropriately) are perceived as high benefits practices by senior level participants.

Our next step is to calculate the strength of association between these groups (junior, intermediate, senior) about perceived value of requirement engineering practices. We used Chi-Square (linear-by-linear association) test to check the significant difference.

Table 4.2 shows detailed application of chi-square test where we can see "P" value is greater than .05 for all practices. This proves that there is no significant difference between these groups about the perceived value of RE practices.

RE Category	Practice	Practitioners Experience (Total Frequency n=54)					
		Total High Value Frequency	Junior n=8	Intermediate n=12	Senior n=34	Chi-square Test (Linear-by-Linear Association) α = .05 df = 1	
						X^2	P
Documentation	RD1 Define a standard document structure	28	3	10	15	.167	.683
Elicitation	RE1 Assess System Feasibility	22	2	7	13	.719	.396
	RE3Identify and consult system stakeholders	37	4	9	24	.000	.993
	RE4 Record requirements sources	25	3	8	14	.777	.378
	RE15 Use different techniques for requirement gathering	26	3	3	20	.938	.333
Analysis & Negotiation	RA 5Prioritise requirements	31	5	8	18	.564	.453
	RA1 Define system boundaries	25	5	6	14	.776	.378
Description	DR2 Use languages simply and concisely	26	3	5	18	.178	.673
	DR3 Use diagrams appropriately	28	4	6	18	.228	.633
	DR6 Have direct contact with customer to avoid unclear requirements	37	5	10	22	.700	.403
Management	RM1 Uniquely identify each requirement	33	5	7	21	.187	.665
	RM10 Have a dedicated role for requirement engineering activities	20	3	7	10	.532	.466

Table 4.2. Application of chi-square test based on practitioners' experience

4.1.3. High Value RE Practices According to Participants w.r.t their Company Type (National, Multinational)

Multinational companies have locations in more than one country while national companies are limited to only one country. 57.4% of our study participants came from multinational companies while 42.6% came from national companies. Detailed responses of participants from different company types are given in Appendix D. Figure 7 shows valuable RE practices identified by practitioners from different company types.

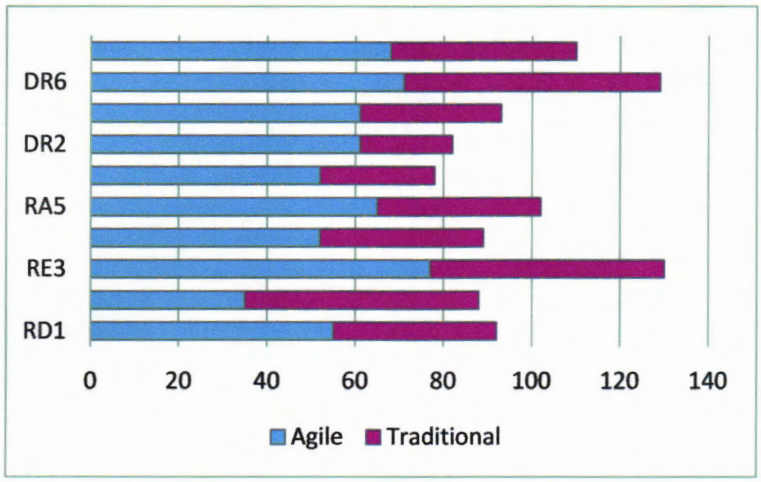
Elicitation	RE10 Prototype poorly understood requirements	19	7	4	8	0	1.327	.249
Elicitation	RE15 Use different techniques for requirement gathering	24	6	5	13	2	.049	.825
Analysis & Negotiation	RA 5Prioritise requirements	30	5	8	17	1	1.616	.204
	RA1 Define system boundaries	23	6	6	11	2	.026	.872
Description	DR3 Use diagrams appropriately	27	6	7	14	1	.021	.885
	DR6 Have direct contact with customer to avoid unclear requirements	34	7	9	18	3	.011	.918
Management	RM1 Uniquely identify each requirement	31	5	6	20	2	2.005	.157
Critical Systems	CS5 Cross-check operational and functional requirements against safety requirement	22	3	7	12	0	.168	.682

Table 4.4. *Application of chi-square test based on practitioners' company size*

4.1.5. High Value RE Practices Identified by Participants that are using Agile or Traditional SDLC

We divided our survey participants into two groups with respect to software development process they are currently following. These groups are agile and traditional. 57.4 % of survey participants reported use of the agile process, 31% reported use of traditional processes and 7% said they are not sure about the process their company is currently following. Figure 9 shows RE practices identified by these groups as having high values. Detailed analysis is given in Appendix E.

RE3 (Identify and consult system stakeholders) and DR6 (Have direct contact with customers to avoid unclear requirements) are commonly perceived high value (>50%) practices by participants following agile or traditional software development process.



RM1. Uniquely identify each requirement
DR6. Have direct contact with customer to avoid unclear
DR3. Use diagrams appropriately
DR2. Use languages simply and concisely **DR1.** Define standard templates for describing requirements
RA5. Prioritise requirements
RE15. Use different techniques for requirement gathering...
RE3. Identify and consult system stakeholders
RD3. Include a summary of the requirements
RD1. Define a standard document structure

Figure 9: RE valuable practices based on software development process

RD1 (Define a standard document structure), RE15 (Use different techniques for requirement gathering), RA5 (prioritize requirements), DR1 (Define standard templates for describing requirements), DR2 (Use languages simply and concisely), DR3 (Include a summary of the requirements), RM1 (Uniquely identify each requirement) are highly cited high value practices by agile practitioners only. And DR3 (Use diagrams appropriately) is perceived as high value practice by practitioners who follow traditional methods for software development.

Table 4.5 shows no significant difference was observed across practitioners following different software development processes (agile, traditional) about perceived value of RD1 (Define a standard document structure), RD3 (Include a summary of the requirements), DR1 (Define standard templates for describing requirements), and RM1 (Uniquely identify each requirement).

While significant difference was observed about perceived value of RE3 (Identify and consult system stakeholders), RE15 (Use different techniques for requirement gathering), RA5 (prioritize

requirements), DR2 (Use languages simply and concisely), DR3 (Use diagrams appropriately) and DR6 (Have direct contact with customer to avoid unclear requirements).

RE Category	Practice	Process Type (Frequency n=54)					
		Total High value frequency	Agile N=31	Traditional n=19	Not Know n=4	Chi-square Test (Linear-by-Linear Association) α = .05 df = 1	
						X^2	P
Documentation	RD1 Define a standard document structure	24	17	7	4	.669	.413
	RD3 Include a summary of the requirements	21	11	10	3	.697	.404
Elicitation	RE3Identify and consult system stakeholders	34	24	10	3	6.460	.011
	RE15 Use different techniques for requirement gathering	23	16	7	3	5.353	.021
Analysis & Negotiation	RA 5Prioritise requirements	31	20	7	4	7.192	.007
Description	DR1 Define standard templates for describing requirements	21	16	5	4	2.832	.092
Description	DR2 Use languages simply and concisely	24	19	5	2	12.906	.000
	DR3 Use diagrams appropriately	25	19	6	3	7.981	.005
	DR6 Have direct contact with customer to avoid unclear requirements	33	22	11	4	4.428	.035
Management	RM1 Uniquely identify each requirement	29	21	8	4	2.007	.157

Table 4.5. Application of chi-square test based on development process

4.1.6. High Value RE Practices Based on Participants' Region

Professionals from five regions (Asia, Europe, America, Australiana and Africa) participated in our survey. 35% participants are from Asia, 31% from Europe, 22% from America, 7% from Australiana and only 3% are from Africa. As the participation rate is very low from Australiana and Africa we excluded them from analysis and included only three regions in analysis. Figure

RE Category	Practice	Region (Frequency n=54)							
		Total High value Frequency	Asia N=19	Europe N=17	America N=12	Australiana N=4	Africa N=2	Chi-square Test (Linear-by-Linear Association) α = .05 df = 1	
								X^2	P
Documentation	RD1 Define a standard document structure	28	11	6	6	3	2	.39	.843
	RD5 Define specialized terms	23	6	6	7	2	2	1.673	.196
Elicitation	RE3Identify and consult system stakeholders	37	13	12	7	3	2	.074	.785
Analysis & Negotiation	RA 5Prioritise requirements	31	11	8	6	4	2	.034	.854
Describing Requirements	DR3 Use diagrams appropriately	28	9	9	7	2	1	.464	.496
	DR6 Have direct contact with customer to avoid unclear requirements	34	16	6	8	3	1	.652	.420
Management	RM1 Uniquely identify each requirement	33	10	7	11	4	1	3.358	.067

Table 4.6. *Application of chi-square test based on participants' region*

4.1.7. Correlation among Different Categories of RE Practices

We needed to know whether participants' responses to one category of RE practices are related to their responses to other categories. To check it out, we applied Correlation test.There are several Correlation Coefficients we applied Pearson Correlation Coefficient as it is most commonly used.

Table 4.7 shows statistical analysis of correlation among different categories of RE practices.

RE Category1	RE Category2	Correlation α = .05 n=54 P<.05	
		r	r2
Documentation	Modelling	.552	.30
Documentation	Elicitation	.748	.56
Documentation	Analysis & Negotiation	.650	.42
Documentation	Describing requirements	.647	.42
Documentation	Management	.549	.30
Documentation	Critical systems	.542	.29
Documentation	Validation	.526	.28
Modelling	Elicitation	.661	.44
Modelling	Analysis & Negotiation	.682	.47
Modelling	Describing requirements	.527	.28
Modelling	Management	.643	.41
Modelling	Critical systems	.505	.26
Modelling	Validation	.617	.38
Elicitation	Analysis & Negotiation	.805	.65
Elicitation	Describing requirements	.719	.52
Elicitation	Management	.686	.47
Elicitation	Critical systems	.527	.28
Elicitation	Validation	.607	.37
Analysis & Negotiation	Describing requirements	.623	.39
Analysis & Negotiation	Management	.630	.40
Analysis & Negotiation	Critical systems	.549	.30
Analysis & Negotiation	Validation	.732	.54
Describing Requirements	Management	.541	.29
Describing Requirements	Critical systems	.472	.22
Describing Requirements	Validation	.648	.42
Management	Critical systems	.508	.26
Management	Validation	.533	.28
Critical systems	Validation	.615	.38

Table 4.7. Correlation among different categories of RE practices

The coefficient of correlation is denoted by symbol 'r'. It ranges between +1 to -1. A correlation of 1 (positive or negative) is perfect correlation, 0 shows no relationship, .8 or .9 shows high correlation and .2 or .3 shows low correlation.

We can see relatively high correlation of elicitation practices with documentation (r=.748), analysis and negotiation (r=.805) and describing requirements (r=.719) practices. While analysis and negotiation category have a high correlation with validation practices (r=.732). Coefficient of determination is denoted by r2. It tells us how much of the variance in one of the variables is accounted for by the variance in the other variable.

4.2. Summary of Findings

In this study 54 requirements engineering experts provide their feedback about the importance of 72 requirements engineering practices. Different statistical techniques like chi-square (linear by linear association) tests are applied to these responses. The summary of finding from this statistical analysis is given below.

4.2.1. Which Requirements Engineering Practices are most valuable?

We consider an RE practice as most valuable only if 50% or greater number of participants rate it as having high value. From the results of our survey, we identified 6 practices that are perceived as high value practices by the majority of participants (>=50). These practices are RD1 (Define a standard document structure), RE3 (Identify and consult system stakeholders), RA5 (Prioritize requirements), DR3 (Use diagrams appropriately), DR6 (Have direct contact with customers to avoid unclear requirements) and RM1 (Uniquely identify each requirement).

Five of these practices are from Summerville's 66 practices while one practice DR6 (Have direct contact with customers to avoid unclear requirements) is from 6 practices which we have identified from current literature. This practice is the most commonly cited (68.5%) high value practice among all. It shows the great importance of customer involvement in software development process these days. Table 4.8 shows details of these practices.

RE Category	ID	Practice (high benefits)	Frequency n=54	Percentage %
Documentation	RD1	Define a standard document structure	28	51.8
Elicitation	RE3	Identify and consult system stakeholders	37	68.5
Analysis & Negotiation	RA5	Prioritise requirements	31	57.4
Description	DR3	Use diagrams appropriately	28	51.8
	DR6	Have direct contact with customers to avoid unclear requirements	37	68.5
Management	RM1	Uniquely identify each requirement	33	61.1

Table 4.8. Summary of most highly cited (>50%) RE practices

4.2.2. Do the valuable RE practices vary across different levels of experts?

Table 4.9 shows senior level experts ranked 7 RE practices as having high value. Intermediate level experts ranked 8 practices as having high value and junior level experts ranked 4 practices as having high value. Three RE practices are considered as having high value by all three expertise levels. These practices are DR5- Prioritize requirements, DR6-Have direct contact with customers to avoid unclear requirements and RM1-Uniquely identify each requirement.

Participant Types	Valuable Requirements Engineering Practices
Junior (n=8)	Following practices are identified as critical • RA1- Define system boundaries (63%) • **RA5- Prioritise requirements (63%)** • **DR6-Have direct contact with customer to avoid unclear requirements (63%)** • **RM1- Uniquely identify each requirement (63%)**
Intermediate (n=12)	Following practices are identified as critical • RD1- Define a standard document structure (83%) • RE1- Assess System Feasibility (58%) • RE3- Identify and consult system stakeholders (69%) • RE4- Record requirements sources (57%) • **RA5- Prioritise requirements (62%)** • **DR6-Have direct contact with customer to avoid unclear requirements (77%)** • **RM1- Uniquely identify each requirement (54%)** • RM10- Have a dedicated role for requirement engineering activities (54%)
Senior (n=34)	Following practices are identified as critical • RE3- Identify and consult system stakeholders (71%) • RE15- Use different techniques for requirement gathering. (59%) • **RA5- Prioritise requirements (53%)** • DR2- Use languages simply and concisely (53%) • DR3- Use diagrams appropriately (53%) • **DR6- Have direct contact with customer to avoid unclear requirements (65%)** • **RM1- Uniquely identify each requirement (62%)**

Table 4.9. *Summary of high value RE practices for different levels of experts*

One practice RE3-Identify and consult system stakeholders has very high frequencies in Intermediate and Senior level experts. It means that the intermediate and senior level expert know very well the importance of taking the key stakeholders onboard during the software

development process. The customer and other stakeholder's involvement is one of the key practices of agile software development as well. Another practice RD1-Define standards document structure have a very high frequency in intermediate level expert type.

4.2.3. Do the valuable RE practices vary across expert from different company sizes?

Table.4.10 shows the participants from large companies ranked 5 practices as having high values according to the criteria given above. The participants from medium companies ranked 7 RE practices as having high value. While the participants from small companies' ranked 8 practices as having high values.

Two practices RE3-Identify and consult system stakeholders and DR6-Have direct contact with customers are perceived as having high values by participants from all type of companies. Both of these practices are related to stakeholder's consultation. It also validates the fact that the customer's full involvement is focus of current software development processes.

Company Size	Valuable Requirements Engineering Practices
Small (n=10)	Following practices are identified as critical • RD3- Include a summary of the requirements (60%) • RD6- Make document layout readable (60%) • **RE3- Identify and consult system stakeholders (70%)** • RE10- Prototype poorly understood requirements (70%) • RE15- Use different techniques for requirement gathering like interviews, document analysis, use cases, scenarios, prototyping, brainstorming and focus groups. (60%) • RA1- Define system boundaries (60%) • DR3- Use diagrams appropriately (60%) • **DR6- Have direct contact with customer to avoid unclear requirements (70%)**
Medium (n=13)	Following practices are identified as critical • RD4- Make a business case for the system (53%) • RD6- Make document layout readable (53%) • **RE3- Identify and consult system stakeholders (53%)** • RA5- Prioritise requirements (62%) • DR3- Use diagrams appropriately (53%) • **DR6- Have direct contact with customer to avoid unclear requirements (69%)** • CS5- Cross-check operational/functional requirements against safety req. (53%) •

Large (n=28)	Following practices are identified as critical • RD1- Define a standard document structure (61%) • **RE3- Identify and consult system stakeholders (75%)** • RA5- Prioritise requirements (61%) • **DR6- Have direct contact with customer to avoid unclear requirements (64%)** • RM1- Uniquely identify each requirement (71%)

Table 4.10. Summary of high value RE practices for different company sizes

4.2.4. Do the valuable RE practices vary across expert from different company types?

Table 4.11 shows that the participants from national companies ranked 6 RE practice as having high value. While participants from multinational companies ranked 7 practices as having high value. Four practices are ranked as having high value by participants from both national and multinational companies these are RD1-Define a standard document structure, RE3-Identify and consult system stakeholders, RA5-Prioritize requirements and DR6-Have direct contact with customers to avoid unclear requirements. We can clearly see from table 4.11 that the participants from both national and multinational companies have a common consensus on most of the RE practices.

Company Type	Valuable Requirements Engineering Practices
Multinational(n=31)	Following practices are identified as critical • **RD1- Define a standard document structure (52%)** • **RE3- Identify and consult system stakeholders (65%)** • **RA5- Prioritise requirements (55%)** • DR2- Use languages simply and concisely (52%) • DR3- Use diagrams appropriately (55%) • **DR6- Have direct contact with customers to avoid unclear requirements (68%)** • RM1- Uniquely identify each requirement (68%)
National(n=23)	Following practices are identified as critical • **RD1- Define a standard document structure (52%)** • RD5- Define specialized terms (52%) • **RE3-Identify and consult system stakeholders (74%)** • RE15- Use different techniques for requirement gathering (57%) • **RA5- Prioritise requirements (61%)** • **DR6- Have direct contact with customers to avoid unclear requirements (70%)**

Table 4.11. *Summary of high value RE practices for different company types*

4.2.5. Do the valuable RE practices vary across expert practicing different software development processes?

Table 4.12 shows that the participants following traditional SDLC ranked 3 Re practices as having high values. While the practitioners following agile SDLC 9 practices as having high values. Two practices RE3-Identify and consult system stakeholders and DR6-Have direct contact with customers to avoid unclear requirements are perceived as having high benefits by participants from both agile and traditional groups. These two practices are also related to stakeholder's consultation.

Process Type	Valuable Requirements Engineering Practices
Agile(n=31)	Following practices are identified as critical • RD1- Define a standard document structure (23%) • **RE3- Identify and consult system stakeholders (77%)** • RE15- Use different techniques for requirement gathering (52%) • RA5- Prioritise requirements (65%) • DR1- Define standard templates for describing requirements (52%) • DR2- Use languages simply and concisely (61%) • DR3- Use diagrams appropriately (61%) • **DR6- Have direct contact with customers to avoid unclear requirements (71%)** • RM1- Uniquely identify each requirement (68%)
Traditional(n=19)	Following practices are identified as critical • RD3- Include a summary of the requirements (53%) • **RE3-Identify and consult system stakeholders (53%)** • **DR6- Have direct contact with customers to avoid unclear requirements (58%)**

Table 4.12. Summary of high value RE practices for different process types

4.2.6. Do the valuable RE practices vary across expert from different regions?

Table 4.13 shows that RE3-Identify and consult system stakeholders is perceived as having high value by participants from all regions (Asia, Europe, America, Australiana and Africa). RD1-Define a standard document structure and RA5-Prioritize requirements is perceived as having high value by participants from Asia, Australiana, and Africa. DR6-Have direct contact with

customers and RM1-Uniquely identify each requirement is perceived as having high value by participants from Asia, America, and Australiana regions.

Region	Valuable Requirements Engineering Practices
Asia(n=19)	Following practices are identified as critical • **RD1- Define a standard document structure (58%)** • **RE3- Identify and consult system stakeholders (68%)** • RA1- Define system boundaries (53%) • **RA5- Prioritise requirements (58%)** • **DR6- Have direct contact with customers to avoid unclear requirements (84%)** • **RM1- Uniquely identify each requirement (58%)**
Europe(n=17)	Following practices are identified as critical • **RE3- Identify and consult system stakeholders (71%)** • DR2- Use languages simply and concisely (53%) • DR3- Use diagrams appropriately (53%)
America(n=12)	Following practices are identified as critical • RD3- Include a summary of the requirements (58%) • RD5- Define specialized terms (58%) • RD6- Make document layout readable (58%) • **RE3- Identify and consult system stakeholders (58%)** • DR3- Use diagrams appropriately (58%) • **DR6- Have direct contact with customers to avoid unclear requirements (67%)** • RV3- Use multi-disciplinary teams to review requirements (67%) • **RM1- Uniquely identify each requirement (92%)** • CS1- Create safety requirement checklists (75%) • CS2- Involve external reviewers in the validation process (67%) • CS3- Identify and analyse hazards (75%) • CS4- Derive safety requirements from hazard analysis (83%) • CS5- Cross-check operational/ functional reqs against safety req (75%) • CS6- Specify systems using a formal specification (58%) • CS8-Learn from incident experience (58%) • CS9- Establish an organizational safety culture (58%)
Australiana(n=4)	Following practices are identified as critical • **RD1- Define a standard document structure (75%)** • RE1- Assess System Feasibility (75%) • **RE3- Identify and consult system stakeholders (75%)** • RE4- Record requirements sources (75%) • RE5- Define the system's operating environment (100%) • RE9- Collect requirements from multiple viewpoints (75%) • **RA5- Prioritise requirements (75%)** • DR1- Define standard templates for describing requirements (75%) • **DR6- Have direct contact with customers to avoid unclear requirements (75%)** • RV3- Use multi-disciplinary teams to review requirements (75%)

	• **RM1- Uniquely identify each requirement (100%)**
Africa(n=2)	Following practices are identified as critical • **RD1- Define a standard document structure (100%)** • RD5- Define specialized terms (100%) • RE2- Be sensitive to organisational and political consideration (100%) • **RE3- Identify and consult system stakeholders (100%)** • RE6- Use business concerns to drive requirements elicitation (100%) • RE7- Look for domain constraints (100%) • RE13- Reuse requirements (100%) • RE15- Use different techniques for requirement gathering. (100%) • **RA5- Prioritise requirements (100%)**

Table 4.13. Summary of high value RE practices for experts from different regions

4.3. Comparison of Results with Previous Studies

We compared the results of our survey with a similar previous study that was conducted by Niazi et al [11]. This study included practices from six areas of RE to identify their perceived value. They identified six high value practice for GSD projects according to the predefined criteria. Figure 11 shows results of the comparison between the high value RE practices identified by both studies. Detailed responses of participants from both studies are given in Appendix D.

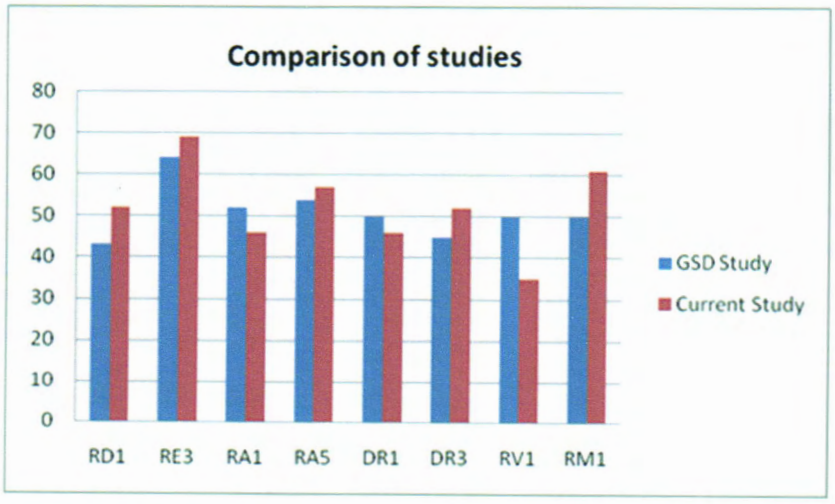


Figure11: Comparison of high perceived value RE practices in GSD and current study

RE3 (Identify and consult system stakeholders), RA5 (Priorities requirements) and RM1 (Uniquely identify each requirement) are commonly cited high value (>=50%) practices in both studies. RA1 (Define system boundaries), DR1 (Define standard templates for describing requirements) and RV1 (Check that the requirements document meets your standards) are commonly cited high value practices in GSD study. RD1 (Define a standard document structure) and DR3 (Use diagrams appropriately) are commonly cited high value practices in the current study.

RE Category	Practice	Comparison of Studies				
		Total High value frequency	GSD Study N=56	Current Study N=54	Chi-square Test (Linear-by-Linear Association) α = .05 df = 1	
					X^2	P
Documentation	RD1 Define a standard document structure	52	24	28	.907	.341
Elicitation	RE3Identify and consult system stakeholders	73	36	37	.014	.905
Analysis & Negotiation	RA1 Define system boundaries	54	29	25	.510	.475
	RA 5Prioritise requirements	61	30	31	.000	.922
Description	DR1 Define standard templates for describing requirements	53	28	25	1.669	.196
	DR3 Use diagrams appropriately	53	25	28	1.372	.241
Validation	RV1 Check that the requirements document meets your standards	47	28	19	1.739	.187
Management	RM1 Uniquely identify each requirement	61	28	33	.658	.417

Table 4.14. Comparison with GSD study through statistical analysis

Table 4.14 shows no significant difference was observed in practices identified by practitioners of GSD and the current study. As the value of p is higher than .05 in case of all practices.

CHAPTER 5

EXTREME REQUIREMENTS ENGINEERING (XRE)

5. Extreme Requirements Engineering (XRE)

In the previous chapter, we reported the results of our survey. The aim of the survey was to assess the perceived value of RE practices. The survey revealed six RE practices that were perceived as having high values according to the RE professionals worldwide. These practices are related to stakeholder's consultation, requirements specification, and requirements management. These three areas are thus the main focus of the solution. Instead of re-inventing the wheel we have complemented the existing most common agile software development methods SCRUM and Extreme programming (XP) with guidelines for agile roles to ensure the extreme use of these six high value practices.

The XRE was designed in two stages. In the first stage, we reviewed the existing methods of Scrum and XP. We looked at the work done by the inventive authors of SCRUM and XP as well as held discussions with practitioners to know the current state of the art through LinkedIn groups. Appendix I shows details of these discussions. We also reviewed the existing literature to know how RE practices are handled in agile.

In the second stage, we developed the guidelines for the roles that they have to implement in different stages of these methods to ensure the proper use of six high value RE practices. In the following paragraphs, we will firstly introduce the SCRUM and XP.

5.1. SCRUM

SCRUM is one of the most widely adopted agile method today [106]. SCRUM adopt an iterative and incremental approach to improving predictability and reduce the risk. In scrum the development is done in sprint that last from 1-4 weeks. At the beginning of each sprint, the team selects requirements from a prioritized list to be implemented in coming sprint. At each day of work the team gathers and reports on their work. At the end of the sprint, the team presents their work which is in the form of workable product increment. Three pillars of SCRUM are transparency, inspection and adoption [116]. Transparency means the process must be visible to those who are responsible for the outcomes. Inspection means that the stakeholders must be involved and inspect the process. And the adoption means that if something is going wrong and it seems that the resulting product will be unacceptable because of it then it should be removed.

SCRUM have two main artifacts product backlog and sprint backlog. The product backlog contains requirements of the product which are refined regularly and time to time new requirements are added. At the beginning of each sprint, some requirements from the product backlog are transferred to the sprint backlog that will be implemented in the sprint.

5.1.1 SCRUM Roles

The SCRUM team have three primary roles these are the product owner, scrum master, and the development team. The customer representative or product owner is responsible for defining the product's features, ordering them, reviewing them and accepting or rejecting the results [108]. The scrum master acts as the leader of the team and makes sure that the process is applied properly. The development team is responsible for the development of the working product.

5.1.2. SCRUM Events

The SCRUM itself is a container of the formal events that are performed in it. The SCRUM consists of sprint planning, daily stand-up meetings, development work, sprint review and the sprint retrospective. Figure 12. Shows the SCRUM events.

The sprint planning meeting is conducted to decide what goal will be achieved in this sprint and what work will be needed to perform to achieve this goal. The items are selected from the product backlog and sprint backlog is prepared. Development team, product owner, and scrum master usually participate in this meeting. Other willing stakeholders can also be invited.

The next event is daily standup meetings. The daily meetings are of 15 minutes usually and the teams synchronize their work in these meetings and plans the work that they will do in next 24 hours. The product owner, scrum master and the development team participates in it. But usually only the development teams speaks about their work. The sprint review meeting is performed at the end of each sprint to review the outcomes of the sprint and discuss the work done in the sprint. The product owner accepts or rejects the results of the sprint. The sprint retrospective meeting is conducted by SCRUM team to inspect itself and plan for improvements.

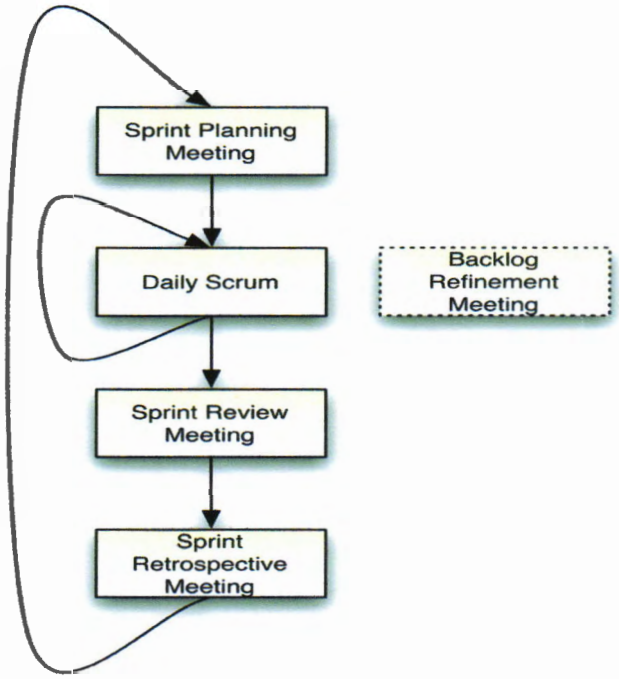


Figure 123: The Scrum events

(From Agile, 2015, http://www.eceltic.ie/our-work/agile/)

5.2. Extreme Programming (XP)

The extreme programming is a light weighted agile methodology as well as framework. It is an incremental and iterative process. XP consists of smaller iterations called releases. Each release in XP consists of couple of iterations and each iteration is at most three weeks [109]. The strength of XP process lies in its rules and practices. XP standard practices are collective code ownership, programming in pairs, continues integration, coding standards, on-site customer, simple design, refactoring, test driven development, planning game, system metaphor, sustainable pace and small releases.

5.2.1. XP Roles

XP have six primary roles. Customer representative, programmers, coach, tracker, testers, and consultant. Customer representative is an onsite customer, who is responsible for refining the user stories and driving functional tests etc. programmer is responsible for writing the code. The

coach and the tracker are responsible for the management of the project. The coach is responsible for technical managing the project while tracker tracks the systems success. The tester helps to choose the functional tests and run them. The consultant is hired to provide the knowledge of the process to team.

5.2.2. XP Events

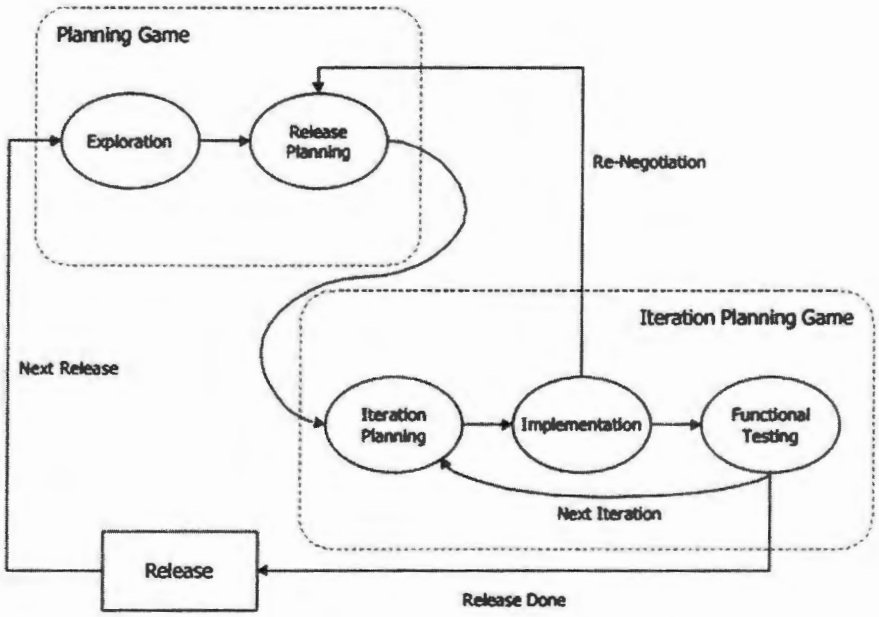


Figure 4: XP simplified process structure

(From eXtreme Programming an Overview, Dudziak, 2000, p.8)

Figure 13 shows the simplified XP process. The planning game consists of three stages exploration, release planning, and steering. In the exploration phase, user provides the user stories and the development team estimates the effort required to complete each user story. If the stories are too big then they are broken down into smaller stories and if they are too small they are combined with other stories. This phase takes from a couple of weeks to few months. Next is the planning phase where the commitment schedule meeting is conducted to determine the commitment schedule. Commitment schedule is the set of stories along with their estimates that are to be implemented in the current release [109]. After planning phase, the steering phase

comes. Steering mean starting the actual process by little moves. These moves include the iterations, recovery from wrong estimates, from wrong release date and adding new stories.

The iterations also consist of three phase's exploration, planning, and steering. In the exploration the customers select the stories from commitment schedule to be implemented in current iteration. And the developers break these stories into smaller tasks. In the planning phase for each task, a developer accepts the responsibility and estimates the ideal engineering time to complete the stories. In the steering phase, the actual coding is done and the fictional tests are run to check whether the completed stories work or not. Daily standup meeting is also conducted on the regular basis. At the end of each iteration, the functional test cases associated with stories are reviewed with the customers if they fail they are included in next iteration.

5.3. How RE Practices are handled in Agile and what are the Challenges Posed by Agile on RE

To answer this question we took help from a recent literature review [128] on RE practices and challenges in agile.

Requirements Engineering Practices in Agile: Minimal requirements documentation and more face-to-face communication is one aspect of agile software development [129]. Documentation is done in the form of user stories to avoid long and complex requirements documents. User stories enhance communication among stakeholders [130] and shift the focus from documentation to communication [133]. User stories eliminate the need of continues updating of requirements specification document [132].

Identification of stakeholders and frequent collaboration with an onsite customer to avoid disagreement and differencing of views on the variety of issues is a characteristic of agile methods [130] [131]. In agile methods, requirements emerge iteratively over time [134]. This iterative approach have many benefits like it makes the requirements clearer over time, reinforces the relationships with customers and allows changes in requirements in less time and cost [131].

Requirements prioritization is done on continues basis by customers mostly based on business value [131] or based on risk [130]. The dynamic nature of agile facilitates the change

management process. The main changes are to add up and drop features [131]. Cross-functional teams have members from different groups with similar goals. For example in agile designers, developers, testers and managers sit and work together. Challenges like communication gaps and over scoping of requirements can be overcome with this way of working [51].

Prototyping is used to review requirements and get feedback from customers. Prototyping is started with requirements that are simple, completely understood and have high priority [135]. In most agile methods like XP testing before coding approach is followed where test specifications are written before writing the actual code. This approach encourages early feedback in case a test fails.

Automated test-driven development is another emerging trend in this field that combines practices from both traditional and agile RE [136]. Requirements modeling is performed in agile but in a different way as compared to requirements modeling in traditional software development methods. One approach used for requirements modeling in agile methods is goal-sketching [128]. In goal sketching goal are refined for each iteration. This technique empowers the decision making in the negotiation process [137].

Requirements management is performed by maintaining the product backlog items and index cards [131] [134]. Developed requirements and remaining product backlog items are constantly reviewed in review meetings and through acceptance tests [133]. The acceptance tests are like unit tests that may result in "pass" or "fail" for a user story. These acceptance tests increase the collaboration between development team, domain experts, and customers and reduce the severity of defects [128]. The requirements analysis is performed by pairs that encourage the stakeholders to perform multiple roles as well [139]. Pairing practice for requirements analysis is one of the ways that closes communication gaps in agile teams [128]. Retrospective meetings are held after the completion of each iteration.

The new requirements are incorporated in the form of changes in deliverables by customers [140]. Instead of a static plan the agile teams do continues planning which helps in changing requirements even in later stages of the project [141].

Requirements Engineering Challenges in Agile: Minimal documentation is a challenge posed by agile methods [131] [134]. Instead of doing conventional requirements documentation agile teams do to-the-point, precise user story oriented documentation [133]. "In some cases especially in large projects where all team members are not co-located verbal communication is insufficient without proper documentation" [128]. To overcome this challenge, mostly user stories are complemented with more detailed artifacts [130] which helps in making right implementation choices in the coding stage.

Customer availability and access is another challenge for agile methods [134]. The customer unavailability is mostly because of certain factors like time, cost, and workload of customer representative [142]. This challenge is mostly overcome by having an on-site proxy customer [130] or moving a developer representative to the customer's site [142].

Budget and schedule planning is another challenge posed by agile methods. Upfront estimation is not possible when there are volatile requirements and planning [134]. "The cost is usually calculated based on user stories and new user stories may be included or some may be discarded in forthcoming iterations" [131].

The inappropriate architecture is another challenge where the architecture finalized by teams in early stage become inappropriate in later stages [134]. Another challenge which can cause massive rework is neglecting non-functional requirements [143]. The agile methods are flexible and allow change, but it can create troubles incorporating this much change and evaluating the consequences of change [128].

5.4. Designing the Solution

The high value RE practices which we have identified through survey are have direct contact with customers to avoid unclear requirements, identify and consult system stakeholders, have a standard document structure, use diagrams appropriately, prioritize requirements and uniquely identify each requirement. Hence, the primary focus of the solution is on three areas of RE stakeholder's collaboration, requirements specifications, and requirements management. Customer collaboration is one of the main feature and success factors of the agile projects [109-114].

Each agile team has an on-site customer representative who is chosen by the stakeholders to make decisions on their behalf. In SCRUM, this representative is called the product owner (P.O). He is usually responsible for prioritizing the requirements, accepting or rejecting the results and optimizing the value of the product in both XP and SCRUM projects. As most of the RE activities like requirements prioritization, requirements specification activities like enforcing the role to implement standard document structure are actually performed by the customer representative/P.O in agile projects. Hence, our solution revolves around the role of the customer representative/P.O. To acquire deep knowledge of the standard responsibilities of the P.O we studied the work by inventive authors of SCRUM and XP. We developed the list of responsibilities of P.O and then we mapped the six high value RE practices onto these responsibilities. Following paragraphs provide detail of these two steps.

5.4.1 Standard Responsibilities of Customer Representative/ P.O

Responsibilities of P.O in SCRUM: According to Jeff Sutherland and Ken Schwaber "The Product Owner is responsible for taking all the inputs into what the product should be from the customer or end-user of the product, as well as from Team Members and stakeholders and translating them into a product vision" [115]. Following are the main responsibilities of the P.O,

- The P.O defines the features of the product and decides on release date and content
- The P.O is responsible for the profitability/ROI of the product
- He prioritizes the product features/requirements according to the business value
- Can change the priorities of the items every thirty days
- He is responsible for accepting or rejecting the results

Jeff and Ken provided the following list of P.O responsibilities w.r.t different events in "The SCRUM papers" [115] in 2009 and "The SCRUM guide" [116] in 2011. In sprint planning meeting, the product owner reviews the product backlog and discusses the goal and context of the items with the scrum team. During the sprint he is mostly responsible for the break down the bigger items in smaller ones, clarify items and verify the tradeoffs. He attends the daily standup meeting. In the sprint review meeting the completed code is demonstrated to the P.O he accepts or reject it. And the remaining items on the product backlog are reprioritized again. Throughout the project, he is responsible for communication with stakeholders and manage their needs.

Responsibilities of customer representative in XP: According to Beck [117] following are the main responsibilities of customer representative in XP.

- He decides the scope and timing of the release
- Customers write the functional tests for the stories
- Collaborate with stakeholders, sits with team full time
- He picks the next release stories and next iteration stories

5.4.2. Mapping the RE practices into customer representative/ P.O responsibilities

In this section firstly we combined the responsibilities of customer representative in XP and P.O in SCRUM, simplified them and then mapped the six valuable RE practices onto these responsibilities. Figure 14 shows this mapping.

RE3: identify and consult system stakeholders is mapped on two responsibilities R1: collaborate with customers and stakeholders and R2: sits with the team on the daily basis. As both of these responsibilities include collaboration with stakeholders which are the development team and other stakeholders like other customers, users, suppliers, and testers etc.

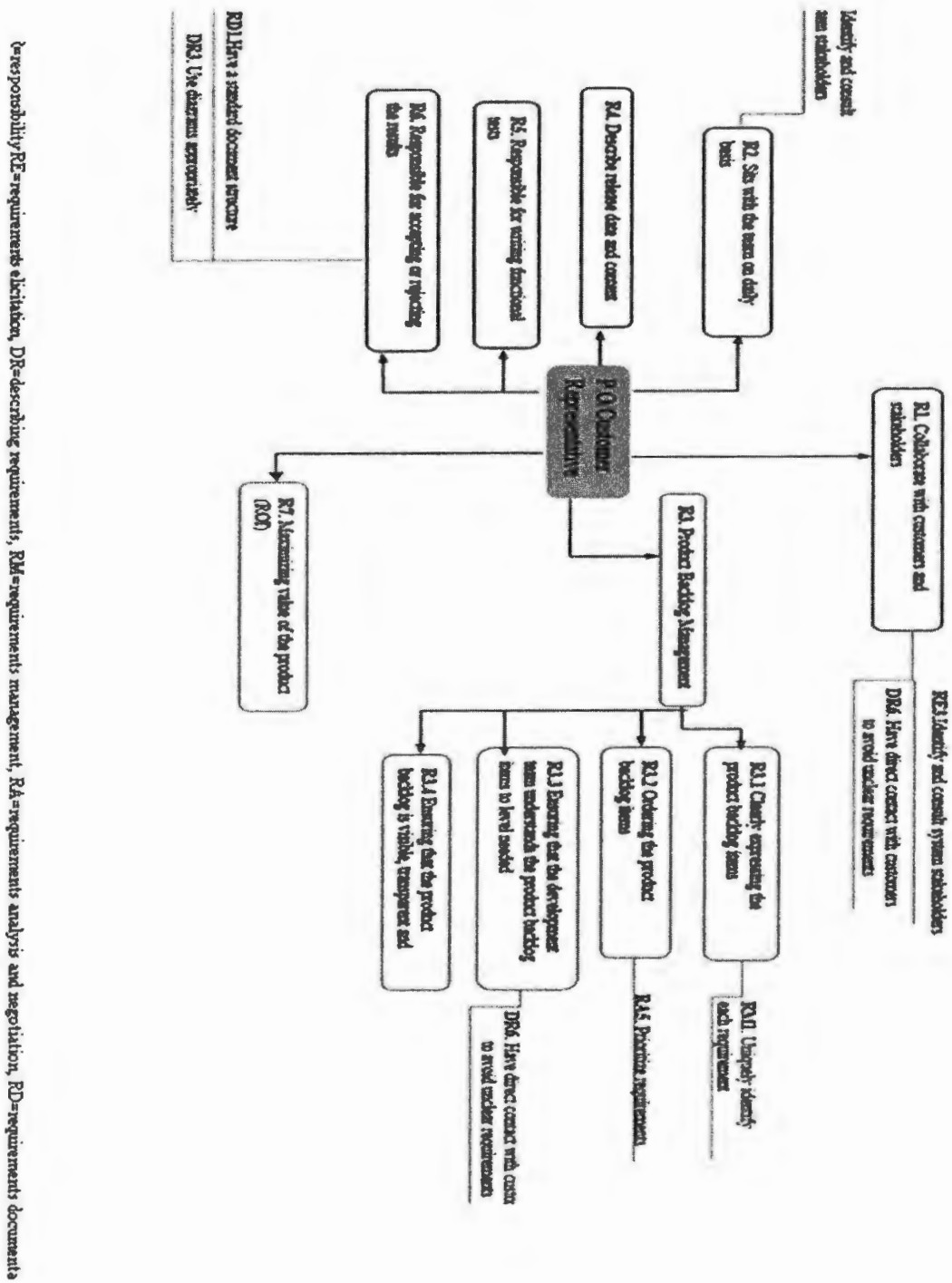
The practice DR6: have direct contact with customers is also mapped onto two responsibilities these are R1: collaborate with customers and stakeholders and R 3.3: Ensuring that the development team understands the product backlog items to the level needed. Both of these responsibilities R1 and R3.3 are enforcing the communication between the customer and the development team so that the customer (customer representative) can clarify the requirements or backlog items so that the team can accurately implement them.

RM1: uniquely identify each requirement is mapped onto responsibility R3.1: clearly expressing the product backlog items. Clearly expressing backlog items include giving each backlog item/requirement a unique identifier to avoid ambiguity among different requirements and avoid traceability problems. Practice RA5: prioritize requirements is mapped to the responsibility R3.2: ordering the product backlog items.

The RE practices RD1: have a standard document structure and DR3: use diagram appropriately are mapped into responsibility R6: responsible for accepting or rejecting the results. As this is the

time when the product owner can accept or reject the results after ensuring that the standards of the organization are properly followed.

Figure 14: Mapping the RE practices into customer representative/P.O responsibilities



5.4.3. Discussions with RE Experts

Having direct contact with the customer is one of the valuable RE practices. Most of the RE activities like requirements prioritization conducted continuously requires customer's direct involvement. The study on critical success factors in agile projects [28] identified customer interaction factor affecting the project scope. However, customer availability and access are also challenging in agile methods. The results of the study done by Rajesh et. al. [4] show that most of the agile teams do not have direct access to customers. In agile, the challenge of the unavailable customer is mostly overcome by having a proxy customer representative [10], [30]. "The on-site customer's role is indeed demanding, requiring a strong ability to resolve issues rapidly" [27]. This indirect link between the customer and the team has a negative impact on the required intense communication [4]. This indirect communication may cause the risk of decreasing clarity on requirements that in turns may cause more requirements change and need of having detailed specification of requirements [29]. Therefore, it is important to know, what are the issues in using the proxy customer representative instead of real from RE perspective?

To further understand the role of customer representative/P.O in practice we conducted discussions with RE practitioners through LinkedIn groups "Agile" (https://www.linkedin.com/grp/home?gid=81780) and "Lean Agile Software Development Community" (https://www.linkedin.com/grp/home?gid=1024087). These are two most popular groups of agile practitioners. The experienced practitioners involved in agile development participated in the discussions. In "Lean Agile" 12 members participated in our discussion. The roles of these members include Sr. Technical Director, Agile Product Owner & Business analyst, and Scrum coach/master. In "Agile" 11 members participated in the discussion. The roles of these members include Scrum agile expert, Certified Scrum Trainer and Agile Coaches. The focus of discussions was on the role of proxy customer representative in the agile development process in real world setting. We asked, "What is the impact of having a proxy customer representative/PO?" A total of 23 professionals participated in the discussion. All of the participants except one pointed out different issues. To analyze the discussion transcript, we applied the four-stage qualitative data analysis process outlined by [31].

RE3 (Identify and consult system stakeholders), RA5 (Priorities requirements) and RM1 (Uniquely identify each requirement) are commonly cited high value (>=50%) practices in both studies. RA1 (Define system boundaries), DR1 (Define standard templates for describing requirements) and RV1 (Check that the requirements document meets your standards) are commonly cited high value practices in GSD study. RD1 (Define a standard document structure) and DR3 (Use diagrams appropriately) are commonly cited high value practices in the current study.

RE Category	Practice	Comparison of Studies				
		Total High value frequency	GSD Study N=56	Current Study N=54	Chi-square Test (Linear-by-Linear Association) α = .05 df = 1	
					X^2	P
Documentation	RD1 Define a standard document structure	52	24	28	.907	.341
Elicitation	RE3Identify and consult system stakeholders	73	36	37	.014	.905
Analysis & Negotiation	RA1 Define system boundaries	54	29	25	.510	.475
	RA 5Prioritise requirements	61	30	31	.000	.922
Description	DR1 Define standard templates for describing requirements	53	28	25	1.669	.196
	DR3 Use diagrams appropriately	53	25	28	1.372	.241
Validation	RV1 Check that the requirements document meets your standards	47	28	19	1.739	.187
Management	RM1 Uniquely identify each requirement	61	28	33	.658	.417

Table 4.14. Comparison with GSD study through statistical analysis

Table 4.14 shows no significant difference was observed in practices identified by practitioners of GSD and the current study. As the value of p is higher than .05 in case of all practices.

CHAPTER 5

EXTREME REQUIREMENTS ENGINEERING (XRE)

5. Extreme Requirements Engineering (XRE)

In the previous chapter, we reported the results of our survey. The aim of the survey was to assess the perceived value of RE practices. The survey revealed six RE practices that were perceived as having high values according to the RE professionals worldwide. These practices are related to stakeholder's consultation, requirements specification, and requirements management. These three areas are thus the main focus of the solution. Instead of re-inventing the wheel we have complemented the existing most common agile software development methods SCRUM and Extreme programming (XP) with guidelines for agile roles to ensure the extreme use of these six high value practices.

The XRE was designed in two stages. In the first stage, we reviewed the existing methods of Scrum and XP. We looked at the work done by the inventive authors of SCRUM and XP as well as held discussions with practitioners to know the current state of the art through LinkedIn groups. Appendix I shows details of these discussions. We also reviewed the existing literature to know how RE practices are handled in agile.

In the second stage, we developed the guidelines for the roles that they have to implement in different stages of these methods to ensure the proper use of six high value RE practices. In the following paragraphs, we will firstly introduce the SCRUM and XP.

5.1. SCRUM

SCRUM is one of the most widely adopted agile method today [106]. SCRUM adopt an iterative and incremental approach to improving predictability and reduce the risk. In scrum the development is done in sprint that last from 1-4 weeks. At the beginning of each sprint, the team selects requirements from a prioritized list to be implemented in coming sprint. At each day of work the team gathers and reports on their work. At the end of the sprint, the team presents their work which is in the form of workable product increment. Three pillars of SCRUM are transparency, inspection and adoption [116]. Transparency means the process must be visible to those who are responsible for the outcomes. Inspection means that the stakeholders must be involved and inspect the process. And the adoption means that if something is going wrong and it seems that the resulting product will be unacceptable because of it then it should be removed.

SCRUM have two main artifacts product backlog and sprint backlog. The product backlog contains requirements of the product which are refined regularly and time to time new requirements are added. At the beginning of each sprint, some requirements from the product backlog are transferred to the sprint backlog that will be implemented in the sprint.

5.1.1 SCRUM Roles

The SCRUM team have three primary roles these are the product owner, scrum master, and the development team. The customer representative or product owner is responsible for defining the product's features, ordering them, reviewing them and accepting or rejecting the results [108]. The scrum master acts as the leader of the team and makes sure that the process is applied properly. The development team is responsible for the development of the working product.

5.1.2. SCRUM Events

The SCRUM itself is a container of the formal events that are performed in it. The SCRUM consists of sprint planning, daily stand-up meetings, development work, sprint review and the sprint retrospective. Figure 12. Shows the SCRUM events.

The sprint planning meeting is conducted to decide what goal will be achieved in this sprint and what work will be needed to perform to achieve this goal. The items are selected from the product backlog and sprint backlog is prepared. Development team, product owner, and scrum master usually participate in this meeting. Other willing stakeholders can also be invited.

The next event is daily standup meetings. The daily meetings are of 15 minutes usually and the teams synchronize their work in these meetings and plans the work that they will do in next 24 hours. The product owner, scrum master and the development team participates in it. But usually only the development teams speaks about their work. The sprint review meeting is performed at the end of each sprint to review the outcomes of the sprint and discuss the work done in the sprint. The product owner accepts or rejects the results of the sprint. The sprint retrospective meeting is conducted by SCRUM team to inspect itself and plan for improvements.

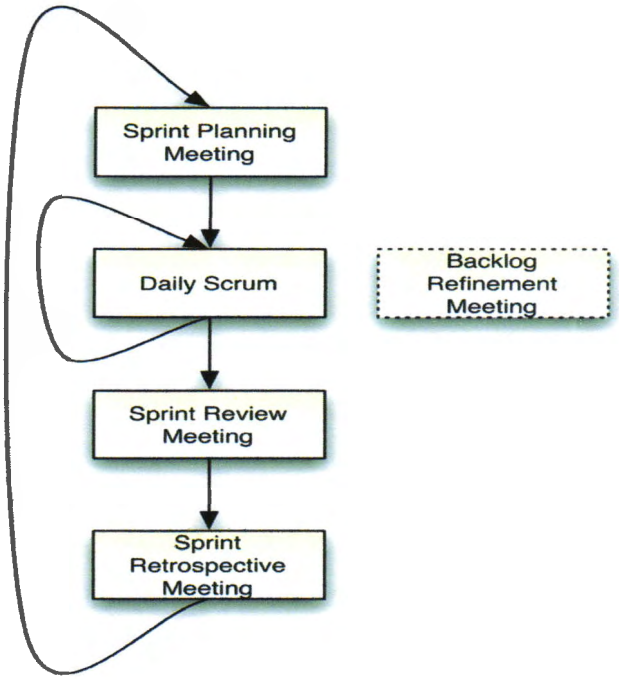


Figure 123: The Scrum events

(From Agile, 2015, http://www.eceltic.ie/our-work/agile/)

5.2. Extreme Programming (XP)

The extreme programming is a light weighted agile methodology as well as framework. It is an incremental and iterative process. XP consists of smaller iterations called releases. Each release in XP consists of couple of iterations and each iteration is at most three weeks [109]. The strength of XP process lies in its rules and practices. XP standard practices are collective code ownership, programming in pairs, continues integration, coding standards, on-site customer, simple design, refactoring, test driven development, planning game, system metaphor, sustainable pace and small releases.

5.2.1. XP Roles

XP have six primary roles. Customer representative, programmers, coach, tracker, testers, and consultant. Customer representative is an onsite customer, who is responsible for refining the user stories and driving functional tests etc. programmer is responsible for writing the code. The

coach and the tracker are responsible for the management of the project. The coach is responsible for technical managing the project while tracker tracks the systems success. The tester helps to choose the functional tests and run them. The consultant is hired to provide the knowledge of the process to team.

5.2.2. XP Events

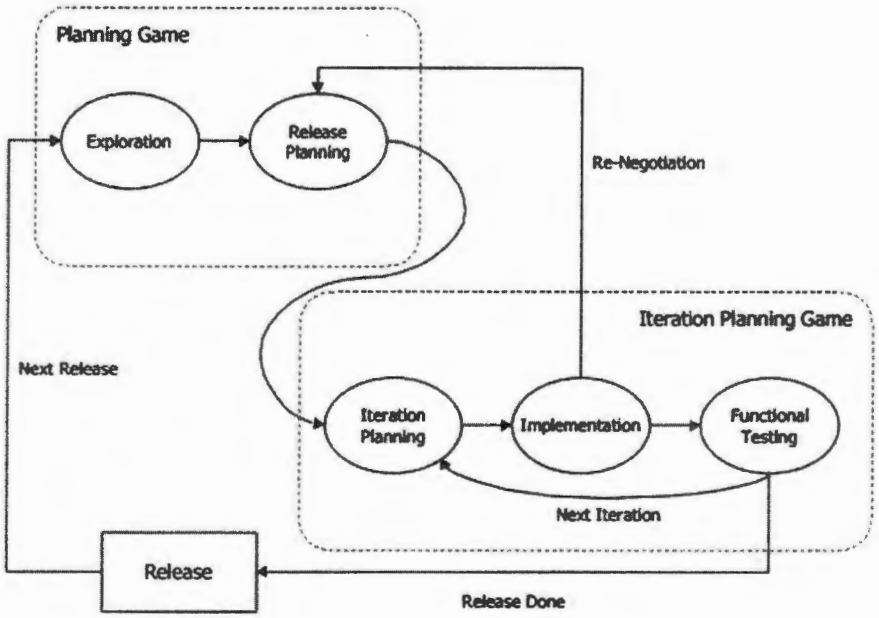


Figure 4: XP simplified process structure

(From eXtreme Programming an Overview, Dudziak, 2000, p.8)

Figure 13 shows the simplified XP process. The planning game consists of three stages exploration, release planning, and steering. In the exploration phase, user provides the user stories and the development team estimates the effort required to complete each user story. If the stories are too big then they are broken down into smaller stories and if they are too small they are combined with other stories. This phase takes from a couple of weeks to few months. Next is the planning phase where the commitment schedule meeting is conducted to determine the commitment schedule. Commitment schedule is the set of stories along with their estimates that are to be implemented in the current release [109]. After planning phase, the steering phase

comes. Steering mean starting the actual process by little moves. These moves include the iterations, recovery from wrong estimates, from wrong release date and adding new stories.

The iterations also consist of three phase's exploration, planning, and steering. In the exploration the customers select the stories from commitment schedule to be implemented in current iteration. And the developers break these stories into smaller tasks. In the planning phase for each task, a developer accepts the responsibility and estimates the ideal engineering time to complete the stories. In the steering phase, the actual coding is done and the fictional tests are run to check whether the completed stories work or not. Daily standup meeting is also conducted on the regular basis. At the end of each iteration, the functional test cases associated with stories are reviewed with the customers if they fail they are included in next iteration.

5.3. How RE Practices are handled in Agile and what are the Challenges Posed by Agile on RE

To answer this question we took help from a recent literature review [128] on RE practices and challenges in agile.

Requirements Engineering Practices in Agile: Minimal requirements documentation and more face-to-face communication is one aspect of agile software development [129]. Documentation is done in the form of user stories to avoid long and complex requirements documents. User stories enhance communication among stakeholders [130] and shift the focus from documentation to communication [133]. User stories eliminate the need of continues updating of requirements specification document [132].

Identification of stakeholders and frequent collaboration with an onsite customer to avoid disagreement and differencing of views on the variety of issues is a characteristic of agile methods [130] [131]. In agile methods, requirements emerge iteratively over time [134]. This iterative approach have many benefits like it makes the requirements clearer over time, reinforces the relationships with customers and allows changes in requirements in less time and cost [131].

Requirements prioritization is done on continues basis by customers mostly based on business value [131] or based on risk [130]. The dynamic nature of agile facilitates the change

management process. The main changes are to add up and drop features [131]. Cross-functional teams have members from different groups with similar goals. For example in agile designers, developers, testers and managers sit and work together. Challenges like communication gaps and over scoping of requirements can be overcome with this way of working [51].

Prototyping is used to review requirements and get feedback from customers. Prototyping is started with requirements that are simple, completely understood and have high priority [135]. In most agile methods like XP testing before coding approach is followed where test specifications are written before writing the actual code. This approach encourages early feedback in case a test fails.

Automated test-driven development is another emerging trend in this field that combines practices from both traditional and agile RE [136]. Requirements modeling is performed in agile but in a different way as compared to requirements modeling in traditional software development methods. One approach used for requirements modeling in agile methods is goal-sketching [128]. In goal sketching goal are refined for each iteration. This technique empowers the decision making in the negotiation process [137].

Requirements management is performed by maintaining the product backlog items and index cards [131] [134]. Developed requirements and remaining product backlog items are constantly reviewed in review meetings and through acceptance tests [133]. The acceptance tests are like unit tests that may result in "pass" or "fail" for a user story. These acceptance tests increase the collaboration between development team, domain experts, and customers and reduce the severity of defects [128]. The requirements analysis is performed by pairs that encourage the stakeholders to perform multiple roles as well [139]. Pairing practice for requirements analysis is one of the ways that closes communication gaps in agile teams [128]. Retrospective meetings are held after the completion of each iteration.

The new requirements are incorporated in the form of changes in deliverables by customers [140]. Instead of a static plan the agile teams do continues planning which helps in changing requirements even in later stages of the project [141].

Requirements Engineering Challenges in Agile: Minimal documentation is a challenge posed by agile methods [131] [134]. Instead of doing conventional requirements documentation agile teams do to-the-point, precise user story oriented documentation [133]. "In some cases especially in large projects where all team members are not co-located verbal communication is insufficient without proper documentation" [128]. To overcome this challenge, mostly user stories are complemented with more detailed artifacts [130] which helps in making right implementation choices in the coding stage.

Customer availability and access is another challenge for agile methods [134]. The customer unavailability is mostly because of certain factors like time, cost, and workload of customer representative [142]. This challenge is mostly overcome by having an on-site proxy customer [130] or moving a developer representative to the customer's site [142].

Budget and schedule planning is another challenge posed by agile methods. Upfront estimation is not possible when there are volatile requirements and planning [134]. "The cost is usually calculated based on user stories and new user stories may be included or some may be discarded in forthcoming iterations" [131].

The inappropriate architecture is another challenge where the architecture finalized by teams in early stage become inappropriate in later stages [134]. Another challenge which can cause massive rework is neglecting non-functional requirements [143]. The agile methods are flexible and allow change, but it can create troubles incorporating this much change and evaluating the consequences of change [128].

5.4. Designing the Solution

The high value RE practices which we have identified through survey are have direct contact with customers to avoid unclear requirements, identify and consult system stakeholders, have a standard document structure, use diagrams appropriately, prioritize requirements and uniquely identify each requirement. Hence, the primary focus of the solution is on three areas of RE stakeholder's collaboration, requirements specifications, and requirements management. Customer collaboration is one of the main feature and success factors of the agile projects [109-114].

Each agile team has an on-site customer representative who is chosen by the stakeholders to make decisions on their behalf. In SCRUM, this representative is called the product owner (P.O). He is usually responsible for prioritizing the requirements, accepting or rejecting the results and optimizing the value of the product in both XP and SCRUM projects. As most of the RE activities like requirements prioritization, requirements specification activities like enforcing the role to implement standard document structure are actually performed by the customer representative/P.O in agile projects. Hence, our solution revolves around the role of the customer representative/P.O. To acquire deep knowledge of the standard responsibilities of the P.O we studied the work by inventive authors of SCRUM and XP. We developed the list of responsibilities of P.O and then we mapped the six high value RE practices onto these responsibilities. Following paragraphs provide detail of these two steps.

5.4.1 Standard Responsibilities of Customer Representative/ P.O

Responsibilities of P.O in SCRUM: According to Jeff Sutherland and Ken Schwaber "The Product Owner is responsible for taking all the inputs into what the product should be from the customer or end-user of the product, as well as from Team Members and stakeholders and translating them into a product vision" [115]. Following are the main responsibilities of the P.O,

- The P.O defines the features of the product and decides on release date and content
- The P.O is responsible for the profitability/ROI of the product
- He prioritizes the product features/requirements according to the business value
- Can change the priorities of the items every thirty days
- He is responsible for accepting or rejecting the results

Jeff and Ken provided the following list of P.O responsibilities w.r.t different events in "The SCRUM papers" [115] in 2009 and "The SCRUM guide" [116] in 2011. In sprint planning meeting, the product owner reviews the product backlog and discusses the goal and context of the items with the scrum team. During the sprint he is mostly responsible for the break down the bigger items in smaller ones, clarify items and verify the tradeoffs. He attends the daily standup meeting. In the sprint review meeting the completed code is demonstrated to the P.O he accepts or reject it. And the remaining items on the product backlog are reprioritized again. Throughout the project, he is responsible for communication with stakeholders and manage their needs.

Responsibilities of customer representative in XP: According to Beck [117] following are the main responsibilities of customer representative in XP.

- He decides the scope and timing of the release
- Customers write the functional tests for the stories
- Collaborate with stakeholders, sits with team full time
- He picks the next release stories and next iteration stories

5.4.2. Mapping the RE practices into customer representative/ P.O responsibilities

In this section firstly we combined the responsibilities of customer representative in XP and P.O in SCRUM, simplified them and then mapped the six valuable RE practices onto these responsibilities. Figure 14 shows this mapping.

RE3: identify and consult system stakeholders is mapped on two responsibilities R1: collaborate with customers and stakeholders and R2: sits with the team on the daily basis. As both of these responsibilities include collaboration with stakeholders which are the development team and other stakeholders like other customers, users, suppliers, and testers etc.

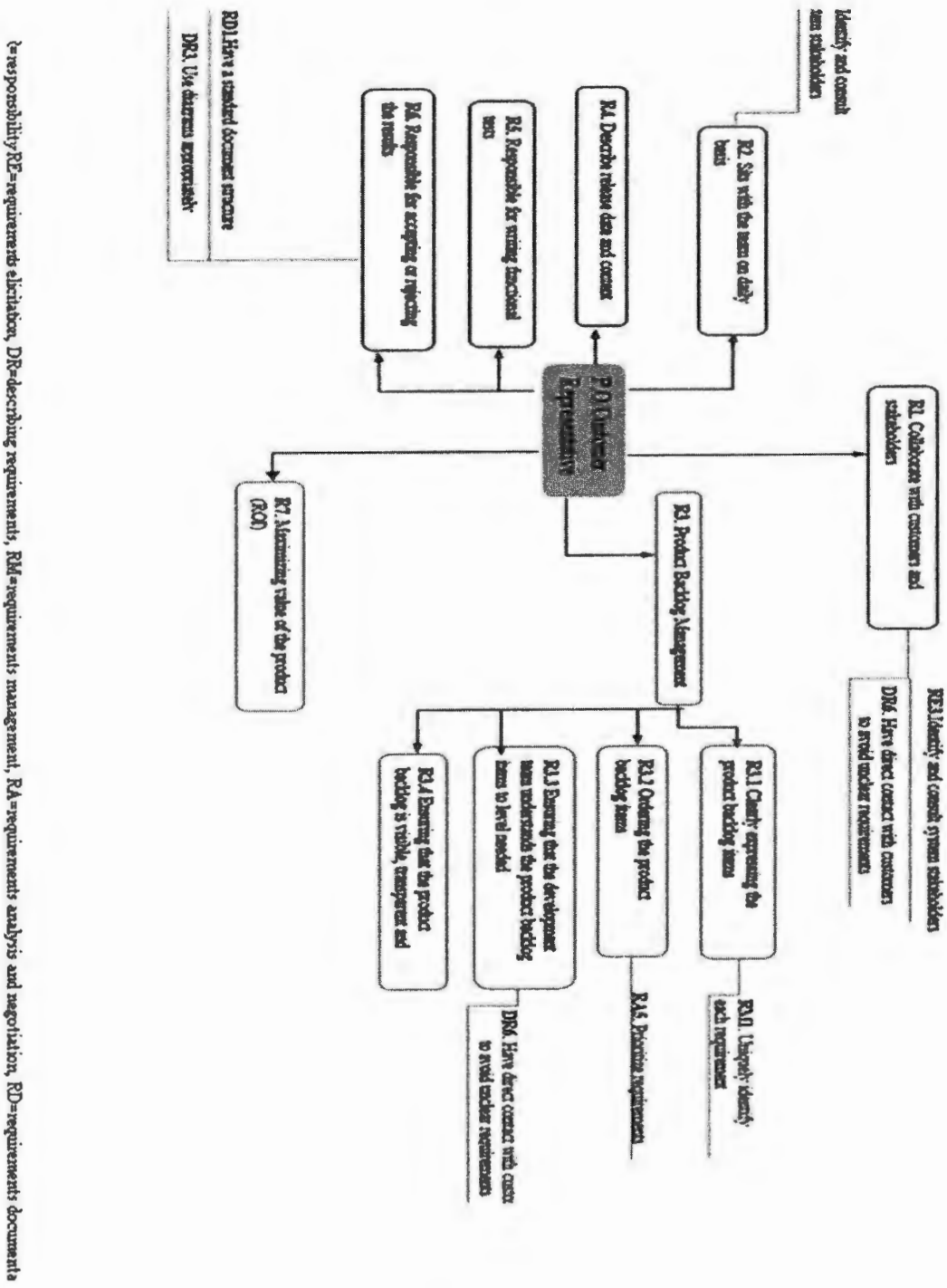
The practice DR6: have direct contact with customers is also mapped onto two responsibilities these are R1: collaborate with customers and stakeholders and R 3.3: Ensuring that the development team understands the product backlog items to the level needed. Both of these responsibilities R1 and R3.3 are enforcing the communication between the customer and the development team so that the customer (customer representative) can clarify the requirements or backlog items so that the team can accurately implement them.

RM1: uniquely identify each requirement is mapped onto responsibility R3.1: clearly expressing the product backlog items. Clearly expressing backlog items include giving each backlog item/requirement a unique identifier to avoid ambiguity among different requirements and avoid traceability problems. Practice RA5: prioritize requirements is mapped to the responsibility R3.2: ordering the product backlog items.

The RE practices RD1: have a standard document structure and DR3: use diagram appropriately are mapped into responsibility R6: responsible for accepting or rejecting the results. As this is the

time when the product owner can accept or reject the results after ensuring that the standards of the organization are properly followed.

Figure 14: Mapping the RE practices into customer representative/P.O responsibilities



5.4.3. Discussions with RE Experts

Having direct contact with the customer is one of the valuable RE practices. Most of the RE activities like requirements prioritization conducted continuously requires customer's direct involvement. The study on critical success factors in agile projects [28] identified customer interaction factor affecting the project scope. However, customer availability and access are also challenging in agile methods. The results of the study done by Rajesh et. al. [4] show that most of the agile teams do not have direct access to customers. In agile, the challenge of the unavailable customer is mostly overcome by having a proxy customer representative [10], [30]. "The on-site customer's role is indeed demanding, requiring a strong ability to resolve issues rapidly" [27]. This indirect link between the customer and the team has a negative impact on the required intense communication [4]. This indirect communication may cause the risk of decreasing clarity on requirements that in turns may cause more requirements change and need of having detailed specification of requirements [29]. Therefore, it is important to know, what are the issues in using the proxy customer representative instead of real from RE perspective?

To further understand the role of customer representative/P.O in practice we conducted discussions with RE practitioners through LinkedIn groups "Agile" (https://www.linkedin.com/grp/home?gid=81780) and "Lean Agile Software Development Community" (https://www.linkedin.com/grp/home?gid=1024087). These are two most popular groups of agile practitioners. The experienced practitioners involved in agile development participated in the discussions. In "Lean Agile" 12 members participated in our discussion. The roles of these members include Sr. Technical Director, Agile Product Owner & Business analyst, and Scrum coach/master. In "Agile" 11 members participated in the discussion. The roles of these members include Scrum agile expert, Certified Scrum Trainer and Agile Coaches. The focus of discussions was on the role of proxy customer representative in the agile development process in real world setting. We asked, "What is the impact of having a proxy customer representative/PO?" A total of 23 professionals participated in the discussion. All of the participants except one pointed out different issues. To analyze the discussion transcript, we applied the four-stage qualitative data analysis process outlined by [31].

Stage 1: Collect and Analyze Discussion Data

We carefully inspected each post in the discussion for its research relevance before being including it in the analysis.

Stage 2: Categorize Data (Coding)

We used coding to identify the concept categories/themes. Coding reduces or groups text into manageable chunks or concept categories. A theme represents the underlying knowledge embedded in the text. For example, we analyzed the statements from discussion and identified that the text (in the transcript of discussion) "How good does your proxy know the business?" and "Proxy Product Owner should be a subject matter expert" belong to same concept category "Understanding of the business". Table 2 shows the categories and reference number of professionals mentioning it in their responses.

Themes	Expert Ref#	Frequency
Frequent Interaction with customers and stakeholders	5, 8, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23	12
Understanding of the business	2, 3, 5, 6, 8, 9, 12, 13, 17, 21	10
Independent decisions	5, 9, 10, 11, 12, 13, 14, 21	8
Stakeholders representation	5, 3, 11, 17, 18, 21	6
Situation Dependent	7, 15, 19	3
Business analyst as proxy works well	1	1
P.O should not be too technical	2	1
Amount of time dedicated by proxy	3	1
Losing quality of communication	4	1

***Table 5.1** Themes identified from discussions with RE experts*

We were able to identify four major themes (Table 5.1 – light pink rows) after the analysis of the discussion transcript. These themes are frequent interaction with customers and stakeholders, understanding of the business, independent decisions, and stakeholders' representation. The participants mostly pointed out the frequent interaction of P.O./C.R. with customers and stakeholders a pre-requisite for success. They were of the opinion that failure to do this will

result in lack understanding about business and business value to deliver which was echoed in the responses such as

"How do you maximize the value created by a development team without involving all of the stakeholders?" Certified Scrum Trainer and Coach.

> *"Alignment needs to be ensured though, as the PO usually is not the (buying) customer. So there is a risk of misalignment which needs to be managed also by getting frequent feedback from real (buying) customers or end users that need the product to provide x value through usable functionality"* Enterprise Agile Coach.

The second most referred theme by the participants was understanding the business for which software is being developed. They viewed the lack of understanding created a negative impact on the project.

> *"The PO can negatively impact the project if he or she does not know enough about the solution/project"* Sr. Agile Practitioner.

One of the responsibility of a P.O. prioritization of product backlog items. This activity requires the knowledge of business value as agile development focuses on value driven development and delivery.

The third theme pointed out by the participants was independent decisions. They were of the view that the success of proxy P.O. is dependent on the authority s/he has in making decisions. This was echoed in a comment made by a P.O. of a large company highlighting the link between understanding business and independent decision making.

> *"What you need in a product owner, is someone who knows about the business enough that they can make independent decisions without having to have meetings with multiple stakeholders every time."* Product Owner (at a large company)

The last of the four themes is stakeholder representation. The participants were of the view that proxy P.O. must be a true representative of the customers.

One minor theme (in responses of three professionals) is "This depends on the situation". Three of issues were raised by three individuals and thus were not included for themes. These are "Proxy should not be too technical as technical people tend to ignore business", "Proxy must spend most of his/her time full filling this role", and "Having proxy will result in loss of quality communication".

Stage 3: Apply Thematic Network Analysis

We selected four frequently mentioned concept categories after organizing the data into concept categories. We applied the thematic network analysis to each concept category to explore the relationship (Figure 15).

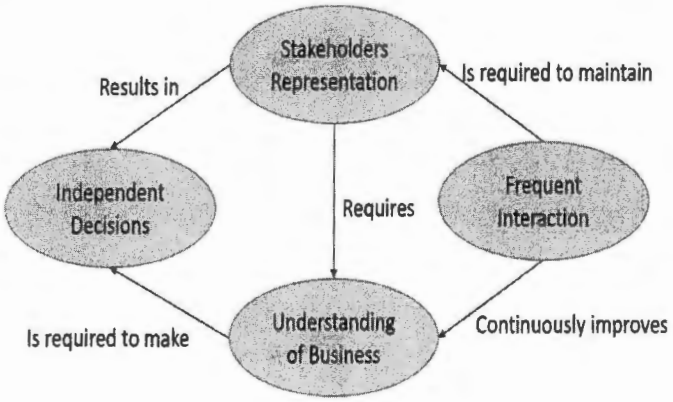


Figure 55: Thematic Network

Stage 4: Interpret the Thematic Network Analysis

Finally, we interpreted the thematic network to identify key points. We identified two critical points if an organization decides for a proxy PO/CR. Firstly PO/CR has to be a true representative that can be visible by the number of independent decisions taken by her/him. Secondly, a true representation of customer requires a good understanding of customer's business and frequent interaction with real customer and stakeholders. These dependencies can be summarized with a help of the comment made by a participant.

> *"My experience is that as long as the proxy understands the problem and business domain, works with the team and is empowered to make decisions, then the value of having a proxy far outweighs the impact of delays to*

development when we are waiting for responses from our seldom available PO." Delivery Lead.

5.4.4. Presenting Guidelines and Implementation Details

The agile practitioners, as discussed in the last section, do not view the use of a proxy as a problem as long as proxy representing customer clearly understand problem and business domain and can make independent decisions. All of this, requires a frequent interaction of proxy with real customers and stakeholders. Figure 16 shows the six guideline that we have developed for customer representatives/product owners. It also shows the Scrum and XP events where these guidelines will be implemented. In following paragraphs, we will give details of each guideline.

A. Guideline 1 (Including fourth question in daily SCRUM)

The daily Scrum and XP meeting is an event where the activities of development teams are synchronized and a plane for next 24 hour is made. Customer representative/P.O also attends this meeting. In Usual Scrum and XP daily standup meeting the development team answers three questions,

1. *What did I do yesterday to help the team meet the sprint/iteration goal?*
2. *What will I do today to help the team meet the sprint/iteration goal?*
3. *Do I see any impediment that prevents me or the Development Team from meeting the sprint/iteration Goal?*

We have proposed a fourth question for the daily SCRUM or XP meetings. This will be answered by the customer representative (real or proxy).

4. *Did you communicate with any stakeholder yesterday, what was the focus of communication?*

The main aim of introducing this question is to ensure stakeholders consultation at the extreme level or on the daily basis. The thematic analysis done in the last section shows that this frequent interaction is required. The customer representative/P.O is responsible for collecting the thoughts and ideas of other stakeholders and sharing them with the development team. The focus of communication can be on specific point like project timeline, cost, requirements etc.

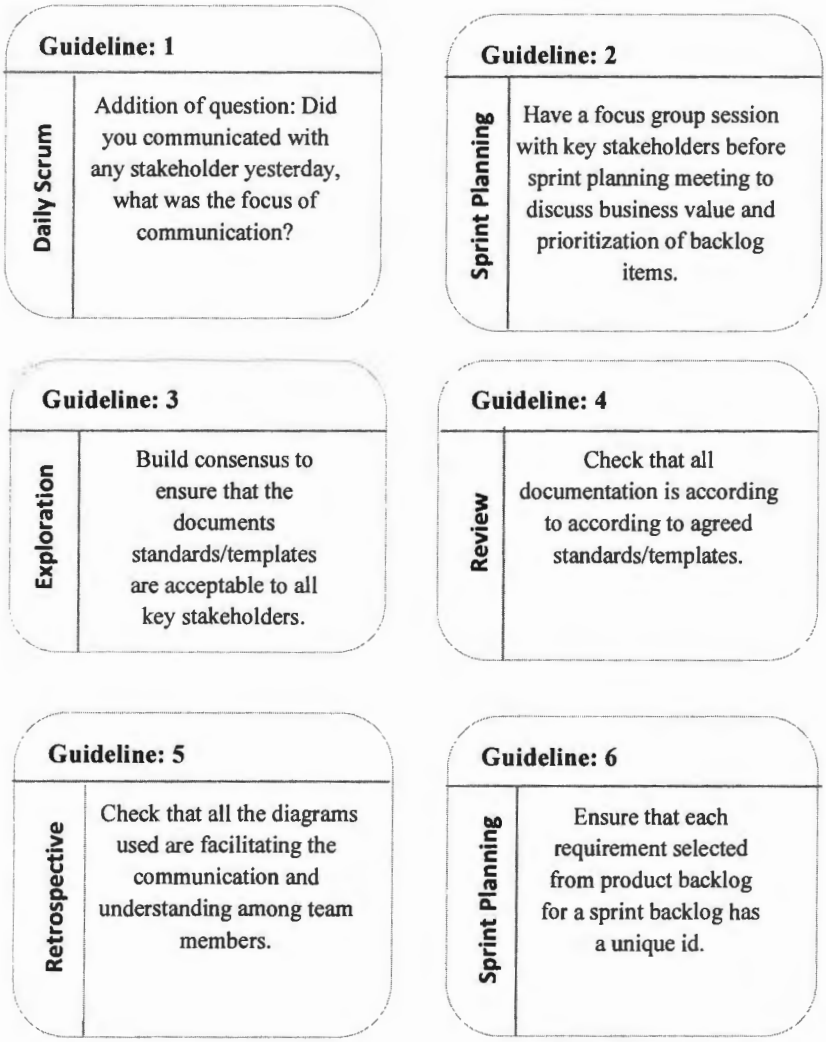


Figure 66: Guidelines for customer representative/ P.O

When ensuring the stakeholder's agreement on work done by development team on the daily basis, we can increase the probability of success of the sprint/iteration. According to Schwaber et al [116] the daily scrum meetings promote quick decision making and improve knowledge of the development teams. This guideline further increases these two reimbursements given by daily meetings. As the daily standup meetings are short in duration usually 15 minutes so the answer to this question should be very brief.

The customer representative/P.O will wait until the development team is done with answering their questions and do not interfere in their discussion, in the end he will answer his question. This rule is followed because the inventors of SCRUM [11] suggest that no one should ask any question or open any discussion while the development team is reporting their work until the meeting concludes. It is not necessary to meet the stakeholders directly the P.O can use any communication medium. It is no necessary to mention the communication medium while answering the question.

B. Guideline 2 (Have a meeting/focus group with key stakeholders before the sprint planning)

The customer representative/P.O should conduct a meeting with all interesting stakeholders and discuss the priorities of the product backlog items and understand business value. This should be done through a focus group session as it will help in resolving the conflicts if any. In SCRUM sprint planning meeting or XP iteration kickoff meeting it should be assured that this prioritization meeting was conducted by the customer representative/P.O.

The primary purpose of this meeting is to understand the business value and prioritize accurately the backlog items with the involvement of key stakeholders in the prioritization process. With the stakeholders close cooperation we can maintain a clear and prioritized list of demand on the product backlog [118]. The most important responsibility of P.O is to manage needs and expectations of different stakeholders [119].

On-site customer is one of the primary rules and strengths of agile. But some XP and SCRUM project studies [120-122] reported that they are not actually able to implement this practice because of problems of customer unavailability and lack of knowledge. In such situations, the practice of on-site customer is implemented by having a knowledgeable engineer or manager play the role of customer who is called a proxy customer. The proxy customer usually does not clearly know the opinion of actual customers and other stakeholders as compared to an actual customer representative. So it is necessary for him to discuss different features with them before making an important business decision. We recommend the use of focus group session to conduct this stakeholders' meeting as it is most appropriate and an easy way to facilitate focused communication among groups of stakeholders [127].

C. Guideline 3: (Build consensus to ensure that the documents standards/templates are acceptable to all key stakeholders)

In agile requirements are documented in the form of user stories. Development teams are often uncertain about what information should be included in the user stories and in what detail [157]. Useful templates are available from many sources that can be used as standards. It is the responsibility of the product owner to develop a contract over the documentation standard that is accepted by all key stakeholders. "Developing a contract for each user story provides a streamlined mechanism for managers, developers, and testers to agree on details of the story" [158]. These standards can be used later as acceptance criteria for user stories.

D. Guideline 4 (Check that all documentation is according to agreed standard)

The agile team must make sure that the document conforms to the agreed standard document structure before accepting the any document including user stories. To check this the team can use any method for example he can use the standard/template as a checklist and can check different sections one by one. The standard structure for documentation is very important. According to Sommerville et al [5] by using a standard document structure we can get several benefits such as the use of knowledge from a previous document that results in understanding the relationships among different documents and parts.

E. Guideline 5 (Check the diagrams are used appropriately)

The agile team should keep checking that the diagrams are used to facilitate the development process and are being used appropriately. In agile software development, different diagrams are used at different levels. For example in Scrum projects progress is tracked and reported by using the release burn down chart, iteration burn down charts and a task boards [123]. The UML diagrams can be used to communicate the design within the team. The team members can understand the solution and contribute faster by looking at these diagrams. In XP before coding short designing sessions are recommended while in FDD (Feature Driven Development) at beginning of each iteration design is executed using UML diagrams [124]. Similarly the agile DBAs work with application developers and model their needs using UML diagrams e.g. class diagrams [125]. All developers in agile should have basic understanding of industry standard

diagrams especially UML diagrams. It is the responsibility of the customer representative/P.O to keep checking time to time that these diagrams are used to facilitate the development, are in the proper format and are simple enough that everyone can understand.

F. Guideline 6 (Check the every requirement has a unique id)

Failure of properly specifying requirements can lead to incorrect, unusable, unverifiable and inconsistent requirements. Traceability problems are one of the major results of not specifying the requirements properly. A clear and unambiguous product backlog contains the requirements with each having a unique identity. "Uniqueness is very important for each requirement to make references to other requirements and build traceability tables" [5]. "A consistent requirement is neither contradictory with other requirements of nor it should describe the same concepts as any other requirement" [126]. When the requirements are entered into the product backlog these are given a unique identifier. At sprint/iteration planning meeting when the requirements are selected from the product backlog for the sprint backlog the P.O should re-check that each requirement has a unique id. Hence, if two requirements are given the same identifier by mistake, it will be removed by the P.O at this stage.

5.5. Expected Benefits of Extreme Requirements Engineering (XRE)

By the implementation of the guidelines provide in XRE agile software development processes can get following benefits,

1. It ensures the stakeholders involvement in the agile project and avoids the problems that come from lack of customer collaboration.
2. It ensures the proper adoption of organizational standards/templates for specifications of different documents produced during the development process.
3. It reduces traceability problems by ensuring the uniqueness of requirements.
4. It reduces the overall cost of software projects by reducing requirements related problems.

6.1. Discussion and Conclusion

This chapter provides the summary of the thesis and analysis the findings with respect to the research questions.

Primarily the thesis was about identifying requirements engineering high value practices perceived by RE practitioners all over the world. Secondly designing an RE approach based on these high value practices for agile software development methods. We identified six high value practices that are related to specification standards, consultation with stakeholders, prioritization of requirements, use of diagrams, and having direct contact with customers.

This study presented some interesting facts about RE valuable practices. One of these facts is bigger use of agile methodologies and the significant differences between agile and traditional groups about the relative perceived value of RE practices. The practices related to direct involvement of the customer in the development process showed high trends. These two facts are quite normal and co-related to some extent, as agile representatives of software engineering XP and Scrum support human centered development approaches. The respondents also gave high importance to practices related to requirements specifications. This fact is normal as well, as requirements specification is the primary input to the design process it should be structured in a way that is easily understandable by the development team.

The software products of today withstand the pressures of rapid market change, fast evolution of existing technologies and the emergence of new technologies, and process change. With all these pressures the requirements management is becoming more complex, therefore increasingly challenging day-by-day. Hence, the software development organizations give high importance to requirements management practices and try to perform them very carefully. The study also revealed that there is no significant difference in responses of different groups of participants based on their geographical regions. This fact indicates that the identified high value RE practices are well-known and adopted worldwide. Therefore, the RE process/approaches based on these practices can be generalized.

We developed an RE approach XRE based on the identified six high value practices. This approach elaborates how these RE practices can be used all the time/ throughout the project to

CHAPTER 7

REFERENCES

References:

[1]. Wieringa. , R.J., "Software requirements engineering: The need for systems engineering and literacy", Requirements Engineering Journal, 6(2):132-134, 2001.

[2]. Sommerville I, Ransom J "an empirical study of industrial requirements engineering process assessment and improvement" ACM Transactions on Software Engineering and Methodology 14 (1):85-117, 2005.

[3]. Hall T, Beecham S, Rainer A "Requirements Problems in Twelve Software Companies: An Empirical Analysis" IEE Proceedings - Software (August):153-160, 2002.

[4]. Chemuturi, M., "Pitfalls and Best Practices in Requirements Engineering and Management", In Requirements Engineering and Management for Software Development Projects pp. 203-215, 2013.

[5]. Sommerville I, Sawyer P, "Requirements engineering: a good practice guide", Wiley, New York, 1997.

[6]. Davis, Alan M., and Didar Zowghi, "Good requirements practices are neither necessary nor sufficient." Requirements Engineering 11.1: 1-3, 2006.

[7]. Wiegers K, "Software requirements". Microsoft Press, Redmond, 2003.

[8]. Young R, "Effective requirements practices", Addison Wesley, Reading, 2001.

[9]. Robertson J, Robertson S, "Mastering the requirements process". Addison Wesley, New York, 1999.

[10]. Cox, K., Niazi, M., & Verner, J., "Empirical study of Sommerville and Sawyer's requirements engineering practices", IET software, 3(5), 339-355, 2009.

[11]. Niazi, M., El-Attar, M., Usman, M., & Ikram, N., "An empirical study identifying high perceived value requirements engineering practices in global software development projects", In ICSEA 2012, The Seventh International Conference on Software Engineering Advances (pp. 283-288), 2012.

[12]. Iqbal, J., Ahmad, R., Nizam, M. H., Nasir, M., & Noor, M. A., "Significant Requirements Engineering Practices for Software Development Outsourcing", In Software Engineering Conference (ASWEC), 22nd Australian pp. 137-144, 2013.

[13]. Kassab, M., Neill, C., & Laplante, P., "State of practice in requirements engineering: contemporary data", Innovations in Systems and Software Engineering, 10(4), 235-241, 2014.

[14]. Kauppinen, M., Vartiainen, M., Kontio, J., Kujala, S., & Sulonen, R., "Implementing requirements engineering processes throughout organizations: success factors and challenges", Information and Software Technology, 46(14), 937-953, 2004.

[15]. J. Stecklein, "Error cost escalation through the project life cycle," http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/ 20100036670 2010039922.pdf, NASA, Tech. Rep., 2004.

[16]. J. J Castillo, "Convenience Sampling", Sep 16, 2009 http://explorable.com/convenience-sampling, last accessed on September 27, 2013

[17]. Dingsøyr, T., Nerur, S., Balijepally, V., & Moe, N. B. , "A decade of agile methodologies: Towards explaining agile software development", Journal of Systems and Software, 85(6), 1213-1221, 2012.

[18]. Hadar, I., Kuflik, T., Perini, A., Reinhartz-Berger, I., Ricca, F., & Susi, A., "An empirical study of requirements model understanding: Use Case vs. Tropos models", Proceedings of the 2010 ACM Symposium on Applied Computing. pp. 2324-2329, 2010.

[19]. Donzelli, P., & Bresciani, P., "Goal-oriented requirements engineering: a case study in e-government", In Advanced Information Systems Engineering. pp. 601-616, 2003.

[20]. Engelsman, W., & Wieringa, R., "Goal-oriented requirements engineering and enterprise architecture: Two case studies and some lessons learned", In Requirements Engineering: Foundation for Software Quality. pp. 306-320, 2012.

[21]. Sen, A. M., & Hemachandran, K., "Elicitation of goals in requirements engineering using agile methods", In Computer Software and Applications Conference Workshops (COMPSACW), IEEE 34th Annual. pp. 263-268, 2010.

[22]. Carod, N. M., & Cechich, A., "A Cognitive Psychology Approach for Balancing Elicitation Goals", In Cognitive Informatics, 6th IEEE International Conference. pp. 232-241, 2007.

[23]. Duboc, L., Letier, E., Rosenblum, D. S., & Wicks, T., "A case study in eliciting scalability requirements", In International Requirements Engineering,. RE'08. 16th IEEE. pp. 247-252, 2008.

[24]. Aranda, G. N., Vizcaíno, A., & Piattini, M., "Analyzing Ontology as a Facilitator During Global Requirements Elicitation", In Global Software Engineering,. ICGSE. Fourth IEEE International Conference. pp. 309-314, 2009.

[25]. Lima, J. F., Garcia, B. P., Amaral, C. M. G., & Caran, G. M., "Building an Ontological Model for Software Requirements Engineering", In ENTERprise Information Systems. pp. 228-237, 2011.

[26]. Kaiya, H., & Saeki, M., "Using domain ontology as domain knowledge for requirements elicitation", In Requirements Engineering, 14th IEEE International Conference. pp. 189-198, 2006.

[27]. Kaiya, H., & Saeki, M., "September. Ontology based requirements analysis: lightweight semantic processing approach", In Quality Software.QSIC. Fifth International Conference on . pp. 223-230, 2005.

[28]. Sillitti, A., Ceschi, M., Russo, B., & Succi, G., "Managing uncertainty in requirements: a survey in documentation-driven and agile companies", In Software Metrics. 11th IEEE International Symposium. pp. 10-pp, 2005.

[29]. Sadraei, E., Aurum, A., Beydoun, G., & Paech, B., "A field study of the requirements engineering practice in Australian software industry", Requirements Engineering, 12(3), pp. 145-162, 2007.

[30]. Lowe, D., & Eklund, J., "Client needs and the design process in web projects", J. Web Eng., 1(1). pp. 23-36, 2002.

[31]. Itoga, H., & Ohnishi, A., "Security Requirements Elicitation via Weaving Scenarios Based on Security Evaluation Criteria", In Quality Software. QSIC'07. Seventh International Conference. pp. 70-79, 2007.

[32]. Juristo, N., Moreno, A. M., & Silva, A., "Is the European industry moving toward solving requirements engineering problems?", Software, IEEE, 19(6). pp. 70-77, 2002.

[33]. Lami, G., & Ferguson, R. W., "An empirical study on the impact of automation on the requirements analysis process", Journal of Computer Science and Technology, 22(3), 338-347, 2007.

[34]. Gonzales, Carol K., and Gondy Leroy, "Eliciting user requirements using Appreciative inquiry" Empirical Software Engineering 16.6: 733-772, 2011.

[35]. Neill, Colin J., and Phillip A. Laplante, "Requirements engineering: the state of the practice." IEEE software 20.6: 40-45, 2003.

[36]. Lowe, David, and John Eklund, "Client needs and the design process in web projects" J. Web Eng. 1.1: 23-36, 2002.

[37]. Juergens, Elmar, et al, "Can clone detection support quality assessments of requirements specifications", Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 2 ACM, 2010.

[38]. och Dag, Johan Natt, et al, "A feasibility study of automated natural language requirements analysis in market-driven development.", Requirements Engineering 7.1: 20-33, 2002.

[39]. Gibson, J. Chris, "Developing A requirements specification for a web service application." Requirements Engineering Conference, 2004 Proceedings 12th IEEE International IEEE, 2004.

[40]. Bhat, Jyoti M., Mayank Gupta, and Santhosh N. Murthy, "Overcoming requirements engineering challenges: Lessons from offshore outsourcing", Software, IEEE 23.5 : 38-44, 2006.

[41]. Cabot, J., Easterbrook, S.M., Horkoff, J., Lessard, L., Liaskos, S., Mazón, J.N, "Integrating sustainability in decision-making processes: A modeling strategy", In: ICSE Companion, pp.207–210. IEEE, 2009.

[42]. Cysneiros, Luiz Marcio, Julio Cesar Sampaio do Prado Leite, and Jaime de Melo Sabat Neto, "A framework for integrating non-functional requirements into conceptual models", Requirements Engineering 6.2 : 97-115, 2001.

[43]. Adam, Sebastian, et al, "Using task-oriented requirements engineering in different domains–experiences with application in research and industry." Requirements Engineering Conference, 2009 RE'09 17th IEEE International IEEE, 2009.

[44]. Albayrak, Ozlem, "An experiment to observe the impact of UML diagrams on the effectiveness of software requirements inspections", Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement IEEE Computer Society, 2009.

[45]. Jarke, Matthias, et al, "The brave new world of design requirements." Information Systems 36.7: 992-1008, 2011.

[46]. Pichler, Mario, Hildegard Rumetshofer, and Wilhelm Wahler, "Agile requirements engineering for a social insurance for occupational risks organization: A case study." Requirements Engineering, 14th IEEE International Conference IEEE, 2006.

[47]. Ko Youngjoong, et al, "Using classification techniques for informal requirements in the requirements analysis-supporting system." Information and Software Technology 49.11: 1128-1140, 2007.

[48]. Lang, Michael, and Jim Duggan, "A tool to support collaborative software requirements management" Requirements Engineering 6.3: 161-172, 2001.

[49]. Sikora, Ernst, Bastian Tenbergen, and Klaus Pohl, "Industry needs and research directions in requirements engineering for embedded systems", Requirements Engineering 17.1: 57-78, 2012.

[50]. Felfernig, Alexander, et al. "Group decision support for requirements negotiation." Advances in User Modeling Springer Berlin Heidelberg, 05-116, 2012.
[51]. Bjarnason, Elizabeth, Krzysztof Wnuk, and Björn Regnell, "A case study on benefits and side-effects of agile practices in large-scale requirements engineering", Proceedings of the 1st Workshop on Agile Requirements Engineering ACM, 2011.

[52]. Basit, Abdul, Ghulam Murtaza, and Naveed Ikram, "Validation of VRRM process model" Proceedings of the 9th WSEAS international conference on Software engineering, parallel and distributed systems World Scientific and Engineering Academy and Society (WSEAS), 2010.

[53]. Corentin Burnay, Ivan J. Jureta, Stéphane Faulkner, "What stakeholders will or will not say: A theoretical and empirical stud y of topic importance in Requirements Engineering elicitation interviews", Information Systems, Volume 46, 61-81, 2014.

[54]. Hadar, I., Soffer, P., & Kenzi, K., "The role of domain knowledge in requirements elicitation via interviews: an exploratory study", Requirements Engineering, 19(2), 143-159, 2014.

[55]. Farfeleder, S., Moser, T., Krall, A., Stålhane, T., Omoronyia, I., & Zojer, H., "Ontology-driven guidance for requirements elicitation", In the Semanic Web: Research and Applications (pp. 212-226), 2011.

[56]. Li, G., Jin, Z., Xu, Y., & Lu, Y., "An engineerable ontology based approach for requirements elicitation in process centered problem domain", InKnowledge Science, Engineering and Management (pp. 208-220), 2011.

[57]. Wang, T., Si, Y., Xuan, X., Wang, X., Yang, X., Li, S., & Kavs, A. J., " A QoS ontology cooperated with feature models for non-functional requirements elicitation", In Proceedings of the Second Asia-Pacific Symposium on Internetware (p. 17), 2010.

[58]. Carrizo, D., Dieste, O., & Juristo, N., "Systematizing requirements elicitation technique selection". Information and Software Technology, 56(6), 644-669, 2014.

[59]. Wellsandt, S., Hribernik, K. A., & Thoben, K. D., "Qualitative Comparison of Requirements Elicitation Techniques that are Used to Collect Feedback Information about Product Use", Procedia CIRP, 21, 212-217, 2014.

[60]. Tiwari, S., Rathore, S. S., & Gupta, A., "Selecting requirement elicitation techniques for software projects", In Software Engineering (CONSEG), 2012 CSI Sixth International Conference on (pp. 1-10), 2012.

[61]. Meth, H., Brhel, M., & Maedche, A., "The state of the art in automated requirements elicitation", Information and Software Technology, 55(10), 1695-1709, 2013.

[62]. Pacheco, C., & Garcia, I., "A systematic literature review of stakeholder identification methods in requirements elicitation", Journal of Systems and Software, 85(9), 2171-2181, 2012.

[63]. Krüger, Ingolf and Farcas, Claudiu and Farcas, Emilia and Menarini, Massimiliano
, "7 Requirements Modeling for Embedded Realtime Systems", Springer Berlin Heidelberg, 155-199, 2010.

[64]. Horkoff, J. M., "Iterative, Interactive Analysis of Agent-goal Models for Early Requirements Engineering", Doctoral dissertation, University of Toronto, 2012.

[65]. Bider, I., & Perjons, E., "Design science in action: developing a modeling technique for eliciting requirements on business process management (BPM) tools". Software & Systems Modeling, 1-30, 2014.

[66]. Schneider, F., Naughton, H., & Berenbach, B., "A modeling language to support early lifecycle requirements modeling for systems engineering", Procedia Computer Science, 8, 201-206, 2012.

[67]. Chaudron, M. R., Heijstek, W., & Nugroho, A., "How effective is UML modeling?" Software & Systems Modeling, 11(4), 571-580, 2012.

[68]. Lee, J., Kang, K. C., Sawyer, P., & Lee, H., " A holistic approach to feature modeling for product line requirements engineering", Requirements Engineering, 1-19, 2014.

[69]. Wnuk, K., Borg, M., & Assar, S., "Towards scalable information modeling of requirements architectures", In Advances in Conceptual Modeling (pp. 141-150). Springer Berlin Heidelberg, 2012.

[70]. Abrahão, S., Insfran, E., Carsí, J. A., & Genero, M., "Evaluating requirements modeling methods based on user perceptions: A family of experiments", Information Sciences, 181(16), 3356-3378, 2011.

[71]. Goknil, A., Kurtev, I., & Millo, J. V., "A metamodeling approach for reasoning on multiple requirements models", In Enterprise Distributed Object Computing Conference (EDOC), 7th IEEE International (pp. 159-166), 2013.

[72]. Amokrane, N., Chapurlat, V., Courbis, A. L., Lambolais, T., & Rahhou, M., "Modeling Frameworks, Methods and Languages for Computerizing Small and Medium-Sized Enterprises: Review and Proposal", In Enterprise Interoperability VI (pp. 77-87). Springer International Publishing, 2014.

[73].Badreddin, O., Lethbridge, T. C., & Elassar, M., "Modeling Practices in Open Source Software", In Open Source Software: Quality Verification (pp. 127-139). Springer Berlin Heidelberg, 2013.

[74]. Bebensee, T., van de Weerd, I., & Brinkkemper, S., "Binary Priority List for Prioritizing Software Requirements", Requirements Engineering: Foundation for Software Quality; p.67-78, 2010.

[75]. A. Perini, A. Susi, P. Avesani, "A machine learning approach to software requirements prioritization", IEEE Trans. Softw. Eng. 39 (4), 445–460, 2013.

[76]. G. Kaur, S. Bawa, "A survey of requirement prioritization methods", Int. J. Eng. Res. Technol. 2 (5)958–962, 2013.

[77]. R. Thakurta, "A framework for prioritization of quality requirements for inclusion in a software project", Softw. Qual. J. 21, 573–597, 2012.

[78]. M. Ramzan, A. Jaffar, A. Shahid, "Value based intelligent requirement prioritization (VIRP): expert driven fuzzy logic based prioritization technique'', Int. J. Innovat. Comput. 7 (3)1017–1038, 2011.

[79]. M. Dabbagh, S. Lee, "A consistent approach for prioritizing system quality attributes", in: Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD),14th ACIS International Conference on, IEEE, pp. 317–322, 2013.

[80]. P. Voola, A. Babu, "Interval evidential reasoning algorithm for requirements prioritization", in: Proceedings of the International Conference on Information Systems Design and Intelligent Applications, Held in Visakhapatnam, India, January 2012.

[81]. A. Ejnioui, C. Otero, A. Qureshi, "Software requirement prioritization using fuzzy multi-attribute decision making", in: Open Systems (ICOS), IEEE
Conference on, IEEE, pp. 1–6, 2012.

[82]. M. Babar, M. Ramzan, S. Ghayyur, "Challenges and future trends in software requirements prioritization,

in: Computer Networks and Information Technology (ICCNIT), International Conference on, IEEE, pp. 319–324, 2011.

[83]. M. Pergher, B. Rossi, "Requirements prioritization in software engineering: a systematic mapping study", in: Empirical Requirements Engineering (EmpiRE), IEEE Third International Workshop on, IEEE, pp. 40–44, 2013.

[84]. A. Felfernig, G. Ninaus, "Group recommendation algorithms for requirements Prioritization", in: Recommendation Systems for Software Engineering (RSSE), Third International Workshop on, IEEE, pp. 59–62, 2012.

[85]. Z. Bakalova, M. Daneva, A. Herrmann, R. Wieringa, "Agile requirements prioritization: what happens in practice and what is described in literature?", in: Requirements Engineering: Foundation for Software Quality, Springer, Berlin Heidelberg, pp 181–195, 2011.

[86]. A. Ahmad, A. Shahzad, V. Padmanabhuni, A. Mansoor, S. Joseph, Z. Arshad, "Requirements prioritization with respect to geographically distributed stakeholders", in: Computer Science and Automation Engineering (CSAE), IEEE International Conference on, IEEE, vol. 4, pp. 290–294, 2011.

[87]. R. Svensson, T. Gorschek, B. Regnell, R. Torkar, A. Shahrokni, R. Feldt, A. Aurum, "Prioritization of quality requirements: state of practice in eleven companies", in: Requirements Engineering Conference (RE),19th IEEE International, IEEE, pp. 69–78, 2011.

[88]. Achimugu, P., Selamat, A., Ibrahim, R., & Mahrin, M. N. R., "A systematic literature review of software requirements prioritization research. Information and Software Technology", 56(6), 568-585, 2014.

[89]. Baskaran, S., "A Survey on Prioritization Methodologies to Prioritize Non Functional Requirements. International Journal of Computer Science and Business Informatics", 12(1), 2014.

[90]. Khari, M., & Kumar, N., "Comparison of six prioritization techniques for software requirements. Journal of Global Research in Computer Science", 4(1), 38-4, 2013.

[91]. Thayer, Richard H., Sidney C. Bailin, and M. Dorfman. "Software requirements engineering", IEEE Computer Society Press, 1997.

[92]. Fowler, Kim. "Mission-critical and safety-critical systems handbook: Design and development for embedded applications", Newnes, 2009.

[93]. Potts, Colin. "Software-engineering research revisited", Software, IEEE 10.5: 19-28, 1993.

[94]. Wohlin, Claes, Martin Höst, and Kennet Henningsson., "Empirical research methods in software engineering", Empirical methods and studies in software engineering Springer Berlin Heidelberg, 7-23, 2003.

[95]. Easterbrook, Steve, et al., "Selecting empirical methods for software engineering research", Guide to advanced empirical software engineering. Springer London, 285-311, 2008.

[96]. Curtis, B., Krasner, H., & Iscoe, N., "A field study of the software design process for large systems". Communications of the ACM, 31(11). pp. 1268-1287, 1988.

[97]. Kendra Cherry, http://psychology.about.com/od/researchmethods/f/survey.htm , last accessed on August 13, 2014.

[98]. J. Singer, S. Sim and T. Lethbridge, "Software Engineering Data Collection for Field Studies," in Guide to Advanced Empirical Software Engineering, ed London: Springer, pp. 9-34, 2008.

[99]. Explorable.com (Apr 24, 2009). Snowball Sampling. Retrieved Sep 18, 2014 from Explorable.com: https://explorable.com/snowball-sampling

[100]. Data Analysis, Faculty Development and Instructional Design Center Northern Illinois University, USA Last accessed September 18, 2014 http://ori.hhs.gov/education/products/n_illinois_u/datamanagement/datopic.html

[101]. Bland, M., "An introduction to medical statistics", (No. Ed. 3). Oxford University Press, 2000.

[102]. THE CHI-SQUARE TEST, Last accessed September 18, 2014 http://www.tiem.utk.edu/~gross/bioed/bealsmodules/chi-square.html

[103]. Yu, Xiaodan, and Stacie Petter., "Understanding agile software development practices using shared mental models theory", Information and Software Technology, 2014.

[104]. Chaudhary, Ashish, Anil Punia, and Malhar Pujar, "Requirements engineering's Role in Agile Development." Infosys labs.[Online]. Available: http://www. infosys.com/Infosyslabs/publications/documents/requirement-engineering-agiledevelopment.pdf, 2008.

[105]. Saiedian, H., & Dale, R., " Requirements engineering: making the connection between the software developer and customer". Information and Software Technology, 42(6). pp. 419-428, 2000.

[106]. M. Pikkarainen, J. Haikara, O. Salo, P. Abrahamsson, J. Still, "The impact of agile practices on communication in software development", Empirical Softw. Eng.
13, 303–337, 2008.

[108]. Hoda, R., Noble, J., & Marshall, S., "The impact of inadequate customer collaboration on self-organizing Agile teams". Information and Software Technology, 53(5), 521-534, 2011.

[109]. Dudziak, T., "eXtreme Programming an Overview". Methoden und Werkzeuge der Softwareproduktion WS, 2000.

[110]. R. Hoda, J. Noble, S. Marshall, "Agile undercover: when customer's don't Collaborate", in: XP2010, Springer, Norway, pp. 73–87, 2010.

[111]. T. Chow, D. Cao, "A survey study of critical success factors in agile software Projects", J. Syst. Softw. 81, 961–971, 2008.

[112]. T. Dybå, T. Dingsoyr, "Empirical studies of agile software development: a Systematic review", Inf. Softw. Technol. 50, 833–859, 2008.

[113]. J. Highsmith, M. Fowler, "The agile manifesto", Softw. Dev. Mag. 9, 29–30, 2001.

[114]. A. Martin, R. Biddle, J. Noble, "The XP customer role: a grounded theory", in: AGILE2009, IEEE

Computer Society, Chicago, pp. 33–40, 2009.

[115]. Sutherland, J., Schwaber, K., Scrum, C. C. O., & Sutherl, C. J., "The scrum papers: Nuts, bolts, and origins of an agile process", 2011.

[116]. Sutherland, J., & Schwaber, K. "The Scrum Guide. The Definitive Guide to Scrum: The Rules of the Game", 2011.

[117]. Beck, K., "Embracing change with extreme programming". Computer, 32(10), 70-77, 1999.

[118]. Hoda, R., Noble, J., & Marshall, S., "The impact of inadequate customer collaboration on self-organizing Agile teams", Information and Software Technology, 53(5), 521–534. doi:10.1016/j.infsof.2010.10.009, 2011.

[119]. Sverrisdottir, H. S., Ingason, H. T., & Jonasson, H. I., "The Role of the Product Owner in Scrum-comparison between Theory and Practices". Procedia-Social and Behavioral Sciences, 119, 257-267, 2014.

[120]. Hoda, R., Noble, J., & Marshall, S., "The impact of inadequate customer collaboration on self-organizing Agile teams". Information and Software Technology, 53(5), 521-534, 2011.

[121]. Judy, K.H., Krumins-Beens, I., "Great Scrums Need Great Product Owners: Unbounded Collaboration and Collective Product Ownership". In: HICSS, Hawai, pp. 462–462, 2008.

[122]. Lohan, G., Lang, M., & Conboy, K., "Having a customer focus in agile software development", In Information Systems Development (pp. 441-453). Springer New York, 2011.

[123]. Miranda, E., & Bourque, P., "Agile monitoring using the line of balance", Journal of Systems and Software, 83(7), 1205-1215, 2010.

[124].Fowler, M., & Highsmith, J., "The agile manifesto. Software Development", 9(8), 28-35, 2001.

[125]. Ambler, S., "Agile database techniques: Effective strategies for the agile software develop", 2012.

[126]. Firesmith, D., "Specifying good requirements". Journal of Object Technology, 2(4), 77-87, 2003.

[127]. Amber D., "Focus Groups", Springer New York, 2013.

[128]. Inayat, I., Salim, S. S., Marczak, S., Daneva, M., & Shamshirband, S., "A systematic literature review on agile requirements engineering practices and challenges", Computers in Human Behavior, 2014.

[129]. Zhu, Y., "Requirements Engineering in an Agile Environment", 2009.

[130]. Daneva, M., van der Veen, E., Amrit, C., Ghaisas, S., Sikkel, K., Kumar, R., et al., "Agile requirements prioritization in large-scale outsourced system projects: An empirical study". Journal of Systems and Software, 86(5), 1333–1353, 2013.

[131]. Cao, L. C. L., & Ramesh, B., "Agile Requirements Engineering Practices: An Empirical Study", IEEE Software, 25(1), 60–67, 2008.

[132]. Bjarnason, E., Wnuk, K., & Regnell, B., "A Case Study on Benefits and Side-Effects of Agile Practices in Large-Scale Requirements Engineering", Agile RE 2011. Lancaster, UK: ACM, 2011.

[133]. Carlson, D., & Matuzic, P., "Practical Agile Requirements Engineering", 13th Annual Systems Engineering Conference. San Diego, CA, 2010.

[134]. Ramesh, B., Baskerville, R., & Cao, L., "Agile requirements engineering practices and challenges: an empirical study", Information Systems Journal, 20(5), 449–480, 2010.

[135]. De Lucia, A., & Qusef, A., "Requirements engineering in agile software development. Journal of Emerging Technologies in Web Intelligence", 2(3), 308–313, 2010.

[136]. Haugset, B., & Stalhane, T., "Automated Acceptance Testing as an Agile Requirements Engineering Practice", 45th Hawaii International Conference on System Sciences (pp. 5289–5298), 2012.

[137]. Boness, K., & Harrison, R., "Goal Sketching: Towards Agile Requirements Engineering", International Conference on Software Engineering Advances (pp. 1–6), 2007.

[139]. Yu, Y., & Sharp, H., "Analysing requirements in a case study of pairing", Workshop on Agile Requirements Engineering, July 2011.

[140]. Carlson, R., Matuzic, P. J., & Simons, R. L., "Applying scrum to stabilize systems engineering execution". CrossTalk, pp. 1–6, June 2012.

[141]. Jun, L., Qiuzhen, W., & Lin, G., "Application of Agile Requirement Engineering in Modest-Sized Information Systems Development", Second World Congress on Software Engineering, pp. 207–210, 2010.

[142]. Racheva, Z., Daneva, M., & Herrmann, A., "A Conceptual Model of Client-driven Agile Requirements Prioritization: Results of a Case Study", IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM), pp. 1–4, 2010.

[143]. Cardinal, M., "Addressing non-functional requirements with agile practices", 2011.

[144]. Tahir, A.; Ahmad, R., "Requirement Engineering Practices - An Empirical Study", Computational Intelligence and Software Engineering (CiSE), 2010 International Conference on, vol., no., pp.1,5, 10-12 Dec. 2010.

[145]. Solemon, B., Sahibuddin, S., & Ghani, A. A. A., "Requirements engineering problems and practices in software companies: An industrial survey", In Advances in Software Engineering (pp. 70-77), 2009.

[146]. Talbot, A., & Connor, A. M., "Requirements engineering current practice and capability in small and medium software development enterprises in New Zealand", arXiv preprint arXiv: 1407.6102, 2014.

[147]. T. Olsson, J. Doerr, T. Koenig and M. Ehresmann, "A Flexible and Pragmatic Requirements Engineering Framework for SME", Proceedings of the 1st International Workshop on Situational Requirements Engineering Processes, pp. 1-12, 2005.

[148]. Adam, S. Eisenbarth, M. "Lessons learned from best practice-oriented process improvement in Requirements Engineering–A glance into current industrial RE application" Softwaretechnik-Trends, 30(1), 2010.

[149]. Daneva, M., & Ahituv, N. "Evaluating cross-organizational ERP requirements engineering practices: a focus group study" In Research Challenges in Information Science (RCIS), 2010 Fourth International Conference on (pp. 279-286), 2012.

[150]. Daneva M, RJ. Wieringa, "Engineering the Coordination Requirements in Cross-organizational ERP Projects A Package of Good Practices", Proc of IEEE International Conference on Requirements Engineering, 2006.

[151]. Ambreen, A. Usman, M. Ikram, N. Bano, M, "Evidence in Software Requirement Engineering: A Systematic Literature Review", ICSEA: The Sixth International Conference on Software Engineering Advances, 2011.

[152]. B.A. Kitchenham, T. Dybå, and M. Jørgensen, "Evidence-Based Software Engineering," Proc. 26th Int'l Conf. Software Eng.(ICSE 2004), IEEE CS Press, pp. 273–281, 2004.

[153]. Dyba, T., Kitchenham, B. A., & Jorgensen, M., "Evidence-based software engineering for practitioners", Software, IEEE, 22(1), 58-65, 2005.

[154]. Niazi, M., & Babar, M. A., "Identifying high perceived value practices of CMMI level 2: an empirical study", Information and software technology, 51(8), 1231-1243, 2009.

[155]. O'Leary-Kelly, S. W., & J Vokurka, R., "The empirical assessment of construct validity", Journal of Operations Management, 16(4), pp. 387-405, 1998.

[156]. Svahnberg, M., Gorschek, T., Nguyen, T. T. L., & Nguyen, M., "Uni-REPM: a framework for requirements engineering process assessment". Requirements Engineering, 1-28, 2013.

[157]. Rees, M. J., "A feasible user story tool for agile software development?" In Software Engineering Conference, Ninth Asia-Pacific (pp. 22-30), 2002.

[158]. Gregorio, D. D., "How the Business Analyst supports and encourages collaboration on agile projects". In Systems Conference (SysCon), IEEE International (pp. 1-4), 2012.

Appendix

Appendix A: Results

ID	Requirements Documents Practice	Type of Assessment (n=56)			
		H	M	L	Z
RD1	Define a standard document structure	28	18	8	0
RD2	Explain how to use the document	18	18	12	6
RD3	Include a summary of the requirements	24	17	8	5
RD4	Make a business case for the system	21	19	9	5
RD5	Define specialized terms	23	16	11	4
RD6	Make document layout readable	24	21	8	1
RD7	Help readers find information	18	23	12	1
RD8	Make the document easy to change	20	23	9	2
Requirements analysis and negotiation practices					
RA1	Define system boundaries	25	18	8	3
RA2	Use checklists for requirements analysis	15	21	11	7
RA3	Provide software to support negotiations	7	14	17	16
RA4	Plan for conflicts and conflict resolution	10	20	13	11
RA5	Prioritise requirements	31	19	2	2
RA6	Classify requirements using a multi-dimensional approach	10	16	18	10
RA7	Use interaction matrices to find conflicts and overlaps	12	9	12	21
RA8	Assess requirements risks	17	20	12	5
RA9	Goals should be included in requirements analysis and negotiation activities	19	16	11	8
RA10	Use automatic tools in requirements analysis process	11	10	14	19
System Modelling Practices					
SM1	Develop complementary system models	H	M	L	Z
SM2	Model the system's environment	9	16	17	12
SM3	Model the system architecture	10	24	9	11
SM4	Use structured methods for system modelling	16	19	10	9

ID	Requirements Elicitation Practices	Type of Assessment (n=56)			
		H	M	L	Z
RE1	Assess System Feasibility	22	20	7	5
RE2	Be sensitive to organisational and political consideration	17	25	4	8
RE3	Identify and consult system stakeholders	37	10	6	1
RE4	Record requirements sources	25	16	7	6
RE5	Define the system's operating environment	17	22	12	3
RE6	Use business concerns to drive requirements elicitation	20	18	10	6
RE7	Look for domain constraints	20	18	13	3
RE8	Record requirements rationale	13	26	9	6
RE9	Collect requirements from multiple viewpoints	16	25	11	4
RE10	Prototype poorly understood requirements	19	15	14	6
RE11	Use scenarios to elicit requirements	14	16	18	6
RE12	Define operational processes	16	22	12	4
RE13	Reuse requirements	13	15	15	11
RE14	Facilitate requirement engineering with domain ontology	13	16	12	13
RE15	Use different techniques for requirement gathering like interviews, document analysis, use cases, scenarios, prototyping, brainstorming and focus groups	26	14	10	4
Describing Requirements Practices					
DR1	Define standard templates for describing requirements	25	12	9	8
DR2	Use languages simply and concisely	26	16	8	4
DR3	Use diagrams appropriately	28	17	7	2
DR4	Supplement natural language with other description of requirement	18	19	10	7
DR5	Specify requirements quantitatively	20	15	13	6
DR6	Have direct contact with customer to avoid unclear requirements	37	13	1	3
Requirements validation Practices					
RV1	Check that the requirements document meets your standards	19	20	8	7
RV2	Organise formal requirements inspections	15	18	14	7

SM5	Use a data dictionary	16	15	15	8		RV3	Use multi-disciplinary teams to review requirements	23	11	10	10
SM6	Document the links between stakeholder requirements and system models	10	20	13	11		RV4	Define validation checklists	17	16	13	8
							RV5	Use prototyping to animate requirements	11	19	12	12
Requirements Management Practices							RV6	Write a draft user manual	20	17	13	6
RM1	Uniquely identify each requirement	33	12	6	3		RV7	Propose requirements test cases	20	14	15	5
RM2	Define policies for requirements management	13	21	10	10		RV8	Paraphrase system models	11	13	14	16
RM3	Define traceability policies	11	18	13	12		**Requirements Engineering for Critical Systems Practices**					
RM4	Maintain a traceability manual	9	17	14	14		CS1	Create safety requirement checklists	18	11	8	17
RM5	Use a database to manage requirements	12	13	9	20		CS2	Involve external reviewers in the validation process	15	12	6	22
RM6	Define change management policies	20	9	14	11		CS3	Identify and analyse hazards	21	9	9	15
RM7	Identify global system requirements	17	16	14	7		CS4	Derive safety requirements from hazard analysis	18	10	10	16
RM8	Identify volatile requirements	14	17	12	11		CS5	Cross-check operational and functional requirements against safety requirement	22	5	10	17
RM9	Record rejected requirements	16	13	13	12		CS6	Specify systems using a formal specification	17	10	11	16
RM10	Have a dedicated role for requirements engineering activities	20	7	13	14		CS7	Collect incident experience	14	12	15	13
							CS8	Learn from incident experience	18	11	12	13
							CS9	Establish an organizational safety culture.	17	11	11	15

Appendix B: RE Practices based on Practitioners' experience

		Junior (N=8)				Intermediate(N=12)				Senior(N=34)			
ID	Practice	H	M	L	Z	H	M	L	Z	H	M	L	Z
RD1	Define a standard document structure	3	3	2	0	10	2	0	0	15	13	6	0
RD2	Explain how to use the document	2	4	2	0	5	4	3	0	11	10	7	6
RD3	Include a summary of the requirements	4	3	1	0	5	4	3	0	15	10	4	5
RD4	Make a business case for the system	2	4	1	1	5	6	1	0	14	9	7	4
RD5	Define specialized terms	2	3	3	0	6	3	1	2	15	10	7	2
RD6	Make document layout readable	2	5	1	0	5	6	1	0	17	10	6	1
RD7	Help readers find information	2	3	3	0	3	7	2	0	13	13	7	1
RD8	Make the document easy to change	2	4	2	0	4	5	2	1	14	14	5	1
RE1	Assess System Feasibility	2	4	2	0	7	5	0	0	13	11	5	5
RE2	Be sensitive to organisational and political consideration	2	4	1	1	3	8	0	1	12	13	3	6
RE3	Identify and consult system stakeholders	4	3	1	0	9	3	0	0	24	4	5	1
RE4	Record requirements sources	3	3	1	1	8	4	0	0	14	9	6	5
RE5	Define the system's operating environment	3	3	2	0	2	6	4	0	12	13	6	3
RE6	Use business concerns to drive requirements elicitation	3	3	1	1	5	4	3	0	12	11	6	5
RE7	Look for domain constraints	2	5	1	0	3	7	2	0	15	6	10	3
RE8	Record requirements rationale	1	6	0	1	2	8	1	1	10	12	8	4
RE9	Collect requirements from multiple viewpoints	4	2	1	1	5	6	1	0	12	11	7	4
RE10	Prototype poorly understood requirements	2	1	5	0	4	3	3	2	13	11	6	4
RE11	Use scenarios to elicit requirements	3	2	3	0	3	2	5	2	8	12	10	4
RE12	Define operational processes	3	3	0	1	4	2	5	1	9	16	7	2
RE13	Reuse requirements	3	3	0	2	2	3	4	3	8	9	11	6
RE14	Facilitate requirement engineering with domain ontology	2	3	1	2	2	5	3	2	9	8	8	9
RE15	Use different techniques for requirement gathering like		2	3	0	3	5	4	0	20	7	3	4

	interviews, document analysis, use cases, scenarios, prototyping, brainstorming and focus groups.	3											
RA1	Define system boundaries	5	2	0	1	6	4	2	0	14	12	6	2
RA2	Use checklists for requirements analysis	3	3	1	1	4	6	1	1	8	12	9	5
RA3	Provide software to support negotiations	2	2	3	1	1	4	4	3	4	8	10	12
RA4	Plan for conflicts and conflict resolution	2	3	2	1	2	5	4	1	6	12	7	9
RA5	Prioritise requirements	5	2	1	0	8	4	0	0	18	13	1	2
RA6	Classify requirements using a multi-dimensional approach	3	1	2	2	3	5	2	2	4	10	14	6
RA7	Use interaction matrices to find conflicts and overlaps	3	3	0	2	4	1	1	6	5	5	11	13
RA8	Assess requirements risks	0	3	3	2	5	6	1	0	12	11	8	3
RA9	Goals should be included in analysis and negotiation activities	3	1	3	1	4	4	2	2	12	11	6	5
RA10	Use automatic tools in requirements analysis process	3	2	0	3	4	0	2	6	4	8	12	10
DR1	Define standard templates for describing requirements	4	2	1	1	9	2	0	1	12	8	8	6
DR2	Use languages simply and concisely	3	3	2	0	5	4	2	1	18	9	4	3
DR3	Use diagrams appropriately	4	3	1	0	6	5	1	0	18	9	5	2
DR4	Supplement natural language with other description of reqs.	2	3	2	1	2	6	3	1	14	10	5	5
DR5	Specify requirements quantitatively	4	1	2	1	4	4	3	1	12	10	8	4
DR6	Have direct contact with customer to avoid unclear requirements	5	3	0	0	10	2	0	0	22	8	1	3
SM1	Develop complementary system models	2	2	2	2	1	4	5	2	6	10	10	8
SM2	Model the system's environment	2	4	1	1	1	7	3	1	7	13	5	9
SM3	Model the system architecture	4	4	0	0	1	7	3	1	11	8	7	8
SM4	Use structured methods for system modelling	3	1	3	1	4	5	2	1	9	9	10	6
SM5	Use a data dictionary	0	4	2	2	2	4	3	3	8	12	8	6
SM6	Document the links between stakeholder reqs and system models	2	3	2	1	5	2	3	2	11	6	9	8
RV1	Check that the requirements document meets your standards	4	3	0	1	5	5	1	1	10	12	7	5
RV2	Organise formal requirements inspections	2	3	2	1	4	4	4	0	9	11	8	6
RV3	Use multi-disciplinary teams to review requirements	3	1	3	1	6	3	1	2	14	7	6	7
RV4	Define validation checklists	3	3	1	1	4	4	2	2	10	9	10	5
RV5	Use prototyping to animate requirements	2	3	2	1	2	5	1	4	7	11	9	7
RV6	Write a draft user manual	2	4	2	0	2	3	4	3	9	5	13	7
RV7	Propose requirements test cases	3	2	3	0	5	1	5	1	12	11	7	4
RV8	Paraphrase system models	3	2	2	1	4	2	4	2	4	9	8	13
RM1	Uniquely identify each requirement	5	1	2	0	7	5	0	0	21	6	4	3
RM2	Define policies for requirements management	2	4	0	2	3	5	4	0	8	12	6	8
RM3	Define traceability policies	1	4	2	1	3	4	3	2	7	10	8	9
RM4	Maintain a traceability manual	0	5	2	1	2	5	3	2	7	7	9	11
RM5	Use a database to manage requirements	1	3	1	3	2	4	2	4	9	6	6	13
RM6	Define change management policies	2	2	3	1	5	2	3	2	13	5	8	8
RM7	Identify global system requirements	2	5	1	0	3	4	3	2	12	7	10	5
RM8	Identify volatile requirements	2	4	1	1	3	5	3	1	9	8	8	9
RM9	Record rejected requirements	2	3	2	1	5	3	2	2	9	7	9	9
RM10	Have a dedicated role for requirement engineering activities	3	1	1	3	7	1	3	1	10	5	9	10
CS1	Create safety requirement checklists	2	3	0	3	3	3	2	4	13	5	6	10
CS2	Involve external reviewers in the validation process	1	3	1	3	3	3	1	5	11	6	4	13
CS3	Identify and analyse hazards	3	2	0	3	5	1	3	3	13	6	6	9
CS4	Derive safety requirements from hazard analysis	2	2	1	3	4	2	3	3	12	6	6	10
CS5	Cross-check operational/functional reqs against safety reqs	2	1	2	3	6	0	3	3	14	4	5	11
CS6	Specify systems using a formal specification	2	3	0	3	4	2	3	3	11	5	8	10
CS7	Collect incident experience	1	2	2	3	2	3	3	4	11	7	10	6
CS8	Learn from incident experience	1	3	1	3	3	4	2	3	14	4	9	7
CS9	Establish an organizational safety culture.	1	3	1	3	3	3	2	4	13	5	8	8

Appendix C: RE Practices based on Company Size

		Small(N=10)				Intermediate(N=13)				Large (N=28)			
ID	Practice	H	M	L	Z	H	M	L	Z	H	M	L	Z
RD1	Define a standard document structure	3	5	2	0	1	3	3	0	17	9	2	0
RD2	Explain how to use the document	3	3	3	1	6	4	1	2	8	9	8	3
RD3	Include a summary of the requirements	6	0	3	1	3	6	2	2	13	10	3	2
RD4	Make a business case for the system	3	6	1	0	7	6	0	0	11	5	8	4
RD5	Define specialized terms	4	3	1	2	4	5	3	1	14	6	7	1
RD6	Make document layout readable	6	3	1	0	7	5	1	0	10	12	6	0
RD7	Help readers find information	5	2	2	1	5	5	3	0	8	13	7	0
RD8	Make the document easy to change	5	3	1	1	8	3	2	0	7	16	5	0
RE1	Assess System Feasibility	3	4	2	1	5	5	2	1	14	9	2	3
RE2	Be sensitive to organisational and political consideration	2	4	2	2	6	3	4	0	8	16	2	2
RE3	Identify and consult system stakeholders	7	2	0	1	7	4	2	0	21	4	3	0
RE4	Record requirements sources	4	3	2	1	5	3	2	3	14	9	3	2
RE5	Define the system's operating environment	3	3	3	1	4	4	4	1	10	13	4	1
RE6	Use business concerns to drive requirements elicitation	3	5	1	1	5	3	3	2	12	9	5	2
RE7	Look for domain constraints	5	2	2	1	4	6	2	1	10	9	8	1
RE8	Record requirements rationale	3	5	0	2	2	6	2	3	7.	13	7	1
RE9	Collect requirements from multiple viewpoints	4	4	2	0	4	6	1	2	10	9	6	3
RE10	Prototype poorly understood requirements	7	0	2	1	4	4	4	1	8	10	7	3
RE11	Use scenarios to elicit requirements	2	6	1	1	2	5	5	1	10	5	10	3
RE12	Define operational processes	1	6	1	2	4	7	1	1	10	8	9	1
RE13	Reuse requirements	3	1	2	4	1	4	4	4	9	7	9	3
RE14	Facilitate requirement engineering with domain ontology	2	3	0	5	2	4	4	3	9	7	7	5
RE15	Use different techniques for requirement gathering like interviews, document analysis, use cases, scenarios, prototyping, brainstorming and focus groups.	6	0	3	1	5	5	3	0	13	9	4	2
RA1	Define system boundaries	6	1	2	1	6	3	3	1	11	13	3	1
RA2	Use checklists for requirements analysis	2	4	2	2	4	5	2	2	8	10	7	3
RA3	Provide software to support negotiations	1	1	4	4	0	4	6	3	6	7	7	8
RA4	Plan for conflicts and conflict resolution	3	2	2	3	1	7	3	2	5	10	8	5
RA5	Prioritise requirements	5	3	1	1	8	5	0	0	17	10	1	0
RA6	Classify requirements using a multi-dimensional approach	2	0	4	4	0	7	4	2	7	8	10	3
RA7	Use interaction matrices to find conflicts and overlaps	2	1	2	5	2	3	4	4	7	4	6	11
RA8	Assess requirements risks	2	4	2	2	5	3	4	1	9	11	6	2
RA9	Goals should be included in analysis and negotiation activities	3	2	2	3	5	4	4	0	10	9	5	4
RA10	Use automatic tools in requirements analysis process	2	1	1	6	1	2	6	4	7	6	7	8
DR1	Define standard templates for describing requirements	3	3	1	3	5	3	3	2	15	6	5	2
DR2	Use languages simply and concisely	5	1	3	1	8	3	2	0	13	10	3	2
DR3	Use diagrams appropriately	6	2	1	1	7	4	2	0	14	10	4	0
DR4	Supplement natural language with other description of reqs.	4	3	1	2	5	4	2	2	9	11	6	2
DR5	Specify requirements quantitatively	4	1	3	2	4	5	3	1	12	7	7	2
DR6	Have direct contact with customer to avoid unclear requirements	7	2	0	1	9	3	0	1	18	8	1	1
SM1	Develop complementary system models	3	2	2	3	0	5	5	3	5	8	10	5
SM2	Model the system's environment	2	4	1	3	0	7	3	3	8	11	5	4
SM3	Model the system architecture	3	2	2	3	3	6	2	2	8	11	6	3

SM4	Use structured methods for system modelling	2	3	2	3	3	5	3	2	9	7	10	2
SM5	Use a data dictionary	1	4	0	5	2	6	4	1	7	9	8	4
SM6	Document links between stakeholder reqs, and system models	3	2	1	4	3	2	5	3	12	5	7	4
RV1	Check that the requirements document meets your standards	3	2	2	3	4	4	2	3	11	12	4	1
RV2	Organise formal requirements inspections	3	2	3	2	3	5	3	2	9	9	8	2
RV3	Use multi-disciplinary teams to review requirements	4	1	2	3	4	3	4	2	13	7	4	4
RV4	Define validation checklists	3	2	1	4	5	4	2	2	8	9	10	1
RV5	Use prototyping to animate requirements	3	4	0	3	3	2	5	3	5	11	7	5
RV6	Write a draft user manual	4	2	1	3	1	6	5	1	8	2	13	5
RV7	Propose requirements test cases	3	3	1	3	4	4	4	1	11	7	10	0
RV8	Paraphrase system models	1	5	0	4	3	1	5	4	6	7	8	7
RM1	Uniquely identify each requirement	5	2	2	1	6	4	3	0	20	5	1	2
RM2	Define policies for requirements management	4	1	3	2	2	4	3	4	7	15	4	2
RM3	Define traceability policies	1	3	2	4	2	4	4	3	7	10	7	4
RM4	Maintain a traceability manual	1	3	2	4	1	5	3	4	6	8	9	5
RM5	Use a database to manage requirements	3	2	2	3	0	6	3	4	9	3	4	12
RM6	Define change management policies	3	0	4	3	4	3	4	2	13	4	6	5
RM7	Identify global system requirements	1	5	1	3	5	3	4	1	11	6	9	2
RM8	Identify volatile requirements	1	5	2	2	4	5	2	2	9	5	8	6
RM9	Record rejected requirements	3	1	1	5	4	3	4	2	9	7	7	5
RM10	Have a dedicated role for requirement engineering activities	2	2	2	4	4	2	3	4	14	2	6	6
CS1	Create safety requirement checklists	4	2	0	4	4	5	2	2	9	3	6	10
CS2	Involve external reviewers in the validation process	3	2	0	5	3	5	3	2	8	5	3	12
CS3	Identify and analyse hazards	4	2	0	4	6	3	2	2	9	4	7	8
CS4	Derive safety requirements from hazard analysis	4	2	0	4	4	3	4	2	9	4	6	9
CS5	Cross-check operational/functional reqs against safety req.	3	1	1	5	7	1	3	2	12	1	6	9
CS6	Specify systems using a formal specification	3	1	1	5	4	4	2	3	10	4	7	7
CS7	Collect incident experience	2	1	3	4	5	3	3	2	7	6	9	6
CS8	Learn from incident experience	5	1	1	3	6	2	3	2	7	6	8	7
CS9	Establish an organizational safety culture.	4	2	3	1	6	2	3	2	7	5	7	9

Appendix D: RE Practices based on Company Type

		Multinational(N=40)				National(N=14)			
ID	Practice	H	M	L	Z	H	M	L	Z
RD1	Define a standard document structure	18	15	6	1	5	5	2	0
RD2	Explain how to use the document	13	14	11	2	5	6	2	0
RD3	Include a summary of the requirements	20	16	2	2	3	7	3	0
RD4	Make a business case for the system	16	16	5	3	6	5	2	0
RD5	Define specialized terms	13	17	8	2	3	7	3	0
RD6	Make document layout readable	15	21	2	2	7	4	2	0
RD7	Help readers find information	11	18	8	3	1	10	2	0
RD8	Make the document easy to change	10	25	1	4	3	8	2	0
RE1	Assess System Feasibility	16	20	2	2	5	5	3	0
RE2	Be sensitive to organisational and political consideration	14	15	10	1	4	5	3	1
RE3	Identify and consult system stakeholders	25	12	2	1	9	3	1	0
RE4	Record requirements sources	22	15	2	1	5	5	3	0
RE5	Define the system's operating environment	18	17	4	1	7	3	2	1
RE6	Use business concerns to drive requirements elicitation	15	18	3	4	4	6	3	0
RE7	Look for domain constraints	17	11	10	2	6	5	1	1
RE8	Record requirements rationale	14	8	14	4	1	8	3	1
RE9	Collect requirements from multiple viewpoints	13	16	8	3	3	7	2	1
RE10	Prototype poorly understood requirements	9	17	10	4	2	6	5	0
RE11	Use scenarios to elicit requirements	12	20	4	4	4	4	4	1
RE12	Define operational processes	17	15	7	1	3	7	2	1
RE13	Reuse requirements	9	21	7	3	2	7	3	1
RA1	Define system boundaries	22	11	5	2	6	6	1	0

RA2	Use checklists for requirements analysis	13	12	12	3	4	6	2	1
RA3	Provide software to support negotiations	9	12	15	4	1	4	6	2
RA4	Plan for conflicts and conflict resolution	14	14	10	2	3	4	4	2
RA5	Prioritise requirements	25	13	1	1	4	8	1	0
RA6	Classify requirements using a multi-dimensional approach	12	11	14	3	2	4	7	0
RA7	Use interaction matrices to find conflicts and overlaps	6	9	18	7	1	4	7	1
RA8	Assess requirements risks	15	14	7	4	3	4	5	1
DR1	Define standard templates for describing requirements	22	12	4	2	6	4	2	1
DR2	Use languages simply and concisely	17	18	3	2	4	8	1	0
DR3	Use diagrams appropriately	20	10	5	5	5	6	2	0
DR4	Supplement natural language with other description of req.	15	17	6	2	4	7	2	0
DR5	Specify requirements quantitatively	11	13	14	2	1	8	3	1
SM1	Develop complementary system models	8	14	13	5	3	6	3	1
SM2	Model the system's environment	9	13	14	4	3	4	5	1
SM3	Model the system architecture	13	20	4	3	5	4	2	2
SM4	Use structured methods for system modelling	11	14	9	6	4	5	3	1
SM5	Use a data dictionary	13	10	12	5	2	6	4	1
SM6	Document links between stakeholder reqs and system models	11	11	14	4	2	6	4	1
RV1	Check that the requirements document meets your standards	21	11	5	3	7	3	2	1
RV2	Organise formal requirements inspections	11	12	13	4	4	5	2	2
RV3	Use multi-disciplinary teams to review requirements	12	12	10	6	4	3	5	1
RV4	Define validation checklists	13	10	12	5	4	5	3	1
RV5	Use prototyping to animate requirements	7	13	13	7	2	5	5	1
RV6	Write a draft user manual	17	10	9	4	3	5	4	1
RV7	Propose requirements test cases	17	15	6	2	3	5	3	2
RV8	Paraphrase system models	5	15	16	4	0	3	8	2
RM1	Uniquely identify each requirement	20	14	4	2	8	2	2	1
RM2	Define policies for requirements management	14	12	11	3	3	6	4	0
RM3	Define traceability policies	13	14	10	3	4	3	4	2
RM4	Maintain a traceability manual	12	10	13	5	3	3	6	1
RM5	Use a database to manage requirements	15	9	8	8	1	6	4	2
RM6	Define change management policies	15	12	7	6	2	8	2	1
RM7	Identify global system requirements	16	9	9	6	2	6	2	3
RM8	Identify volatile requirements	8	14	12	6	3	2	5	3
RM9	Record rejected requirements	7	9	16	8	3	1	6	3
CS1	Create safety requirement checklists	13	12	7	8	2	5	1	5
CS2	Involve external reviewers in the validation process	8	15	9	8	1	4	2	6
CS3	Identify and analyse hazards	14	9	12	5	2	5	1	5
CS4	Derive safety requirements from hazard analysis	11	10	11	8	1	6	2	4
CS5	Cross-check operational/ functional reqs against safety req.	13	9	10	8	2	2	5	4
CS6	Specify systems using a formal specification	10	16	9	5	2	3	2	6
CS7	Collect incident experience	12	11	10	7	0	5	3	5
CS8	Learn from incident experience	16	11	6	7	2	5	2	4
CS9	Establish an organizational safety culture.	10	16	9	5	0	6	3	4

Appendix E: RE Practices based on Software Development Process Type

		Agile(N=31)				Traditional(N=19)			
ID	Practice	H	M	L	Z	H	M	L	Z
RD1	Define a standard document structure	17	9	5	0	7	9	3	0
RD2	Explain how to use the document	12	8	6	5	4	8	6	1
RD3	Include a summary of the requirements	11	11	6	3	10	5	2	2
RD4	Make a business case for the system	14	8	6	3	5	9	3	2
RD5	Define specialized terms	11	10	7	3	9	5	4	1
RD6	Make document layout readable	13	13	3	0	7	6	5	1
RD7	Help readers find information	12	10	9	0	5	10	3	1
RD8	Make the document easy to change	14	11	5	1	5	9	4	1
RE1	Assess System Feasibility	14	11	3	3	5	8	4	2
RE2	Be sensitive to organisational and political consideration	9	16	1	5	6	7	3	3
RE3	Identify and consult system stakeholders	24	6	1	0	10	3	5	1
RE4	Record requirements sources	14	11	4	2	7	5	3	4
RE5	Define the system's operating environment	10	12	8	1	5	8	4	2

RE6	Use business concerns to drive requirements elicitation	12	11	5	3	6	5	5	3
RE7	Look for domain constraints	13	11	5	2	5	5	8	1
RE8	Record requirements rationale	9	15	5	2	3	8	4	4
RE9	Collect requirements from multiple viewpoints	12	12	5	2	5	7	4	3
RE10	Prototype poorly understood requirements	14	10	6	1	4	4	6	5
RE11	Use scenarios to elicit requirements	12	10	7	2	1	6	8	4
RE12	Define operational processes	11	16	4	0	2	5	8	3
RE13	Reuse requirements	7	8	9	7	4	6	5	4
RE14	Facilitate requirement engineering with domain ontology	10	8	7	6	2	5	5	7
RE15	Use different techniques for requirement gathering like interviews, document analysis, use cases, scenarios, prototyping, brainstorming and focus groups	16	10	5	0	7	3	5	4
RA1	Define system boundaries	14	12	5	0	8	5	3	3
RA2	Use checklists for requirements analysis	11	11	5	4	1	9	6	3
RA3	Provide software to support negotiations	5	7	11	8	6	1	4	8
RA4	Plan for conflicts and conflict resolution	7	11	9	4	3	5	4	7
RA5	Prioritise requirements	20	11	0	0	7	8	2	2
RA6	Classify requirements using a multi-dimensional approach	6	10	9	6	2	5	8	4
RA7	Use interaction matrices to find conflicts and overlaps	6	7	7	11	3	2	5	9
RA8	Assess requirements risks	12	10	8	1	4	7	4	4
RA9	Goals should be included in analysis and negotiation activities	15	8	6	2	2	7	5	5
RA10	Use automatic tools in requirements analysis process	7	5	11	8	2	5	2	10
DR1	Define standard templates for describing requirements	16	6	6	3	5	6	3	5
DR2	Use languages simply and concisely	19	10	2	0	5	4	6	4
DR3	Use diagrams appropriately	19	10	2	0	6	6	5	2
DR4	Supplement natural language with other description of req.	14	11	4	2	3	5	6	5
DR5	Specify requirements quantitatively	13	11	7	0	4	3	6	6
DR6	Have direct contact with customers to avoid unclear requirements	22	9	0	0	11	4	1	3
SM1	Develop complementary system models	5	9	11	6	3	5	5	6
SM2	Model the system's environment	6	15	5	5	3	6	4	6
SM3	Model the system architecture	10	10	6	5	4	7	4	4
SM4	Use structured methods for system modelling	9	9	9	4	4	5	6	4
SM5	Use a data dictionary	8	10	8	5	2	6	5	6
SM6	Document links between stakeholder reqs and system models	12	12	7	7	4	5	6	4
RV1	Check that the requirements document meets your standards	12	12	4	3	4	7	4	4
RV2	Organise formal requirements inspections	9	12	7	3	5	3	7	4
RV3	Use multi-disciplinary teams to review requirements	14	6	7	4	7	3	3	6
RV4	Define validation checklists	10	9	9	3	5	5	4	5
RV5	Use prototyping to animate requirements	10	8	9	4	1	8	2	8
RV6	Write a draft user manual	7	8	12	4	5	3	5	6
RV7	Propose requirements test cases	13	10	7	1	4	4	7	4
RV8	Paraphrase system models	7	8	7	9	2	4	6	7
RM1	Uniquely identify each requirement	21	6	2	2	8	6	4	1
RM2	Define policies for requirements management	7	14	5	5	5	6	5	3
RM3	Define traceability policies	6	11	8	6	4	4	5	6
RM4	Maintain a traceability manual	5	11	7	8	4	2	7	6
RM5	Use a database to manage requirements	7	8	6	10	4	4	2	9
RM6	Define change management policies	12	5	9	5	6	2	5	6
RM7	Identify global system requirements	12	7	8	4	3	7	6	3
RM8	Identify volatile requirements	9	9	8	5	3	6	4	6
RM9	Record rejected requirements	8	7	7	9	6	4	6	3
RM10	Have a dedicated role for requirement engineering activities	12	4	7	8	6	1	6	6
CS1	Create safety requirement checklists	11	8	3	9	5	2	4	8
CS2	Involve external reviewers in the validation process	10	8	2	11	4	2	4	9
CS3	Identify and analyse hazards	13	6	4	8	6	1	5	7
CS4	Derive safety requirements from hazard analysis	10	7	5	9	5	2	5	7
CS5	Cross-check operational/ functional reqs against safety req.	15	2	4	10	4	2	6	7
CS6	Specify systems using a formal specification	10	7	5	9	4	2	6	7
CS7	Collect incident experience	8	7	9	7	4	4	5	6
CS8	Learn from incident experience	10	6	8	7	6	3	4	6
CS9	Establish an organizational safety culture.	9	8	7	7	5	2	4	8

Appendix E: Based on participants region

ID	Practice	Asia(N=19)			Europe(N=17)			America (N=12)			Australiana(N=4)			Africa(N=2)		
		H	M	L	H	M	L	H	M	L	H	M	L	H	M	L
RD1	Define a standard document structure	11	4	4	0	0	0	0	4	2	0	3	1	0	0	2	0	0
RD2	Explain how to use the document	5	7	6	1	0	0	4	5	2	4	4	1	3	2	2	0	0
RD3	Include a summary of the requirements	8	6	3	2	0	0	5	5	1	7	3	0	2	2	2	0	0
RD4	Make a business case for the system	6	8	4	1	0	0	7	3	1	6	1	2	3	2	2	0	0
RD5	Define specialized terms	8	6	5	2	0	0	5	5	1	7	3	1	1	2	2	0	0
RD6	Make document layout readable	9	9	1	0	0	0	5	5	1	7	3	2	0	2	2	0	0
RD7	Help readers find information	5	11	3	0	5	0	5	5	1	6	3	3	0	1	3	0	0
RD8	Make the document easy to change	8	10	2	1	5	0	7	2	2	5	5	1	1	2	2	0	0
RE1	Assess System Feasibility;	7	7	3	2	6	0	7	2	2	5	5	1	1	3	1	0	0
RE2	Be sensitive to organizational and political consideration	4	11	1	3	7	1	2	3	4	2	3	1	3	0	0	0	0
RE3	Identify and consult system stakeholders	13	5	1	0	12	1	3	1	7	4	1	0	3	0	1	0	0
RE4	Record requirements sources	9	6	1	3	6	6	3	2	6	2	3	1	3	1	0	0	0
RE5	Define the system's operating environment	4	9	5	1	8	5	2	2	4	4	4	0	3	0	1	0	0
RE6	Use business concerns to drive requirements elicitation	7	6	4	2	6	8	2	1	3	3	3	3	2	1	1	0	0
RE7	Look for domain constraints	5	8	6	0	7	5	4	1	5	3	2	2	2	1	1	0	0
RE8	Record requirements rationale	3	8	5	3	5	8	3	1	4	6	0	2	0	4	0	0	0
RE9	Collect requirements from multiple viewpoints	5	8	4	2	8	5	2	2	4	5	2	1	3	1	0	0	0
RE10	Prototype poorly understood requirements	4	5	6	4	8	3	6	0	5	5	1	1	1	2	1	0	0
RE11	Use scenarios to elicit requirements	5	4	7	3	4	7	5	1	4	4	2	2	1	0	3	0	1
RE12	Define operational processes	3	9	5	2	5	8	3	1	6	3	2	1	1	2	1	0	1
RE13	Reuse requirements	2	8	6	5	4	6	3	4	5	0	5	2	0	3	1	0	2
RE14	Facilitate requirement engineering with domain ontology	3	8	4	5	4	5	3	5	5	1	4	2	0	4	0	0	1
RE15	Use different techniques for requirement gathering like interviews, document analysis, use cases, scenarios, prototyping, brainstorming and focus groups	7	8	0	0	0	5	3	3	9	1	1	1	2	2	0	0	2
RA1	Define system boundaries	10	4	3	2	6	7	2	2	6	4	2	0	1	2	1	0	1
RA2	Use checklists for requirements analysis	4	8	4	3	5	6	3	3	4	3	1	1	1	2	0	0	1
RA3	Provide software to support negotiations	2	7	2	8	2	7	6	2	3	5	1	0	1	3	0	0	1
RA4	Plan for conflicts and conflict resolution	4	6	5	4	2	6	3	5	3	5	3	1	0	3	1	0	0
RA5	Prioritise requirements	11	6	1	1	8	7	1	1	6	6	0	0	4	0	0	2	0
RA6	Classify requirements using a multi-dimensional approach	3	6	5	5	1	5	8	3	4	3	3	2	1	2	1	0	1
RA7	Use interaction matrices to find conflicts and overlaps	4	5	2	8	2	2	6	7	4	1	3	4	1	1	0	2	1
RA8	Assess requirements risks	5	7	5	2	6	6	2	3	5	4	3	0	0	3	1	0	1
RA9	Goals should be included in analysis and negotiation activities	5	6	3	2	8	3	1	5	4	5	3	0	1	2	0	1	0
RA10	Use automatic tools in requirements analysis process	4	1	6	8	2	4	4	7	3	4	4	1	1	2	0	1	0
DR1	Define standard templates for describing requirements	9	3	2	5	8	4	3	2	4	4	4	0	3	1	0	0	0
DR2	Use languages simply and concisely	9	5	3	2	9	3	4	1	6	5	0	1	1	3	0	0	1
DR3	Use diagrams appropriately	9	7	2	1	9	4	3	1	7	4	1	0	2	2	0	0	1
DR4	Supplement natural language with other description of req.	5	9	1	4	7	4	5	1	5	4	1	2	0	2	0	1	0
DR5	Specify requirements quantitatively	5	4	7	3	6	5	3	3	6	5	1	0	2	1	0	1	0

DR6	Have direct contact with customers to avoid unclear requirements	10	2	0	1	0	9	0	2	8	3	1	0	3	1	0	0	1	1	0	0
SM1	Develop complementary system models	3	8	3	5	0	5	8	8	4	3	4	1	1	0	3	0	1	0	1	0
SM2	Model the system's environment	2	9	4	4	3	9	0	5	4	4	2	2	0	2	2	0	1	0	1	0
SM3	Model the system architecture	4	8	3	4	5	4	5	3	5	4	1	2	1	3	0	0	1	0	1	0
SM4	Use structured methods for system modelling	7	4	5	3	3	4	6	4	3	6	2	1	2	1	1	0	1	0	1	0
SM5	Use a data dictionary	3	7	5	4	3	5	4	5	3	6	2	1	0	2	1	1	1	0	1	0
SM6	Document links between stakeholder reqs and system models	5	3	8	3	4	5	2	6	7	1	2	2	1	2	1	0	1	0	1	0
RV1	Check that the requirements document meets your standards	8	6	2	3	4	7	3	3	4	4	3	1	2	2	0	0	1	1	0	0
RV2	Organise formal requirements inspections	4	6	6	3	3	8	3	3	6	1	4	1	1	2	1	0	1	1	0	0
RV3	Use multi-disciplinary teams to review requirements	5	6	3	5	6	2	5	4	8	2	1	1	3	0	1	0	1	1	0	0
RV4	Define validation checklists	6	4	5	4	5	4	4	4	5	3	4	0	0	4	0	0	1	1	0	0
RV5	Use prototyping to animate requirements	4	8	2	5	4	2	5	6	2	5	4	1	0	3	1	0	1	1	0	0
RV6	Write a draft user manual	3	5	7	4	4	3	5	5	5	2	4	1	0	1	3	0	1	1	0	0
RV7	Propose requirements test cases	6	6	4	3	7	4	5	1	5	3	3	1	1	0	3	0	1	1	0	0
RV8	Paraphrase system models	4	7	3	5	2	2	5	8	3	4	2	3	1	0	3	0	1	0	1	0
RM1	Uniquely identify each requirement	10	4	4	1	7	6	2	2	11	1	0	0	4	0	0	0	1	1	0	0
RM2	Define policies for requirements management	4	7	4	4	1	7	4	5	6	4	2	0	1	2	0	1	1	1	0	0
RM3	Define traceability policies	3	7	4	5	3	4	5	5	3	4	3	2	1	3	0	0	1	0	1	0
RM4	Maintain a traceability manual	1	8	5	5	3	3	4	7	3	3	4	2	1	3	0	0	1	0	1	0
RM5	Use a database to manage requirements	4	3	3	9	2	5	3	7	5	3	2	2	0	2	1	1	1	0	0	1
RM6	Define change management policies	8	3	3	5	5	3	5	4	5	2	4	1	1	1	2	0	1	0	0	1
RM7	Identify global system requirements	6	5	4	4	7	3	5	2	3	5	3	1	0	3	1	0	1	0	1	0
RM8	Identify volatile requirements	4	7	3	5	6	3	5	3	3	4	3	2	0	3	1	0	1	0	0	1
RM9	Record rejected requirements	4	6	4	5	5	2	4	6	4	3	4	1	2	2	0	0	1	0	1	0
RM10	Have a dedicated role for requirement engineering activities	8	2	4	5	5	2	5	5	4	2	3	3	2	1	0	1	1	0	1	0
CS1	Create safety requirement checklists	4	3	4	8	3	6	1	7	9	1	1	1	1	1	1	1	1	0	1	0
CS2	Involve external reviewers in the validation process	2	5	3	9	3	4	1	9	8	3	0	1	1	0	2	1	1	0	1	0
CS3	Identify and analyse hazards	6	1	6	6	4	5	1	7	9	1	1	1	1	2	0	1	1	0	1	0
CS4	Derive safety requirements from hazard analysis	2	5	5	7	3	5	1	7	10	0	1	1	2	0	1	1	1	0	1	0
CS5	Cross-check operational/functional reqs against safety req.	5	2	5	7	6	1	2	8	9	1	1	1	1	1	1	1	1	0	1	0
CS6	Specify systems using a formal specification	3	4	4	8	5	2	4	6	7	1	2	1	1	2	0	1	1	1	0	0
CS7	Collect incident experience	2	5	5	7	5	3	5	4	6	2	3	1	0	1	2	1	1	1	0	0
CS8	Learn from incident experience	3	5	4	7	6	3	4	4	7	1	3	1	0	1	2	1	1	1	0	0
CS9	Establish an organizational safety culture.	3	5	4	7	5	4	3	5	7	1	2	2	0	1	2	1	1	0	1	0

Appendix G: Comparison with GSD study

		GSD Study(N=56)				Our Study(N=54)			
ID	Practice	H	M	L	Z	H	M	L	Z
RD1	Define a standard document structure	24	22	9	1	28	18	8	0
RD2	Explain how to use the document	16	22	16	2	18	18	12	6
RD3	Include a summary of the requirements	26	23	5	2	24	17	8	5
RD4	Make a business case for the system	19	25	9	3	21	19	9	5
RD5	Define specialized terms	18	22	14	2	23	16	11	4
RD6	Make document layout readable	23	25	6	2	24	21	8	1
RD7	Help readers find information	13	29	10	4	18	23	12	1
RD8	Make the document easy to change	13	35	4	4	20	23	9	2
RE1	Assess System Feasibility	21	27	6	2	22	20	7	5
RE2	Be sensitive to organisational and political consideration	19	21	4	2	17	25	4	8
RE3	Identify and consult system stakeholders	36	16	3	1	37	10	6	1
RE4	Record requirements sources	27	22	6	1	25	16	7	6
RE5	Define the system's operating environment	26	22	6	2	17	22	12	3
RE6	Use business concerns to drive requirements elicitation	20	25	7	4	20	18	10	6
RE7	Look for domain constraints	23	19	11	3	20	18	13	3
RE8	Record requirements rationale	15	18	18	5	13	26	9	6
RE9	Collect requirements from multiple viewpoints	16	25	11	4	21	19	9	5
RE10	Prototype poorly understood requirements	11	24	16	5	19	15	14	6
RE11	Use scenarios to elicit requirements	16	26	9	5	14	16	18	6
RE12	Define operational processes	20	23	10	3	16	22	12	4
RE13	Reuse requirements	11	29	11	5	13	15	15	11
RE14	Facilitate requirement engineering with domain ontology					13	16	12	13
RE15	Use different techniques for requirement gathering like interviews, document analysis, use cases, scenarios, prototyping, brainstorming and focus groups					26	14	10	4
RA1	Define system boundaries	29	18	7	2	25	18	8	3
RA2	Use checklists for requirements analysis	18	19	15	4	15	21	11	7
RA3	Provide software to support negotiations	10	17	23	6	1	14	17	16
RA4	Plan for conflicts and conflict resolution	17	19	16	4	10	20	13	11
RA5	Prioritise requirements	30	23	2	1	31	19	2	2
RA6	Classify requirements using a multi-dimensional approach	14	17	21	4	10	16	18	10
RA7	Use interaction matrices to find conflicts and overlaps	7	14	25	10	12	9	12	21
RA8	Assess requirements risks	18	19	14	5	17	20	12	5
RA9	Goals should be included in analysis and negotiation activities					19	16	11	8
RA10	Use automatic tools in requirements analysis process					11	10	14	19
DR1	Define standard templates for describing requirements	28	17	8	3	25	12	9	8
DR2	Use languages simply and concisely	21	28	5	2	26	16	8	4
DR3	Use diagrams appropriately	25	18	7	6	28	17	7	2
DR4	Supplement natural language with other description of req.	19	26	9	2	18	19	10	7
DR5	Specify requirements quantitatively	12	23	18	3	20	15	13	6
DR6	Have direct contact with customers to avoid unclear requirements					37	13	1	3
SM1	Develop complementary system models	11	22	17	6	9	16	17	12
SM2	Model the system's environment	12	19	20	5	10	24	9	11
SM3	Model the system architecture	18	25	8	5	16	19	10	9
SM4	Use structured methods for system modelling	15	21	13	7	16	15	15	8
SM5	Use a data dictionary	15	18	17	6	10	20	13	11
SM6	Document links between stakeholder reqs and system models	13	18	19	6	18	11	14	11
RV1	Check that the requirements document meets your standards	28	16	7	5	19	20	8	7
RV2	Organise formal requirements inspections	15	18	17	6	15	18	14	7
RV3	Use multi-disciplinary teams to review requirements	16	16	17	7	23	11	10	10
RV4	Define validation checklists	17	17	15	7	17	16	13	8
RV5	Use prototyping to animate requirements	9	20	18	9	11	19	12	12
RV6	Write a draft user manual	20	17	13	6	13	12	19	10
RV7	Propose requirements test cases	20	21	10	5	20	14	15	5
RV8	Paraphrase system models	5	19	24	8	11	13	14	16
RM1	Uniquely identify each requirement	28	17	8	3	33	12	6	3

RM2	Define policies for requirements management	17	19	16	4	13	21	10	10
RM3	Define traceability policies	17	18	14	7	11	18	13	12
RM4	Maintain a traceability manual	15	14	19	8	9	17	14	14
RM5	Use a database to manage requirements	16	17	12	1	12	13	9	20
RM6	Define change management policies	17	21	10	8	20	9	14	11
RM7	Identify global system requirements	18	16	12	1	17	16	14	7
RM8	Identify volatile requirements	11	17	18	1	14	17	12	11
RM9	Record rejected requirements	10	11	23	12	16	13	13	12
RM10	Have a dedicated role for requirement engineering activities					20	7	13	14
CS1	Create safety requirement checklists	15	18	9	14	18	11	8	17
CS2	Involve external reviewers in the validation process	9	20	12	15	15	12	6	22
CS3	Identify and analyse hazards 16	16	15	14	11	21	9	9	15
CS4	Derive safety requirements from hazard analysis	14	16	13	13	18	10	10	16
CS5	Cross-check operational/ functional reqs against safety req.	16	11	16	13	22	5	10	17
CS6	Specify systems using a formal specification	12	20	13	11	17	10	11	16
CS7	Collect incident experience	12	17	14	13	14	12	15	13
CS8	Learn from incident experience	18	17	10	11	18	11	12	13
CS9	Establish an organizational safety culture.	10	23	12	11	17	11	11	15

Appendix H: Questionnaire

113

Full Name (optional) _____ Email Address _____

Job Title / Position _____ Company's country _____

128

Experience (in years)

☐ Less then 5 ☐ 5-10 ☐ 11-15 ☐ More then 15

Page description:

115

What is the primary business function of your company? (You may tick more than one)

☐ In-house development ☐ Outsource Development ☐ Other (Please specify) _____

118

Approximately how may staff are employed by your company? (please tick as appropriate)

☐ Less than 20 ☐ Greater than 200

☐ 20-199 ☐ Not Sure

117

What is the scope of your company? (please tick as appropriate)

☐ National ☐ Don't know

☐ Multinational ☐ Other (Please specify) _____

Tom De Lancey
Product Owner at Humana

Oh, and the Product Owner should be a subject matter expert. That is another way of saying that they can make decisions that the stakeholder would support

Sonia Naz
Research Scholar /Software Engineering

@Tom agree but if the P.O is a real customer he is in a better position to present the perspective of other customers and stakeholders isn't it the case?

Tom De Lancey
Product Owner at Humana

@Sonia - What you need in a product owner, is someone who knows about the business enough that they can make independent decisions without having to have meetings with multiple stakeholders every time.

In addition, the PO needs to know how the development process works so that they can have some feel for how to weigh options and costs of different decisions.

Often, the first attribute means that the PO works every day with or as the customer.

Daniel Beland
Software Developer at Thales Canada

What is also important is that the PO comes prepared to the sprint planning with clear and concise user stories, He/she must be available during the sprint to answer the team's questions without too much delay.

By experience, when the stakeholder(s) cannot aford enough time to do the work required of the PO, then with a proxy PO the team's performance will be much better.

Sonia Naz
Research Scholar /Software Engineering

So it means, in general we can say that proxy is not a bad idea if he is capable enough.

John Motz
Vice President of Product at ADP

I think it depends on where you are and what you are trying to do. In a case of s very mature product where your objective is to maintain and maximize I am in line with the above comments. If you are looking to disrupt a market,

enter a new market, drive innovation in a product I don't think you can just pull anyone who is an sme and would argue against it.

Moises Soto Jr
Project Manager - Senior Practitioner, Lean-Agile-SCRUM.

At bit late responding but it is indeed an issue of "what type of project" is the one you are trying to complete: A hybrid or truly an Agile/lean project. You see in a hybrid you may have approved requirements within UCs, if so then the PO mainly adds/suggests modifications due to changes that can occur in time (time does fly). If not then the proxy or the person representing the stakeholders (PO) must know more than other persons within the SCRUM team. The PO can negatively impact the project if he or she does not know enough about the solution/project.Hope that helps.

Steve Fastabend
Agile Coach at Centare

From the Scrum Guide ...
"The Product Owner is responsible for maximizing the value of the product and the work of the Development Team. How this is done may vary widely across organizations, Scrum Teams, and individuals." The guide goes on to explain the PO's responsibilities in regards to the backlog.

I agree, with some of the statements above, that it matters less where the PO comes from but it matters a great deal that the PO has the abilities to do the job.

One of the things that concerns me is the term "Proxy Product Owner". If this means that the real product owner is in the background second guessing decisions it could be counterproductive. The PO needs to make decisions about the product. Often these decisions are both detailed and timely. For this reason I would prefer a Product Owner over a Proxy Product Owner.

Lee Roche
Vice President at HCL Technologies

For me it really boils down to decision authority. If the proxy has no authority then you may just be wasting time and not getting the decisions made.
So the question really is, are decisions being made and do they stick?

Moises Soto Jr
Project Manager - Senior Practitioner, Lean-Agile-SCRUM.

Agree with the last two comments. The Product Owner is the key person when dealing with the requirements, they should be salient and the PO directs the development team to implement -- the Scrum Master can help but the proxy or substitute must know the product that is being build and the impact to the client. So if the PO is not capable of doing the task then the project will suffer and no incremental software will be created. This brings to mind also the

amount of collaboration that needs to happen for this methodology to succeed.
Thanks.

Sonia Naz
Research Scholar /Software Engineering
Top Contributor

@Moises "the amount of collaboration that needs to happen for this methodology to succeed." completely agree I am actually trying to focus on this aspect only regardless of the project's types and other various attributes.

@Steve " If this means that the real product owner is in the background second guessing decisions it could be counter prodcutive." this is also in line with above fact the product owner is actually the customer representative if he can't represent the actual customer it is off-course a disaster and to represent the actual customer he needs to collaborate with them in a fairly frequent way.

Leonor Urena, CSP, CSM, CSPO, PMP
Agile Coach & Trainer Scrum Mastering llc NY, NJ

Sonia, what really matters is how well aligned the proxy is to their Product Owner. The proxy will need to be empowered to make all the decisions representing the PO so there must be full agreement on the vision of that product.

Grant Sutton
Delivery Lead at Odecee

It's interesting that you ask whether having a proxy negatively impacts the team.
I'll make the assumption that the person who should be your PO has insufficient time to dedicate to the role and needs to delegate responsibility to someone else from the team.

With that assumption in mind it's a balancing act between the following factors
(1) how much domain/problem knowledge does the proxy have?
(2) Has the PO delegated authority and responsibility to the proxy in regards to decision making?
(3) Is the proxy happy to make timely independent decisions without validation from the PO?
(4) Does the proxy understand and value your agile development practices?
(5) Does the proxy themselves have sufficient time to do the task justice?

My experience is that as long as the proxy understands the problem and business domain, works with the team and is empowered to make decisions, then the value of having a proxy far outweighs the impact of delays to development when we are waiting for responses from our seldom available PO.

Nitin Khanna, Scrum Practitioner
Product Owner / Agile Mentor / Scrum Coach

Is the product backlog visible and open to all?

@Sonia - what if there was a positive outcome?

Sonia Naz

Research Scholar /Software Engineering
Top Contributor

@Nitin "Is the product backlog visible and open to all?" by all what do you mean? If it's P.O, Scrum master and development teams off course yes. Its good if there was a positive impact but how?

Nitin Khanna, Scrum Practitioner
Product Owner / Agile Mentor / Scrum Coach

@Sonia, the Product Backlog is intended to be visible to all, including stakeholders. This allows to to see what's ordered and is expected next. This is also a nice way to keep a PO in check, amongst other things....

Positive Impact to what I was describing --
A member of the Dev Team, having business analysis skills, was acting as a proxy PO. Such a person was aligned to the PO and had been empowered to move things ahead as much as possible. Just happened that his domain knowledge was also at par, if not more in some areas.
A key thing people miss is that the backlog has to be visible to as many as possible.
:)

Sean Gates
Scrum professional (CSM / CSP) now available

@Grant Sutton is asking exactly the right questions.
In addition to that it is important that the "real PO" either maintains regular and good interaction with his proxy and the work that is being done, or commits to accepting what was delivered in his absence. For this to work the PO does not have a right of VETO at a late stage in the project just because the solution is different to the one that he envisaged.

In summary answering yes to the question:
"has the PO delegated authority and responsibility to the proxy in regards to decision making?"
Means that the PO acknowledges that he is willing to accept those decisions.

- **In agile how frequently the customer representative (Scrum P.O) meets with other stakeholders like users etc.**

Sonia Naz **Research Scholar /Software Engineering Top Contributor**

Comments:

Hemant Vashisth, M.B.A., M.S.
Scrum Agile Expert | Software Change Specialist | Engineering Leader

Depending on the state of the project | release, the Scrum Product Owner should meet stakeholders once a week, or at least once every two weeks.

Seasonality will play into the equation. For example, if your company is heavily into e-commerce, and holiday shopping season is nearing, the PO might want to collaborate with key stakeholders multiple times (twice) a week to ensure that:

1. customer requirements are met, and the train on the right track.

2. Based on any new information, quickly course correct and prioritize backlog better for upcoming sprints.

Communication is crucial and any bad news must travel super-fast. This will increase the possibility of releasing a solid product | service to your customers.

Adrian Howard
Generalizing Specialist (mostly in Agile/UX)

As often as possible would be my advice. And I'd try and ensure that it's not just the PO who has contact with other stakeholders.

The team having contact time with the actual users really helps overall product quality in my experience (and others, see http://www.uie.com/articles/user_exposure_hours/ for example.)

Sonia Naz
Research Scholar /Software Engineering

@ Adrian Howard "it's not just the PO who has contact with other stakeholders" completely agree. Because through this we can share the burden of P.O as we mostly want the customer representative to devote his time being on-site working with the development team while he also need to work with the end users and business stakeholders. This dual role can be exhausting and creates sustainability issues for P.O.

Iain McKenna
Certified Scrum Trainer and Agile Coach at Project Success

As often as needed. If it's too much then that should become apparent as there will be very little to collaborate on and if it's not enough, that should also become apparent as there will be a lack of alignment between the PO and the stakeholders. Inspect and adapt - every context is unique so there is no "one size fits all" answer.

Adrian Lander, Enterprise Agile & Lean Coach

Sr. International Agile & Lean Transformation Coach / Agile & Lean Capability and Delivery Evangelist

As @Ian said : "very context is unique so there is no "one size fits all" answer"

Alignment needs to be ensured though, as the PO usually is not the (buying) customer. So there is a risk of misalignment which needs to be managed also by getting frequent feedback from real (buying) customers or end users that need the product to provide x value through usable functionality

What we successfully did on a few products in organizations in transformation from "traditional" is doing "key user feedback groups" after the sprint review. Once there was enough to involve key users in, so after 1 or 2 sprints, we would conduct a time boxed session with PO and key users and developers to maximize feedback on the current product increment, to be priorities by the PO. This also helps with the actual moving into effective use. You work with user champions. They can help prepare the users with what's coming as they are involved early and feel they can help shape the product.

This is an investment of max 2 hours per sprint. Does not need to involve all developers just 2 for gathering feedback and support users.

This is just a "try" technique in environments where this can pay off. Especially when there is some kind of UI (E.g. Web/Mobile with some backend)

I also have seen this as a great Scrum Team gelling experience where the Dev Team and SM support the PO in maximizing ROI and managing stakeholders.

Dusko Jovanovic, PhD, dipl.ing

Senior requirements engineer for Zoover, Meteovista, Vakantiereiswijzer, Founder of ReArch Integral Requirements Archs

Sonia,

With your question you are actually hitting one of the weakest points of the agile development: complex projects, with different disciplines and many different stakeholders involved. Agile works effectively on a narrow set of industrial projects, in other words: agile development has many restrictions.

You have discovered one of them: **"how frequently he meets with the other stakeholders?"** In the pure agile, there is only one customer that is involved in the daily/weekly agile practices. When there are many customer/user side stakeholders, pure agile approach cannot be applied. Think of developing a railway system. With such project, agile also suffers from other limitations, as for example: dynamics of developing software is much different than developing electronics, which in turn is much different than developing mechanics, let alone construction works. In the later, refactoring also almost always out of question too.

Therefore I would suggest to you to think and write about sorts of projects and product pure agile development is applicable too. Let me see your early version of this consideration, and I might add to your conditions a few I have been noticing along the way. Greetings

Asif Raza
Head of SW Development & Sales

Finland/Telecommunication

It depends on organizational size and internal process, even in medium size organization PO don't meet with end user due to organizational hierarchy, it has more to do with organization process then agile.

Abhijeet Nikte MBA, CSM
Collaborative Servant Leader || Lean/Agile Enthusiast and Practitioner || Change Agent

I do not think this has to do with Agile or non-Agile implementations. I do agree that Lean/Agile frameworks has tried to refocus more and more on customer collaboration and that is why we could be misled to some degree.

I am of the opinion that there should be a complete 360 degree evaluation - what does that mean? - just like a scrum team has its retrospectives, there should be retrospectives with the customers/users/stakeholders in terms of revisiting if what is being done and the way it is being done indeed provides value to the customer. If it is not, then there is a problem that needs immediate attention. Retrospectives are a very powerful tool. One key thing I am implying here is that the retrospectives should produce actionable items and be followed up on.

And the actual answer is - figure out for yourself. You will know when it is too little or it is too much. Be ready to listen :-)

Richard Green
Software Engineer - Retired

For any process to be under control,
the feedback has to have a periodicity matching the cycle time.

Anil Jaising, CSP
Executive Director at JPMorgan Chase

In Scrum the PO and team meet the user on the scrum review at the end of every sprint. The PO also can individually meet the user regularly to analyze and review future stories

Iain McKenna
Certified Scrum Trainer and Agile Coach at Project Success

@Anil -the team can also meet stakeholders during the sprint, after all, not? Sometimes stakeholders are better placed than the PO to answer the team's questions.

Anil Jaising, CSP
Executive Director at JPMorgan Chase

agree

Sonia Naz
Research Scholar /Software Engineering
Top Contributor

@Iain McKenna agree especially when the P.O IS from development organization (proxy).

Sonia Naz
Research Scholar /Software Engineering
Top Contributor

But the question is that is it realistic to involve all stakeholders or it's just a theory. Any suggestions? For example can we use computer mediated collaboration to connect with different stakeholders throughout the project?

Iain McKenna
Certified Scrum Trainer and Agile Coach at Project Success

@Sonia, the bigger question is how do you maximize the value created by a development team without involving all of the stakeholders? Remember that Scrum doesn't tell you how to do your work or what work you should do, instead it provides a framework in which to inspect what work you have done and will do and how that work is done so that you can learn and improve over time.

If stakeholders aren't involved, that would suggest to me an impediment (and perhaps organizational dysfunction) which can be improved and through improvement, you should be able to increase the value delivered.

Use whatever mechanisms you can to engage with the stakeholders, inspect how those mechanisms are working and if you see opportunities to improve by changing the mechanisms, then change them.

Abhijeet Nikte MBA, CSM

Collaborative Servant Leader || Lean/Agile Enthusiast and Practitioner || Change Agent

Product Owner - proxy or real should be CRACK!

My apologies that I do not remember where I first read this acronym, but a google search yielded this link - http://www.agilechanger.com/2013/08/crack-product-owner/

Collaborative: Work well with development team and stakeholders.
Representative: Have a good understand and vision of the product and being able to represent the stakeholder.
Authorized: Empowered to make decision.
Committed: Share the development team and stakeholder goal.
Knowledgeable: Understand and experienced in the specific domain to guide the project to success.
If you have a PO with these characteristics then you should be fine.