2009

CHARACTERISTICS AND IMPACT ON REWORK OF INFORMATION LOST DUE TO LACK OF DOCUMENTATION DURING REQUIREMENTS ENGINEERING AND SOFTWARE ARCHITECTING

Muhammad Iftekher Chowdhury

CHARACTERISTICS AND IMPACT ON REWORK OF INFORMATION LOST DUE TO LACK OF DOCUMENTATION DURING REQUIREMENTS ENGINEERING AND SOFTWARE ARCHITECTING

(Spine title: Information Lost during Software Engineering)

(Thesis format: Monograph)

by

Muhammad Iftekher Chowdhury

Graduate Program in Computer Science

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science

The School of Graduate and Postdoctoral Studies
The University of Western Ontario
London, Ontario, Canada

THE UNIVERSITY OF WESTERN ONTARIO
SCHOOL OF GRADUATE AND POSTDOCTORAL STUDIES

CERTIFICATE OF EXAMINATION

Supervisor	Examiners
Dr. Nazim Madhavji	Dr. Michael Bauer
	Dr. Sylvia Osborn
	Dr. Isola Ajiferuke

The thesis by

Muhammad Iftekher Chowdhury

entitled:

Characteristics and Impact on Rework of Information Lost due to Lack of Documentation during Requirements Engineering and Software Architecting

is accepted in partial fulfillment of the
requirements for the degree of
Master of Arts

Date_____

_____
Chair of the Thesis Examination Board

Abstract

Requirements engineering (RE) and software architecting (SA) are two front end activities in the software development lifecycle which have great impact on the overall success of a software project. Because of the highly communication intensive nature of these activities, different types of communication channel from face to face discussions to email, chat, video conferencing etc., are used for these activities. All types of information discussed during these activities are not possible to document. Information lost during this time might cause problems in later stages. We conducted an industrial survey on 32 software professionals from a total of 23 different companies with 1 to 15 years of industrial experience to investigate the characteristics and impact in terms of introduced rework of information lost during RE and SA due to lack of documentation. Our result shows that the types of information that are lost most frequently during RE and SA due to lack of documentation are: "Issues", "Rationale, priority, source and assumptions behind requirements" and "Tactics". Information lost during RE introduces rework in "RE" and "SA" most frequently whereas information lost during SA introduces rework in "Design and coding" and "System integration" most frequently. Also the mediums that have the propensity of losing information if used for communication during RE and SA were identified. This knowledge could help software practitioners to decide which medium to avoid or to use it with caution in the RE and SA processes and could motivate researchers to venture into other areas of software engineering (such as design, coding, testing, maintenance, etc.) from the point of view of information lost due to lack of documentation. To the best of my knowledge, this is the first work which focuses on characteristics and impact of information lost during RE and SA.

Keywords: software engineering, requirements engineering, software architecting, software documentation, information loss, empirical study.

Acknowledgements

First and foremost, I would like to thank Dr. Nazim Madhavji for his constant guidance and supervision. Without his strong motivation, valuable inspiration and support, this work could not have been completed. Your passion for education and research are admirable.

I would like to thank all the members in my research team. Special thanks goes to Remo Ferrari who has suggested many valuable ideas for conducting the research.

I would also like to thank Dr. Shahidul Islam for giving suggestions on statistical analysis with his diverse knowledge in Statistics.

I would like to thank the participants of the study; I greatly appreciate the effort that was expended in the research, and hope that the research output will be of good value to everyone.

Last, but not the least, I would like to thank my wife, Sharmin Ahmed: my fellow research member, and continuous source of support and ideas. I am grateful for her love, patience and trust that acted as a driving force for the completion of the work.

Table of Contents

List of Tables

List of Figures

List of Appendices

Chapter 1. Introduction

Requirements engineering (RE)[1] and software architecting (SA)[2] are two front end activities in the software development lifecycle which have great impact on the overall success of a software project. They involve communication between varieties of stakeholders using different types of communication channels ranging from face to face discussions to email, chat, video conferencing etc., to produce the software artefacts. Textbooks (e.g., [4], [25], etc.) and standards (i.e., [21], [22]) suggest recording all the artefacts and necessary related information as part of the documentation which seems impossible in reality. Because there is usually little or no traceability between the software artefacts and their context (i.e. face to face discussions, e-mails, chat sessions, etc.), it often becomes quite difficult or even impossible to retrieve lost information. This can cause problems in the software development process in the form of errors, misunderstandings and parallel work [35] and can have an effect on the actual product in the form of architectural drifts and incorrect design [15] leading to software defects [14][7]. In fact, information lost during RE and SA due to the lack of documentation is one of the main reasons behind defects in software development and maintenance [52]. To mitigate this problem we need to understand the characteristics and impact of information lost during RE and SA. Even though several aspects of a software project (such as time[41], cost[41], quality[7], etc.) can be affected by this lost information, for time and resource constraints, this thesis focuses on impact in terms of frequency of introducing rework on different activities (e.g., RE, SA, coding, system integration, etc.)

1.1 Significance of research

Determining the types of information lost during RE and SA due to lack of documentation is significant because this knowledge adds to the body of knowledge on

[1] For the rest of the thesis, the acronym RE refers to requirements engineering.

[2] For the rest of the thesis, the acronym SA refers to software architecting.

documentation during RE and SA. Such knowledge can also be used in improving software processes and technologies. For example, knowing the medium through which information is most frequently lost could help software practitioners to decide which medium to avoid or to use it with caution in RE and SA processes. There are already some research tools that are available (e.g., [14] and EGRET [38]) and tools used in industry (e.g., Rational Team Concert [20]), which provide traceability between software artefacts and communication artefacts (e.g., meeting videos, email, chat, etc.). To the best of my knowledge, there is no empirical evidence that shows the impact on rework due to the information that is lost during RE and SA. The results of this work could motivate researchers to venture into other areas of software engineering (such as design, coding, testing, maintenance, etc.) from the point of view of information lost due to lack of documentation and could motivate the practitioners to use the tools mentioned above.

1.2 Originality of research

To the best of my knowledge, characterisation of the information lost during RE and SA due to lack of documentation has not been carried out by any other researcher. Even though there is mention (e.g., [14], [35], [38], etc.) of information loss using different communication mediums, there is no scientific study I am aware of that explores the degree of information loss using different communication mediums. While there is some evidence in the literature of the impact of lost information on product quality (such as architectural drift [15] and software defect [7]) it is not known how frequently this lost information introduces rework.

1.3 Thesis organization

Chapter 2 discusses the relevant background literature and research gap. Chapter 3 explains the research questions, metrics, experiment design and threats to validity. Chapter 4 presents the data analysis, results and interpretations. Chapter 5 explores the

implications of the findings and Chapter 6 closes this thesis with limitations of this study, future work, and conclusions of this research.

Chapter 2. Background

The review of the literature culminated into three broad areas discussed in sections 2.1, 2.2 and 2.3. Section 2.1 describes taxonomy of software documentation and what is meant by requirements document and architecture document in this thesis. Section 2.2 discusses why the current practices of software documentation are not sufficient enough to prevent information loss and how severe the problem is. Section 2.3 discusses key empirical research in the area of software documentation. Section 2.4 describes the overall research gap based on sections 2.2 and 2.3.

2.1 Taxonomy of software documentation

In [42], Sommerville discussed different types of documentation which are usually produced during the software development process. He also discussed document quality, document standards, process of documentation and documentation management. The author has derived[3] the taxonomy of software documentation shown in **Figure 1** from the textual description of different types of documentation given in [42]. Some key types of documentation from the taxonomy are discussed below to show where requirements documents and architecture documents stand in the context of overall software documentation.

Process documentation [42]: Process documentation provides a description of development and maintenance processes. Process documents such as plans, schedules, process quality documents, organizational, project standards, etc. are developed to manage software processes.

[3] UML 2.0 guidelines for drawing class diagram from [6] were followed to create the taxonomy.

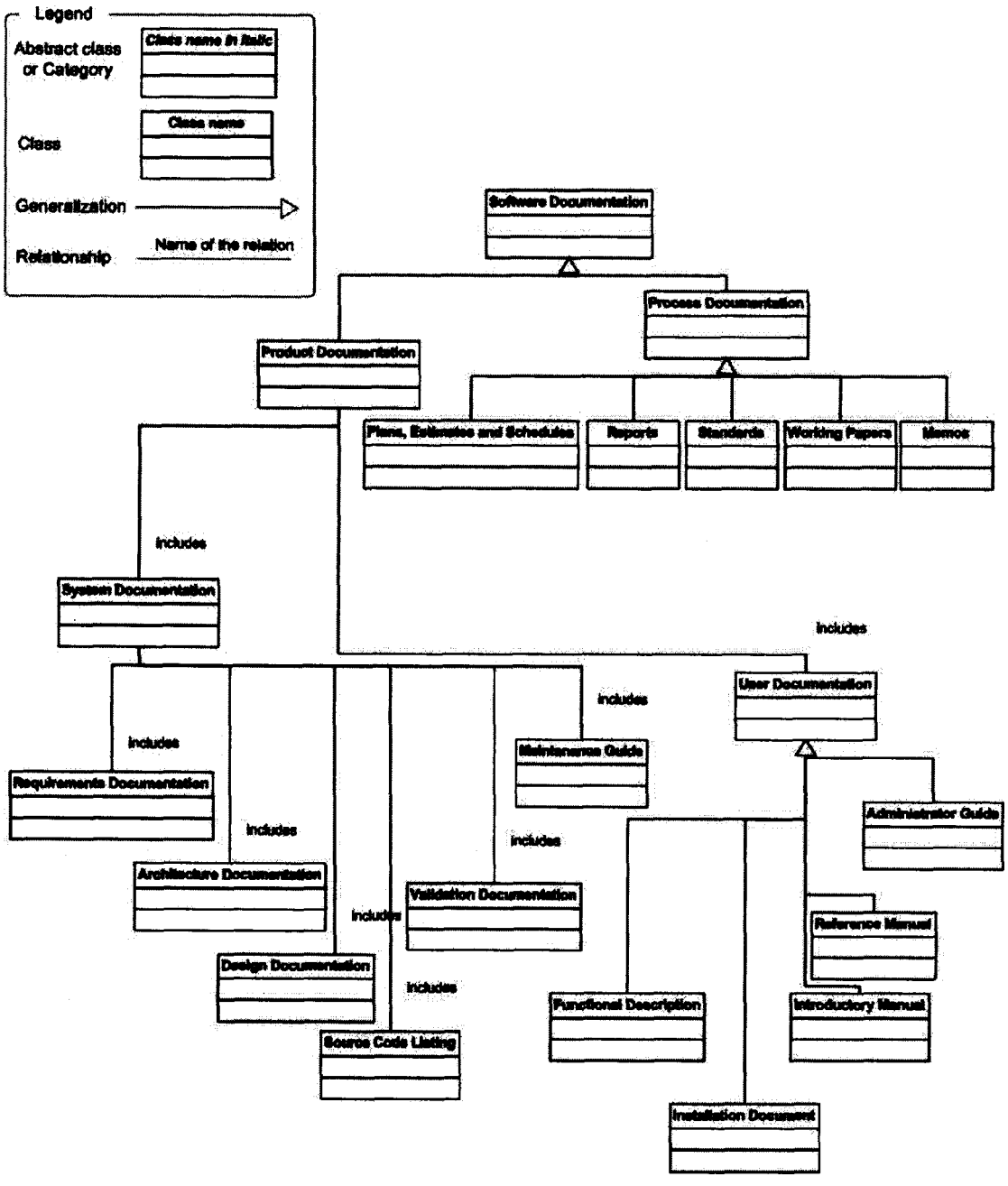

Figure 1: Taxonomy of software documentation

Product documentation [42]: This documentation provides descriptions of the software product. It can be further divided into two categories, system documentation and user documentation.

User documentation [42]: User documentation tells users how to use the software product. The target reader of different user documents can vary from end users to system administrators. User documentation is a category of documents (e.g., introductory manual, installation document, etc.) rather than being an actual document.

System documentation [42]: System documentation provides description of the software product from the viewpoints of the software professionals responsible for the development and maintenance of the system. System documentation includes a number of documents such as requirements documents, architecture documents, design documents, maintenance guides, etc.

Requirements documentation [25]: A requirement is a statement of what a system is required to do and the constraints under which it is required to operate. The requirements document is the official collection of all the requirements of a system for different stakeholders (e.g., customers, end users, developers etc.). It is also known as 'functional specification', 'requirements definition', 'software requirements specification (SRS)' etc.

Requirements can be written in different formats like natural language descriptions [25], formal specifications [25], user stories [2] etc. Also, different tools varying from simple word files [13] to complex multitier data base management systems [31], are used for storing requirements documents. In this thesis we refer to any form of requirements and any representation of requirements documents using the terminology 'requirements' and 'requirements documents' respectively.

Architecture documentation [4]: The software architecture of a system is a structure of the system consisting of software elements, their externally visible properties and the relationship between different elements. The architecture document of a software system is the unambiguous, sufficiently detailed and properly organized representation of the

software architecture for different stakeholders from the developers end (e.g., architects, requirements engineers, testers, integrators, managers etc.).

Software architecture can be represented using different notations varying from general unified modeling language (UML [50]) [18] to specific architecture description languages [9]. A variety of tools, from drawing tools [1] to professional industry tools (e.g., Rational Software Architect [19], Borland Together [40], Argo UML [49] etc.) are used for recording software architecture. In this thesis we refer to any form of software architecture and any representation of software architecture document using the terminology 'architecture' and 'architecture document' respectively.

2.2 Why documentation in practice is inefficient such that information gets lost

Different types of communication channels (e.g., face to face, e-mail, chat, etc.) are used for RE and SA because of their highly communication intensive nature. All types of information discussed during these activities are not possible to document. Here are some of the reasons why information gets lost during RE and SA:

- In current documentation practice one of the stakeholders present in the meeting takes the role of the scribe [14]. Because different stakeholders involved in the process may not share a common language or project knowledge, the notes of the scribe can be incomplete, inconsistent or incorrect [14]. For example, the scribe may misinterpret a statement, note something incorrectly or partially, or omit important statements made by a stakeholder [14]. This might cause loss of information from the overall documentation.
- Often important information is communicated outside formal documentation [35][8]. Fluid information (such as meetings and oral communications, blogs, chats, informal wikis, and phone calls, etc.) usually gets ignored from explicit documentation [35].

- Documentation is effort consuming [35] but quickly becomes outdated [8] [11]. Often it takes a few weeks to a few months for the documentation to be updated [27]. This delay in update might result in information loss.
- Because of the lack of integration between software artefacts and communication environment (e.g., e-mail client, instant messenger, video conferencing tools, etc.), information gets fragmented across several media (e.g., documentation tools or databases, bug repository, e-mail, text chats, etc.) which leads to frequent context switching (e.g., rechecking old email, chat sessions, meeting minutes) during work and results in a lack of common understanding and awareness [38]. Important domain information is personalized (i.e., "at best retained in the team member's mind") and gets lost when the team member leaves the project [38].

Here is some empirical evidence from the existing literature which shows the severity of the problems caused by information lost during RE and SA due to lack of documentation:

- In a Siemens project the root cause analysis done by Siemens Corporate Research(SCR) showed that 40% of the defects were caused by incorrect, incomplete or not at all recorded requirements[14][7].
- In [41], a survey was conducted in 63 software companies in Malaysia. The companies cited problems like incomplete requirements (79.4%), ambiguous requirements (76.2%), and misplaced requirements in a requirements document (37.1%) as some of the reasons behind late delivery of products (76.2%), budget over-runs (58.7%) and poor quality products (44.4%).
- From three surveys of 39, 41 and 44 software maintenance professionals (with overlapping participants) in June 1991 to September 1991, 19 major problems in software maintenance were identified [12]. Incomplete or non-existent system documentation was ranked number 3 amongst them.

2.3 Empirical research on software documentation

There is empirical research on software documentation as far back as 1982 (e.g., [36]) and as recent as 2009 (e.g., [51]). The focus of different empirical studies in the area of software documentation varies from the overall software documentation process (e.g., [52]) to documentation of a particular software artefact (e.g., design rationale [47]). Here we will discuss some of the related work. It is important to note that the actual research focus for some of the studies mentioned below spans a broader area than just software documentation. But because of our scope, we only discuss issues and findings that are related to software documentation.

Visconti and Cook developed a process maturity model (DPMM) and assessment procedure to assess system documentation process [52]. They used a questionnaire to collect data from 91 projects at 41 different companies to assess against their model. According to their assessment the general software documentation practice in industry was not satisfactory. They found the assessed organizations mainly following the key documentation practices necessary to ensure the existence of policies or standards but failed to ensure the monitoring of compliance of those policies or standards. Their result also showed dissatisfaction with key practices necessary to assure the quality and usability of actual documentation.

Tang et al. reported a survey on 81 practitioners to determine their perception about the importance of design rationale and how they use and document design rationale [47]. The study shows that the use of design rationale is quite frequent among the participants and that they are aware of the importance of documenting design rationale but face problems like absence of methodology and tools support.

Lethbridge et al. reported three studies on the use of software documentation [27]. Their study method included interviews, surveys and observing individuals' work. Their study showed that documentation other than testing and quality documentation (such as test cases and plans) are rarely updated. Even if the changes are made to the documentation it usually takes several weeks for the documents to reflect actual system changes. They also found that the out-dated documentation might remain useful in some cases, particularly if the high level abstractions remain valid. They also mentioned the necessity of simple and powerful documentation tools and formats.

Beecham et al. conducted a study in 12 software companies ranging from CMM level 1 to 4 to find their software process improvement problems [5]. They divided 200 employees from these companies into 45 groups and used focus group techniques to collect data. Their result shows that developers, project managers and senior managers report similar types of process improvement problems and the documentation issue (i.e., coordination and document management, feedback, post-mortems and data collection) is one of them. Amongst the project issues, documentation was ranked number 2 and amongst the top 6 problems identified from all the areas (i.e., organizational issues, project issues and software development lifecycle process), documentation was ranked number 3.

Smolander and Päivärinta studied three software development organizations to find out how the rationale behind documenting software architecture varies between different stakeholders [39]. They used semi structured interviews for data collection and examined the organizations' software process specifications and actual architectural documents. Their result shows that only designers emphasized architecture as a foundation for later development whereas the other stakeholders' rationale behind documentation was mainly for communication, interpretation and taking decisions.

Singer et al. reported four studies on daily activities of software engineers [37]. One of the studies showed that the numbers of people involved in reading documentation was greater than the number of people involved in writing documentation. One of the other studies showed that more than 70% of the people are somehow related to documentation.

Sommerville and Ransom reported a study in companies to evaluate a RE process maturity model and whether RE process maturity leads to business improvements [43]. One of the eight RE process areas for "good" RE practices was documentation. During initial assessment only one company was in maturity level 4 in the documentation area. But after process improvement, six out of the remaining 8 companies were up-graded to level 4 in the area of documentation.

To the best of our knowledge there is no study which specifically focuses on the overall characteristics of the lost information during RE and SA due to lack of documentation and to what degree the lost information introduces rework in different software engineering activities.

2.4 Analysis and research gap

It was seen in section 2.2 that some of the major reasons behind the lack of documentation and information loss during RE and SA were: lack of common language or project knowledge between the different stakeholders [14], ignoring fluid information (such as meetings and oral communications, blogs, chats, informal wikis, and phone calls, etc.)[35] , and absence of integration between software artefacts and communication artefacts [38]. Tools like [14] and EGRET[38] were developed to mitigate the problem by providing traceability to contextual artefacts like videos of face to face discussion, emails and chat sessions. Section 2.3 shows that documentation from the points of view of process maturity (e.g., [52] and [43]) and process improvement problems (e.g., [5]) are

well studied. Use of software documentation in practice (e.g., [27]), how documentation comes in software engineers' daily activity (e.g., [37]) and diversity of different stakeholders' rationale behind documentation (e.g., [39]) also got focus. The work of Tang et al. [47] shows how frequently design rationales are discussed and documented. But the overall characteristics of the information lost during RE and SA due to lack of documentation, how information is lost while using different communication mediums and to what degree the lost information introduces rework in different software engineering activities remain unexplored, which motivates the author to empirically investigate this area.

Chapter 3. The Empirical Study

This chapter describes the details of the study and the research procedures that were carried out. Section 3.1 defines the overall research goal, specific research questions and associated metrics. Section 3.2 describes the research procedures in terms of instrument design, data collection and data analysis. Sections 3.3 and 3.4 describe the participants and threats to validity respectively.

3.1 Goal, Questions and Metrics

We used GQM [3] to formulate the overall research goal, research questions necessary to achieve that goal and associated metrics to gather appropriate data.

The Goal:

Purpose: To determine and analyze

Issue: The characteristics and impact on rework of

Object: Lost information due to lack of Documentation

Viewpoint: From the viewpoint of internal stakeholders (i.e., developers and management)

Context: In the context of software development projects with focus on requirements engineering (RE) and software architecting (SA).

Questions and Associated Metrics:

The goal stated above has two dimensions, *characteristics* and *impact on rework* of lost information. In order to formulate appropriate research questions we mapped different question formats mentioned by Yin [55] and possible substances of interest from these two dimensions to the object of measurement in Figure 2 and Figure 3. As we can see in

Figure 2, the possible substances of interest for the question formats "How?", "Why?" and "Who?" are embedded in the object and viewpoint of the goal. So these substances became the context of the *characteristics* questions. Question 1 and Question 3 are formulated from the remaining substances of interest to address the *characteristics* of the lost information.

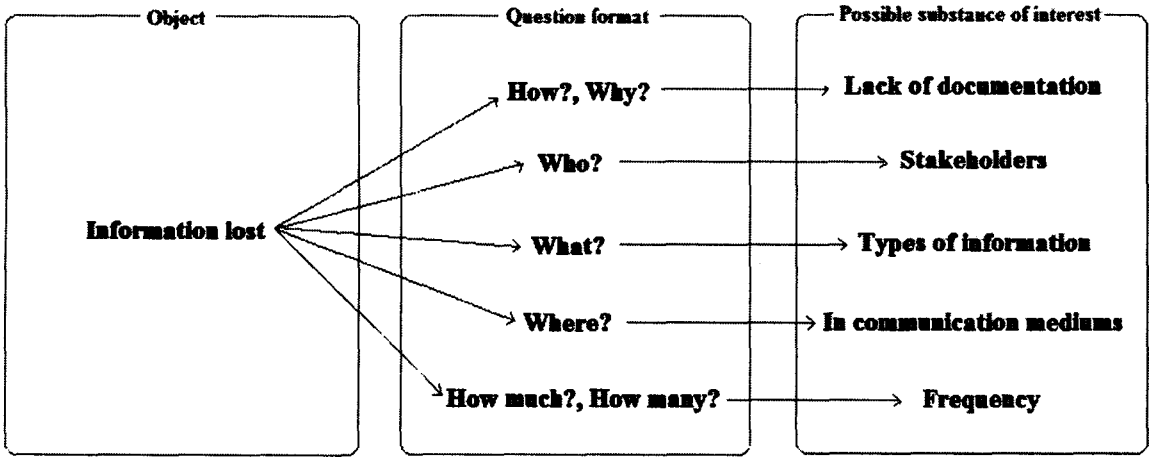

Figure 2: Possible substances of interest for the *characteristics* questions

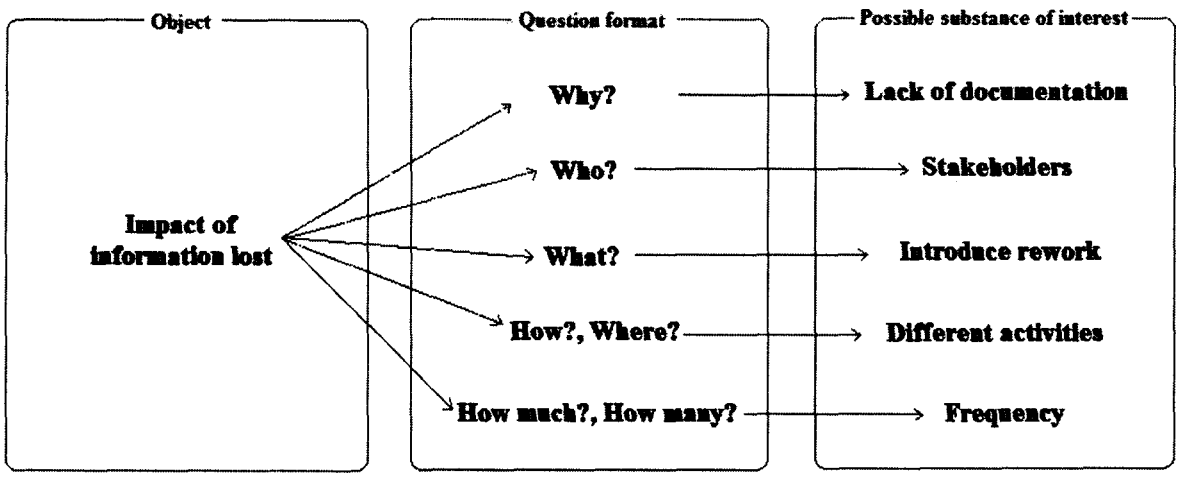

Figure 3: Possible substances of interest for the *impact on rework* questions

Similarly from Figure 3 we can see that the possible substances of interest for the question formats "How?", "Why?", "Who?" and "What?" are embedded in the object and viewpoint of the goal. So these substances became the context of the *impact on rework* question. Question 2 is formulated from the remaining substances of interest to address the *impact on rework* of the lost information.

Question 1: What types of information are lost, and to what degree, due to a lack of documentation during RE and SA?

Metrics Associated with Question 1:

M1: Frequency[4] of *discussion* of different types of information in RE and SA meetings.

Table 1: Metric M1-Frequency of discussion of different types of information in RE and SA meetings

Types of Information	**Frequency of discussion**					
	Never	Rarely	Sometimes	Most of the time	Always	Not sure
Requirements						
Quality attributes						
...						
Other artefacts						

M2: Frequency of *documentation* of different types of information from RE and SA meetings.

[4] The term "Frequency", as used in this thesis, applies to the ordinal scale mentioned in Table 1("Never" to "Always") and not only to the traditional quantitative implication of the term.

Table 2: Metric M2-Frequency of documentation of different types of information from RE and SA meetings

Types of Information	**Frequency of documentation**					
	Never	Rarely	Sometimes	Most of the time	Always	Not sure
Requirements						
Quality attributes						
...						
Other artefacts						

M3 : Difference between M1 and M2.

Question 2: What is the impact[5] in terms of rework of information lost on the different software engineering activities (e.g., design, coding etc.) due to the lack of documentation during RE and SA?

Metrics Associated with Question2:

M4 : Frequency of *rework* introduced (to different activities) due to information loss during RE.

Table 3: Metric M4-Frequency of rework introduced (to different activities) due to information loss during RE

Activity	**Frequency of rework introduced as requirement document and related knowledge are not adequate**					
	Never	Rarely	Sometimes	Most of the time	Always	Not sure
Requirements						

[5] As mentioned in the Introduction chapter, the focus of this thesis is limited to the impact in terms of frequency of introduced rework. The author realizes that there are other measures of the variable rework (such as cost of rework, project delays, customer satisfaction etc.) but these are not in the scope of this thesis and subject to future work.

engineering						
Design and coding						
...						
System integration						

M5: Frequency of *rework* introduced (to different activities) due to information loss during SA.

Table 4: Metric M5-Frequency of rework introduced (to different activities) due to information loss during SA

Activity	**Frequency of rework introduced as architecture document and related knowledge are not adequate**					
	Never	Rarely	Sometimes	Most of the time	Always	Not sure
Requirements engineering						
Design and coding						
...						
System integration						

Question3: What is the frequency of information lost while using different *communication mediums* during RE & SA?

Metrics Associated with Question3:

M6: Frequency of use of different *communication mediums* for RE discussion.

Table 5: Metric M6-Frequency of use of different communication mediums for RE discussion

Communication mediums	Frequency of use for requirement discussion					
	Never	Rarely	Sometimes	Most of the time	Always	Not sure
Face to face discussion						
Email						
...						
Wiki, blog, forum etc.						

M7: Frequency of use of different *communication mediums* for SA discussion.

Table 6: Metric M7-Frequency of use of different communication mediums for SA discussion

Communication mediums	Frequency of use for architecture discussion					
	Never	Rarely	Sometimes	Most of the time	Always	Not sure
Face to face discussion						
Email						
...						
Wiki, blog, forum etc.						

M8: Correlation (if any) between M4 & M6.

M9: Correlation (if any) between M5 & M7.

Figure 4 shows the relationships between the goal, questions and metrics.

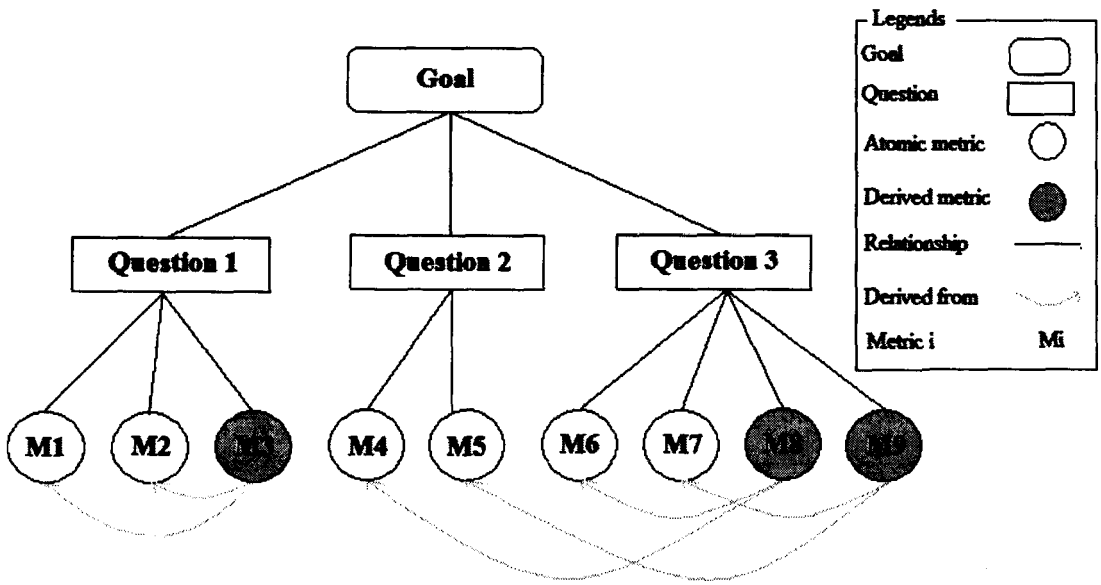

Figure 4: Relationships between the goal, questions and metrics

3.2 Research procedures

In order to investigate the stated research questions we conducted a knowledge seeking empirical investigation involving a number of software professionals. According to Yin [55] (also shown in Table 7) a survey should be considered as a research method when the form of research questions are "who, what, where, how many, how much?". As all our research questions are in form of "what, where, how much" we decided to select survey as our research method.

Table 7: Relevant Situations for Different Research Strategies [55]

Strategy	Form of Research Question	Requires Control of Behavioural Events	Focuses on Contemporary Events
Experiment	How, why?	Yes	Yes

Survey	Who, what, where, how many, how much?	No	Yes
Archival analysis	Who, what, where, how many, how much?	No	Yes / No
History	How, why?	No	No
Case study	How, why?	No	Yes

The "inclusion criteria"[24] for the survey were software professionals with minimum one year of industrial experience. Everyone else was the "exclusion criteria" [24]. Availability sampling and snowball sampling were used to choose the survey participants. Availability sampling refers to the seeking of responses that fall under the inclusion criteria and are available and willing to participate in the survey [17]. Snowball sampling is to rely on reference from initial respondents to generate additional respondents [17].

Data for this study has been collected from the subjects involved using a web based questionnaire in the summer of 2009. Details of the instrument design and data collection will be discussed in section 3.2.1 and 3.2.2. As the study was conducted without a significant underlying conjecture (on characteristics and impact of information lost during RE and SA) on which to build a hypothesis, the study can be described as an exploratory study [34]. So, the only types of hypotheses that are discussed in this thesis are the "null hypothesis' which are necessary for null hypothesis statistical testing. Details of the statistical analysis methods will be discussed in section 3.2.3.

The industrial survey was conducted to collect data for two studies: the first study is the one described in this thesis and the second is a complementary investigation[6]. The survey

[6] This complementary investigation is on the concept of "feed-forward" in software engineering.

questionnaire was divided into three sections. Section 1 was the background section and development of this part can be attributed to the authors of both the studies. Section 2 and section 3 focused on individual topics and the design for these sections can be attributed to the respective researchers. Also, both the studies involved analysis of the participants demographic data found from the background section.

3.2.1 Instrument design

The survey instrument was designed accordance to the guidelines mentioned in [23] . Some of the major points of our instrument design and support tool are described below.

Questionnaire design:

- Different questions were formulated to cover all the metrics. Table 8 shows that all the atomic metrics were covered by the survey questions (please see Appendix A for actual questions).

Table 8: Relation between survey questions and actual research metrics

Survey questions	Metrics					
	Metric M1	Metric M2	Metric M4	Metric M5	Metric M6	Metric M7
Question 7	X	X				
Question 8					X	X
Question 9			X	X		

- The purpose of the survey was explained at the beginning of the questionnaire to give the participants an introduction to the survey. To avoid the researcher bias the purpose was described in general terminology instead of indicating the actual research goal.

- To motivate the participants to give their honest opinion it was mentioned in the questionnaire that the gathered information will be used for scientific research purpose only and no individual or organization will be identified in the aggregate results.

- Also, it was mentioned that the aggregate results from the survey will be freely provided to them.

- Necessary terminologies were defined at the beginning of the survey to avoid confusion.

- Who should fill out the survey in terms of qualification was mentioned before starting the actual questions.

- All the instructions and questions were kept on a single page instead of splitting over multiple pages so that the participants did not need to submit the questionnaire multiple times. In this way it was also possible that the participants could see all the questions at a time to give consistent answers and did not need to change page to change any answer.

- Estimated time was mentioned at the beginning of the questionnaire so that the participants had a metal preparation for answering all the questions.

- Necessary contact information was provided in the beginning of the questionnaire to make the survey reliable to the participants and providing the facility to give any off-line feedback.

Support tool design:

A web based tool[7] was developed to conduct the survey. The tool allowed the participants to log in to the actual survey with their email id. Once logged in the users were forwarded to the actual survey. There were options for saving the survey response so that a participant could complete the survey in multiple sessions. After completing the survey the participants were forwarded to the thanks giving page which also provided contact for support. Some of the major points of the tool design:

- Font size 12 to 14 was used all over the questionnaire to make sure of visibility of the content.
- Frameset was used to separate each questions.
- Horizontal line was used to separate different sections.
- Simple HTML elements like checkbox, radio button and tables were used to format the questions.
- Bolding, underlining and capitals were used to show emphasis.

3.2.2 Data collection

The survey instrument was hosted in a publicly available location ([45]) in the Internet and the URL was emailed to participants. We invited 29 professionals to participate in the survey. We also requested them to forward the invitation to others who were eligible for participation and willing to participate and provide us the contact for the requested individual. They invited 12 more professionals. Out of these 41 professionals 32 (approximately 78%) completed the survey. The responses from the participants were

[7] The tool was a combination of web forms, business logic developed in PHP [30] and a MySQL [28] database. The major points for the forms design are mentioned here. The actual architecture of tool is not relevant for this thesis and so is not included here.

stored in the remote database associated with the instrument. Data from the remote database was exported to an Excel spreadsheet for further use (please see Appendix B for the condensed survey data). For statistical analysis necessary information was saved in comma separated (.csv) format and imported in statistical analysis tool R [48].

3.2.3 Data analysis

When data are measured on an ordinal scale, only their ranks are meaningful rather than the actual data values [46]. For analyzing such data, nonparametric or distribution-free statistical methods are used that make very few assumptions about the distribution of the population [46]. As all our atomic metrics are measured in ordinal scale, we used nonparametric statistical methods such as one and two samples Wilcoxon signed rank test, and Spearman rank order correlation test. The statistical language and environment R[48] were used for the ease of statistical analysis.

The one sample Wilcoxon signed rank test was used to measure the medians for all the atomic metrics and associated confidence intervals. As the two sample Wilcoxon signed rank test can be used to compare the medians of difference of two sets of paired ordinal data [46][10], we used it to calculate the metric M3 which is the difference between M1 and M2. Also, 95% confidence intervals (CI) for the medians were calculated. If the 95% confidence intervals do not include zero (i.e., either the upper and lower bound of CI are positive or both of them are negative) then the medians are considered statistically reliable [44].

The Spearman rank correlation test was used to calculate metrics M8 and M9. The Spearman rank correlation test calculates the spearman rank correlation coefficient (ρ) between two set of paired ordinal data. If the ρ value is in between 0.4 and 0.7 then the correlation is considered to be moderate whereas a ρ value more than 0.7 is considered as

high correlation [33]. In the case of the two sample Wilcoxon signed rank test and Spearman rank correlation test the associated p-values were calculated. P-values <0.05 were considered as statistically significant evidence for rejecting the null hypotheses in accordance with [44].

3.3 Participants[8]

Determining the participants for the study was an important step in the design of the empirical study. By looking at the profile of the participants that took part in the survey, it will be possible to determine the context of the study and which area of software engineering it is applicable to, and where the new knowledge can be used and what other complementary studies can be conducted.

The participants of the survey ranged from programmer to consultants and chief technical officers with a varying number of years of experience and different geographical distribution. In total there were 32 participants from a total of 23 different companies with 1 to 15 years of industrial experience. The background of the participants will be described in more details in the following three subsections. In the participants' background subsection, the role or title of the participants, their area of expertise and number of years of experience will be described. In participants' organization section, the team and project size and the type of process models followed in the organization will be discussed. In the final subsection, the geographical distribution of the participants will be discussed.

[8] As mentioned in section 3.2, this participants section is used as it is in a complementary investigation.

3.3.1 Participants' background

The role of majority of the participants for the survey was that of programmer (38%) or senior software engineer and analyst (38%) while only 3% of the participants were software maintenance engineer or a consultant. The main focus of this study was in the area of RE and SA. Upon verbal communication with the participants, it was determined that in the organizations the participants worked in, there was no explicit role or title of software architect or requirement engineer and the senior software engineer and system analysts were responsible for architecting and requirements engineering. A reason behind this was that a good number of the participants followed agile methods and so even though an individual in an organization had one role they could have several different responsibilities. The distribution of the role of the participants in the organizations is shown in Table 9. The difference between the role of programmer and senior software engineer and analyst is that a programmer is mostly responsible for coding and low level design whereas a senior software engineer and Analyst are responsible for upfront activities such as RE, SA, high level design and planning. It should also be noted that other stakeholders were also involved in RE and SA activities.

Table 9: Percentage distribution of role or title of the participants

Participants role or tile	Percentage
Programmer	38%
Senior software engineer and analyst	38%
Quality assurance engineer	6%
Testers	6%
Software maintenance engineer	3%
Management	6%
Consultant	3%

The background experience of the participants is shown in Table 10. A high number of the participants have experience of design and coding. This may be due to the fact that in many of the organizations, the participants joined in the entry level job of programmer and/ or tester and are either promoted or switch to other areas such as RE and SA. However, a large number of the participants have background experience of both RE and SA as well.

Table 10: Frequency distribution of background experience of the participants

Area of background experience	Number of participants
Requirement engineering	17
Software architecting	17
Design and Coding	28
Testing	20
Software maintenance	14
Project management	9
Quality control and assurance	9
Process improvement	10

The participants have a range of experience from 1 to 15 years. One of the minimum criteria for the selection of the participants for this survey was at least one year of industrial experience in software engineering. The number of years of experience of the participants was broken down into four clusters as shown in Table 11.

Table 11: Percentage distribution of years of experience of the participants

Number of years of experience	Percentage

1 year	13%
2 years	25%
3 to 4 years	31%
More than 5 years	31%

3.3.2 Participants' organization

The software development lifecycle models followed by the participants ranged from traditional models like waterfall and iterative to a combination of agile methods like XP and scrum. Table 12 represents the different lifecycle models followed by the participants. The "Others" models followed include lifecycle models such as rational unified process, model driven development and any other customized lifecycle model followed in the organization.

Table 12: Frequency distribution of the software development lifecycle models followed by participants

Software development lifecycle models followed	Number of respondents	Cumulative number of respondents
Waterfall	16	
Iterative	12	
Spiral	6	21
Agile-eXtreme Programming	8	
Agile-Scrum	14	
Feature Driven Development	5	20
Others	4	4

The typical team sizes of the participants ranged from 1 to 5 to more than 10 which is why the team size were grouped into three clusters. The clusters and their percentage are shown in Table 13.

Table 13: Percentage distribution of typical team size in the participants' organization

Typical team size (in persons)	Percentage
1 to 5	53%
6 to 10	41%
More than 10	6%

The typical project duration of participants' team ranges from less than 1 month to more than 2 years. The duration of the projects has been split into 4 ranges and is shown in Table 14.

Table 14: Frequency distribution of typical project distribution in the participants' organization

Typical project duration	Number of respondents
< 1 Month	4
>=1 month and < 6 months	20
>=6 months to <1 year	9
>=1 year to <2 years	9
>= 2 years	4

The organizations of the participants ranged from small (<50 people) to large (>2000 people). This was also broken down into four clusters as shown in Table 15.

Table 15: Percentage of organization size of the participants

Organization size	Percentage
< 50 people	50.00%
>=50 people and < 200 people	15.63%
>=200 people to <2000 people	9.38%
>= 2000 people	25.00%

3.3.3 Participants' geographic distribution

It is important to note that the geographical distributions were based on the participants and not the organizations in which they work because some of the companies were multinational and hence had branches all over the world so it was more important to consider the location of the participation rather than the location of the main branch of the organization. The geographical distributions of the participants are shown in Table 16.

Table 16: Percentage distribution of the participants' geographic location

Participants' geographic location	Percentage
Bangladesh	47%
Canada	31%
US	6%
Finland	6%
Australia	10%

According to government statistics reported in [54] on the experience of developers in China in 2007, it was found that 42% of the developers had less than two years of industry experience. This is comparable to the percentage of respondents in our survey

with 1 to 2 years of experience (38%). The number of developers in China with experience of 2 to 5 years is 38%, which is comparable to the percentage of respondents in the survey with more than 2 and less than 5 years of experience (31%). 20% of the developers in China had an experience of more than 5 years which in the case of the survey was 31%. Thus, if the percentage distribution of experience of the survey respondents is compared to the statistics of developers in China it can be said that the experience of the respondents of the survey is equal or more than that of the developers experience distribution in China. According to a report published in 2008, China was the 4th largest software producer in the world [26].

According to a survey on 1298 software professionals by Forrester research [16], in 2009 the ratio between agile and traditional development in development teams is approximately 0.82 (45% agile: 55% traditional). If we look at the ratio between agile and traditional development in the survey, it is approximately 0.95 (63% agile: 66% traditional; please see section 3.3.2).

So, from the above examples it can be concluded that the respondents of this survey are a good representation of a large population of software professionals.

3.4 Threats to validity

Sections 3.4.1 to 3.4.5 discuss different types of validity and how they were addressed in the study.

3.4.1 Construct validity

For construct validity, correct operational measures needs to be established based on theoretical constructs for the concept being studied [55]. The construct validation of the

questionnaire was critical to the overall validity of the study. The two types of *construct validity* that applied to the design of this questionnaire were *content* and *face* validity.

Content validity is concerned with whether the research instrument (i.e., in our case the questionnaire) properly represents the specific intended domain of content [32]. The different types of information mentioned in M1, M2, M4 and M5 are derived from text books (e.g., [4], [25] and [29]). The different communication mediums mentioned in M6 and M7 are derived from research work (e.g., [11] and [35]). So, all the contents of the parameters are rooted in literature.

Face validity is concerned with whether any contents (i.e., in our case the questionnaire) used for conducting the study are appropriately translated from the construct [32]. This is met in our study by involving three researchers in reviewing the questionnaire, both content and form.

3.4.2 Internal validity

Internal validity is of concern when a study tries to deduce that a relationship between two variables is causal [34]. This issue is not applicable to an exploratory study like ours.

3.4.3 External validity

External validity deals with the problem of knowing whether findings of a study can be generalized beyond the actual subjects of the study [34]. As we can see in section 3.3 that the participants of the survey had background experience in all the key areas of software engineering. The software models followed in the organization of the participants cover the different popular software development models. All the participants had substantial years of experience (1 to 15 years) and had experience working with a diverse magnitude of teams and organizations and hence were capable of grasping the problem posed in the questions. So it can be said that any findings common to this sample can be generalized

to the general population (i.e., software professionals participating in RE and SA processes).

3.4.4 Reliability

Reliability is concerned with minimizing the errors and researcher biases in a study [34]. Section 3.2.1 shows how researcher bias was removed from the questionnaire. Also another researcher was involved in the data collection and data analysis to remove errors and researcher bias from data collection and analysis.

3.4.5 Conclusion validity

Conclusion validity is about whether the conclusions we make based on the findings are reasonable [32]. Our conclusions are rooted in the results. The significance of our findings is tested using statistical analysis.

Chapter 4. Results and Interpretations

This chapter describes the results of our study and their interpretations. Sections 4.1, 4.2 and 4.3 describe the results and findings associated to research questions 1, 2 and 3 respectively. Section 4.4 summarizes the findings.

4.1 Types of information lost during RE and SA

Section 4.1.1 discusses the frequency of discussing different types of information during RE and SA, and section 4.1.2 discusses the frequency of documenting different types of information during RE and SA. Based on the frequency of discussion and frequency of documentation section 4.1.3 discusses the frequency of the types of information that are lost during RE and SA. Section 4.1.3 also discusses in details the three types of information that are lost most frequently.

4.1.1 Types of information discussed

Table 17 shows the frequency of different types of information discussed in RE and SA meetings. The first column in Table 17 shows the different types of information. The median frequency of discussion is shown in the second column, which was calculated using the one sample Wilcoxon signed rank test. The next two columns show the lower and upper bounds of the 95% confidence interval (CI) for the median. In Table 17, the 95% confidence interval in none of the rows include zero. So, all the medians are statistically reliable (please see section 3.2.3).

Table 17: Frequency[9] of the types of information discussed in RE and SA meetings

Types of information	Median	95% CI(-)	95% CI(+)
Requirements	4.500036	4.499997	4.999911
Rationale, priority, source and assumptions behind Requirements	3.999956	3.499954	4.499997
Architectural relevance of requirements	3.499968	2.999963	3.999972
Relationship between requirements	3.999994	3.499992	4.000041
Quality attributes	3.499969	2.999983	3.999981
Quality scenarios	3.000041	2.500054	3.500041
Use case scenarios	3.000019	2.500024	3.500078
Domain related information	3.499925	3.000047	3.999981
Issues	4.000026	3.500014	4.499991
Design decisions and rationale	3.999952	3.499947	4.000023
Architectural driver	2.999946	2.499959	3.499955
Tactics	2.999927	2.499966	3.499956
Patterns	2.500051	2.000042	3.000053
Other architectural artefacts	2.000013	1.500072	2.99999

The four most frequently discussed types of information are:

- Requirements (median 4.500036 which is between "Most of the time" and "Always" in the ordinal scale).
- Issues (median 4.000026 which is very close to "Most of the time" in the ordinal scale).
- Relationship between Requirements (median 3.999994 which is very close to "Most of the time" in the ordinal scale).
- Design decisions and rationale (median 3.999952 which is very close to "Most of the time" in the ordinal scale).

[9] The degree of precision of the values in the table may not be warranted in this study given the subject matter of lost information. However, we chose to leave the values produced by the statistical tool used and the rounding of is left to the reader.

4.1.2 Types of information documented

Table 18 shows the frequency of different types of information documented from RE and SA meetings. The first column in Table 18 shows the different types of information. The median frequency of documentation is shown in the second column, which was calculated using the one sample Wilcoxon signed rank test. The next two columns show the lower and upper bounds of the 95% confidence interval (CI) for the median. In Table 18, the 95% confidence interval in none of the rows include zero. So, all the medians are statistically reliable (please see section 3.2.3).

Table 18 Frequency of the types of information documented from RE and SA meetings

Types of information	Median	95% CI(-)	95% CI(+)
Requirements	4.000031	3.500078	4.500021
Rationale, priority, source and assumptions behind requirements	3.000015	2.500018	3.500076
Architectural relevance of requirements	2.999944	2.499986	3.499978
Relationship between requirements	3.000024	2.500004	3.500036
Quality attributes	2.999998	2.499957	3.499987
Quality scenarios	2.50009	2.000036	3.000004
Use case scenarios	2.999962	2.499949	3.500066
Domain related information	2.999939	2.499936	3.499928
Issues	3.000036	2.500049	3.500059
Design decisions and rationale	3.000007	2.500022	3.499962
Architectural driver	2.500011	2.000028	3.000055
Tactics	2.000002	1.999969	2.50005
Patterns	2.000023	1.500028	2.500052
Other architectural artefacts	2.000045	1.500036	2.999995

The four most frequently documented types of information are:

- Requirements (median 4.000031 which is very close to "Most of the time" in the ordinal scale).
- Issues (median 3.000036 which is very close to "Sometimes" in the ordinal scale).
- Relationship between Requirements (median 3.000024 which is very close to "Sometimes" in the ordinal scale).
- Rationale, Priority, Source and Assumptions behind Requirements (median 3.000015 which is very close to "Sometimes" in the ordinal scale).

4.1.3 Types of lost information

Table 19 shows the frequency of different types of information lost during RE and SA. This was derived by considering the frequency of discussion of information (discussed in section 4.1.1) and the frequency of documentation of information (discussed in section 4.1.2) as paired data. This was possible because the data were collected from the same participants and were based on the same types of information. The comparison between the paired data was done using the two sample Wilcoxon signed rank test (discussed in details in section 3.2.3). The first column in Table 19 shows the different types of information. The median frequency of information lost is shown in the second column, which was calculated from the rank test. The next two columns show the lower and upper bounds of the 95% confidence interval (CI) for the median. The final column shows the associated p-value for the test.

In Table 19, the 95% confidence interval in none of the rows include zero. So, all the medians are statistically reliable (please see section 3.2.3).

For a particular row (i.e., type of information, **i**) if the p-value is less than 0.05, then it shows that the result is statistically significant (please see section 3.2.3) and for that particular type of information, we can reject the null hypothesis:

$H(M3)_0$: *The information of type* **i** *is as frequently documented as discussed (i.e., no information is lost)*[10].

In Table 19, all the p-values are less than 0.05, which means that the null hypothesis can be rejected for all the different types of information. So, all the different types of information discussed are more or less lost during documentation.

Table 19 Frequency of the types of information lost during RE and SA

Types of information	Median	95% CI(-)	95% CI(+)	p-value
Requirements	1.00005	0.99994	Infinity[11]	0.00359
Rationale, priority, source and assumptions behind requirements	1.49997	0.99998	Infinity	0.00131
Architectural relevance of requirements	0.99997	0.99991	Infinity	0.00011
Relationship between requirements	1.00001	0.99999	Infinity	0.00097
Quality attributes	0.99999	0.00005	Infinity	0.00575
Quality scenarios	1.00002	0.99997	Infinity	0.00289
Use case scenarios	0.99996	0.0000375	Infinity	0.01833
Domain related information	1.00008	0.49997	Infinity	0.00673

[10] As this null hypothesis is related to metric M3 we name it $H(M3)_0$. Remaining null hypotheses are named similarly.

[11] As the null hypothesis is concerned with information loss (i.e., the frequency of discussion being greater than the frequency of documentation), one tailed confidence intervals are computed. It is important to note that here the sign of the confidence intervals are important rather than their numerical values because we are using them only for testing the statistical reliability (please see section 3.2.3) of the medians rather than using it for any further interpretation.

Issues	1.49999	0.99996	Infinity	0.00048
Design decisions and rationale	1.00001	0.99996	Infinity	0.0006
Architectural driver	1.00002	0.99996	Infinity	0.00169
Tactics	1.49997	1	Infinity	0.00148
Patterns	1	1	Infinity	0.00104
Other architectural artefacts	Could not be computed[12]			

The three types of information most frequently lost are:

- Issues (median 1.499992).
- Rationale, Priority, Source and Assumptions behind Requirements (median 1.499972).
- Tactics (median 1.49997).

These three are discussed in more detail in the remainder of this section.

Frequency of *issues* getting lost:

Figure 5 and Table 20 show that 50% of the respondents agreed that they document *issues* less frequently than they discuss. Among these 50 % respondents, the low documentation of *issues* is not as severe in 40%, who reported that they "rarely" or "sometimes" document *issues* less frequently than they discuss. For the remaining 10%, the low documentation of *issues* is severe as they reported that they "Always" or "Most of the time" document *issues* less frequently than they discuss.

[12] Because majority of responses were either tie or not sure values (17 not sure; 10 ties) the median could not be computed.

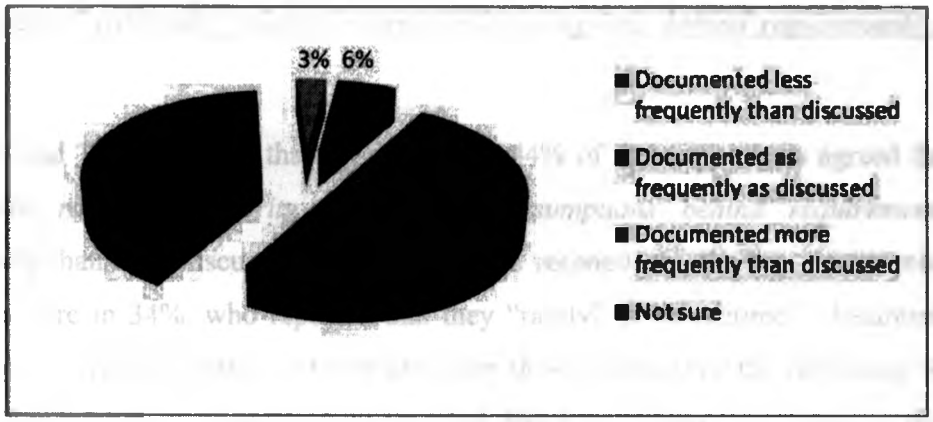

Figure 5: Frequency of *issues* getting lost

Table 20: Breakdown of *issues*-"Documented less frequently than discussed"

Breakdown of *issues*-"Documented less frequently than discussed"	Number of respondents	Percentages
Rarely	9	28.125
Sometimes	4	12.5
Most of the time	2	6.25
Always	1	3.125
Total	16	50

Issues are often raised in the form of questions (e.g., "Should we have databases from vendor X?", "Who will be responsible for keeping track of session events?", etc.) during meetings [53]. It is possible that some of these *issues* are also resolved during the meeting and during documentation, the resolution is documented instead of the issue, and the original issue gets lost. Open issues, on the other hand, may be documented for further discussion.

Frequency of *rationale, priority, source and assumptions behind requirements* getting lost:

Figure 6 and Table 21 show that approximately 44% of the respondents agreed that they document *rationale, priority, source and assumptions behind requirements* less frequently than they discuss. Among these 44% respondents, the low documentation is not as severe in 34%, who reported that they "rarely" or "sometimes" document these attributes of requirements less frequently than they discuss. For the remaining 9%, the low documentation is severe as they reported that they "Always" or "Most of the time" document these attributes of requirements less frequently than they discuss.

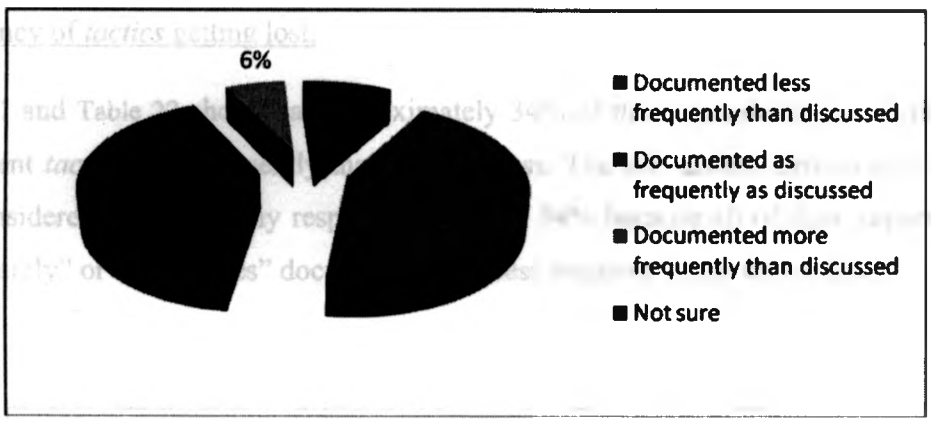

Figure 6: Frequency of *rationale, priority, source and assumptions behind requirements* getting lost

As we know explicit documentation of requirements are needed for several purposes like analysis, validation and management's approval whereas these attributes of requirements which basically details requirements are not. So, one plausible reason behind losing this information can be that developers are using (i.e., discussing) them to understand and selecting appropriate requirements for a particular product, version of a product or for a particular iteration but not documenting them explicitly.

Table 21: Breakdown of *rationale, priority, source and assumptions behind requirements*-"Documented less frequently than discussed"

Breakdown of *rationale, priority, source and assumptions behind requirements*-"Documented less frequently than discussed"	**Number of respondents**	**Percentages**
Rarely	8	25
Sometimes	3	9.375
Most of the time	3	9.375
Always	0	0
Total	14	43.75

Frequency of *tactics* getting lost:

Figure 7 and Table 22 show that approximately 34% of the respondents agreed that they document *tactics* less frequently than they discuss. The low documentation of *tactics* is not considered severe by any respondents of this 34% because all of them reported that they "rarely" or "sometimes" document *tactics* less frequently than they discuss.

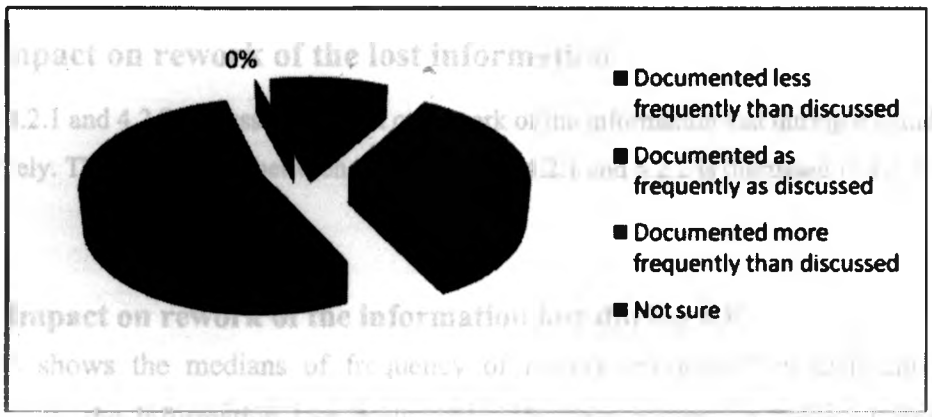

Figure 7: Frequency of *tactics* getting lost

Tactics are in lower level of abstraction than *patterns* and often *patterns* package *tactics* [4]. Table 17 shows that *tactics* are more frequently discussed than *patterns* whereas Table 18 shows that *tactics* are less frequently documented than *patterns*. One plausible reason for this can be that developers are using (i.e., discussing) different *tactics* to come up with the appropriate *patterns* for the required architecture but documenting the outputs of the discussion which are the *patterns*. Another possible reason can be that developers are not documenting the *patterns* explicitly so that developers in later stagers (e.g., design and coding) get more flexibility to replace them with appropriate low level design decisions.

Table 22: Breakdown of *tactics*-"Documented less frequently than discussed"

Breakdown of *tactics*-"Documented less frequently than discussed"	**Number of respondents**	**Percentages**
Rarely	6	18.75
Sometimes	5	15.625
Most of the time	0	0
Always	0	0
Total	11	34.375

4.2 Impact on rework of the lost information

Section 4.2.1 and 4.2.2 discuss the impact on rework of the information lost during RE and SA respectively. The comparison between the findings of 4.2.1 and 4.2.2 is discussed in 4.2.3.

4.2.1 Impact on rework of the information lost during RE

Table 23 shows the medians of frequency of rework introduced in different project activities by the information lost during RE. The first column in Table 23 shows the different project activities. The median frequency of introducing rework is shown in the second column, which was calculated using the one sample Wilcoxon signed rank test. The next two columns show the lower and upper bounds of the 95% confidence interval (CI) for the median.

In Table 23, the 95% confidence interval in none of the rows include zero. So, all the medians are statistically reliable (please see section 3.2.3).

The two activities where reworks are most frequently introduced by the information lost during RE are:

- Software architecting (median 3.500038 which is in between "Sometimes" and "Most of the time" in the ordinal scale).
- Requirements engineering (median 3.499957 which is in between "Sometimes" and "Most of the time" in the ordinal scale).

These two are discussed in more detail at the end of this section.

Table 23 Frequency of rework introduced in different activities by the information lost during RE

Project activities	Median	95% CI(-)	95% CI(+)
Requirements engineering	3.499957	2.999931	3.500035
Software architecting	3.500038	2.999935	4.00001
Design and coding	2.999938	2.499986	3.499957
Testing	2.999983	2.499975	3.000045
Maintenance	2.999971	2.499985	3.000043
Project management	2.499933	1.999964	2.999914
Quality control and assurance	2.999945	2.499966	3.499962
Process improvement	3.000055	2.500021	3.50001
System integration	2.500002	2.499968	3.000043

Frequency of rework introduced in *SA* due to the information lost during RE:

Figure 8 shows that only 6% (approximately) of respondents said that they never face the situation where rework is introduced in *SA* due to the information lost during RE whereas

84% (approximately) of the respondents agreed that at least some rework is introduced in *SA* from RE. Among these 84% of respondents, the frequency of introducing rework is not as severe in 34% (approximately), who reported that they faces the situation where rework is introduced in *SA* "rarely" or "sometimes". For the remaining 50%, introducing rework in *SA* by RE is "Always" or "Most of the time". It is important to note that *SA* gets most of its input from RE and hence information lost during RE is introducing rework most frequently in *SA*.

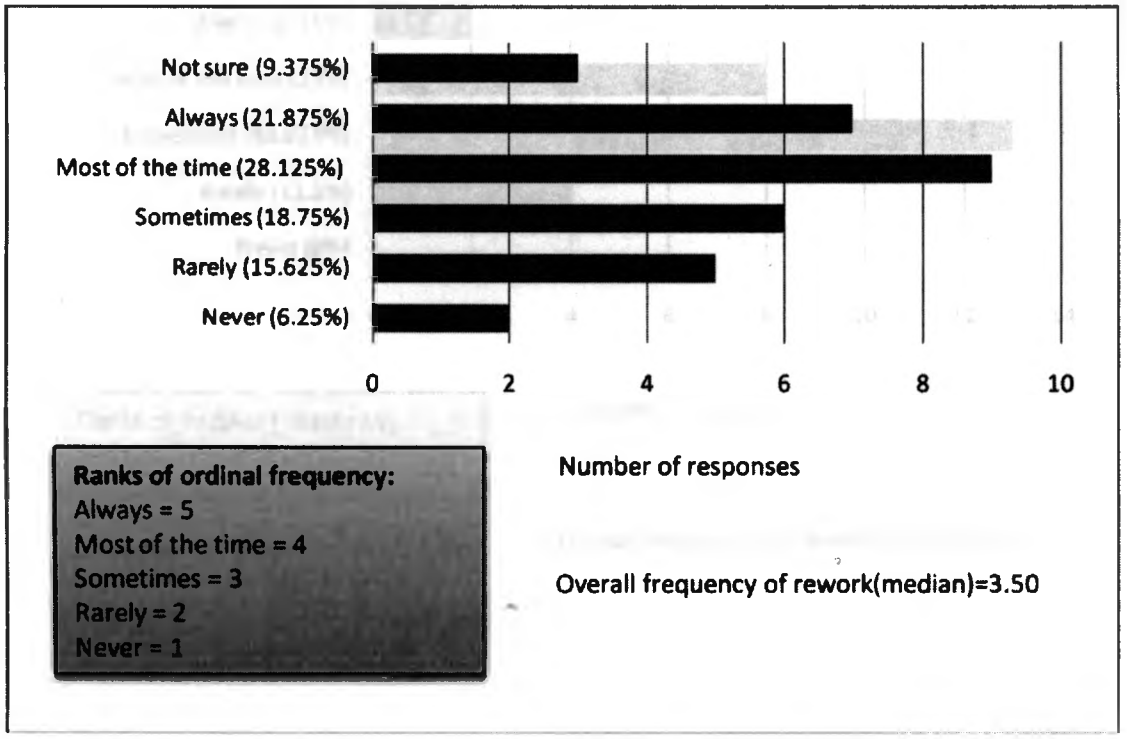

Figure 8: Frequency of rework introduced in *SA* due to the information lost during RE

Frequency of rework introduced in *RE* due to the information lost during RE:

Figure 9 shows that none of the respondents said that they never face the situation where rework is introduced in *RE* due to the information lost during RE whereas 84% (approximately) of the respondents agreed that some rework is introduced in *SA* from RE. Among these 84% respondents, the frequency of introducing rework is not as severe in

53% (approximately), who reported that they face the situation where rework is introduced in *RE* "rarely" or "sometimes". For the remaining 31% (approximately), introducing rework in *RE* is "Always" or "Most of the time". It means that if information are lost during RE than in later stages of RE or during RE work of later versions, rework is needed to retrieve or regenerate that lost information.

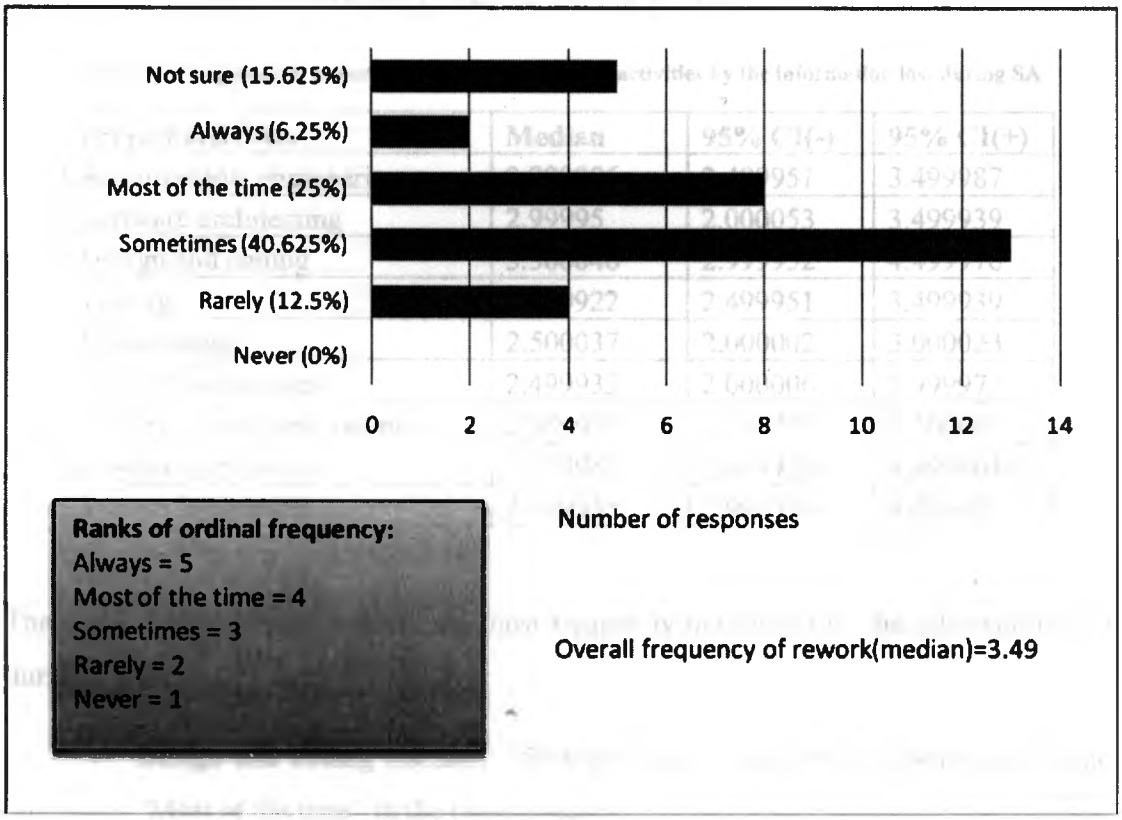

Figure 9: Frequency of rework introduced in *RE* due to the information lost during RE

4.2.2 Impact on rework of the information lost during SA

Table 24 shows the medians of frequency of rework introduced in different project activities by the information lost during SA. The first column in Table 24 shows the different project activities. The median frequency (i.e., the frequency of introducing rework) is shown in the second column, which was calculated using the one sample

Wilcoxon signed rank test. The next two columns show the lower and upper bounds of the 95% confidence interval (CI) for the median.

In Table 24, the 95% confidence interval in none of the rows include zero. So, all the medians are statistically reliable (please see section 3.2.3).

Table 24: Frequency of rework introduced in different activities by the information lost during SA

Project activities	Median	95% CI(-)	95% CI(+)
Requirements engineering	2.999996	2.499951	3.499987
Software architecting	2.99995	2.000053	3.499939
Design and coding	3.500046	2.999952	4.499976
Testing	2.999922	2.499951	3.499939
Maintenance	2.500037	2.000002	3.000023
Project management	2.499933	2.000006	2.999972
Quality control and assurance	2.999973	2.499997	3.500007
Process improvement	2.999987	2.499929	3.499964
System integration	3.499955	2.999954	4.000032

The two activities where reworks are most frequently introduced by the information lost during SA are:

- Design and coding (median 3.500046 which is in between "Sometimes" and "Most of the time" in the ordinal scale).
- System integration (median 3.499955 which is in between "Sometimes" and "Most of the time" in the ordinal scale).

These two are discussed in more detail in the remainder of this section.

Frequency of rework introduced in *design and coding* due to the information lost during SA:

Figure 10 shows that only 3% (approximately) of respondents said that they never face the situation where rework is introduced in *design and coding* due to the information lost during SA whereas 66% (approximately) of the respondents agreed that at least some rework is introduced in *design and coding* from SA. Among these 66% respondents, the frequency of introducing rework is not as severe in 25%, who reported that they face the situation where rework is introduced in *design and coding* "rarely" or "sometimes". For the remaining 41%, introducing rework in *design and coding* by SA is "Always" or "Most of the time". It is important to note that during *design and coding* developers mainly use the software artefacts produced by SA and hence information lost during SA is introducing rework most frequently in *design and coding*.

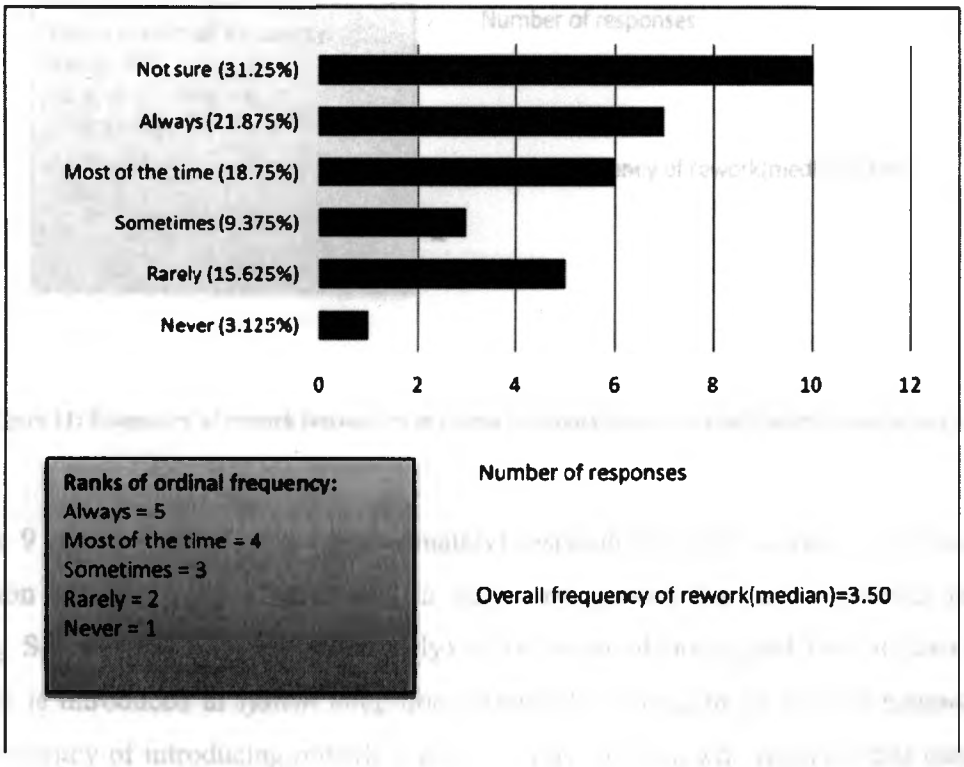

Figure 10: Frequency of rework introduced in *design and coding* due to the information lost during SA

Frequency of rework introduced in *system integration* due to the information lost during SA:

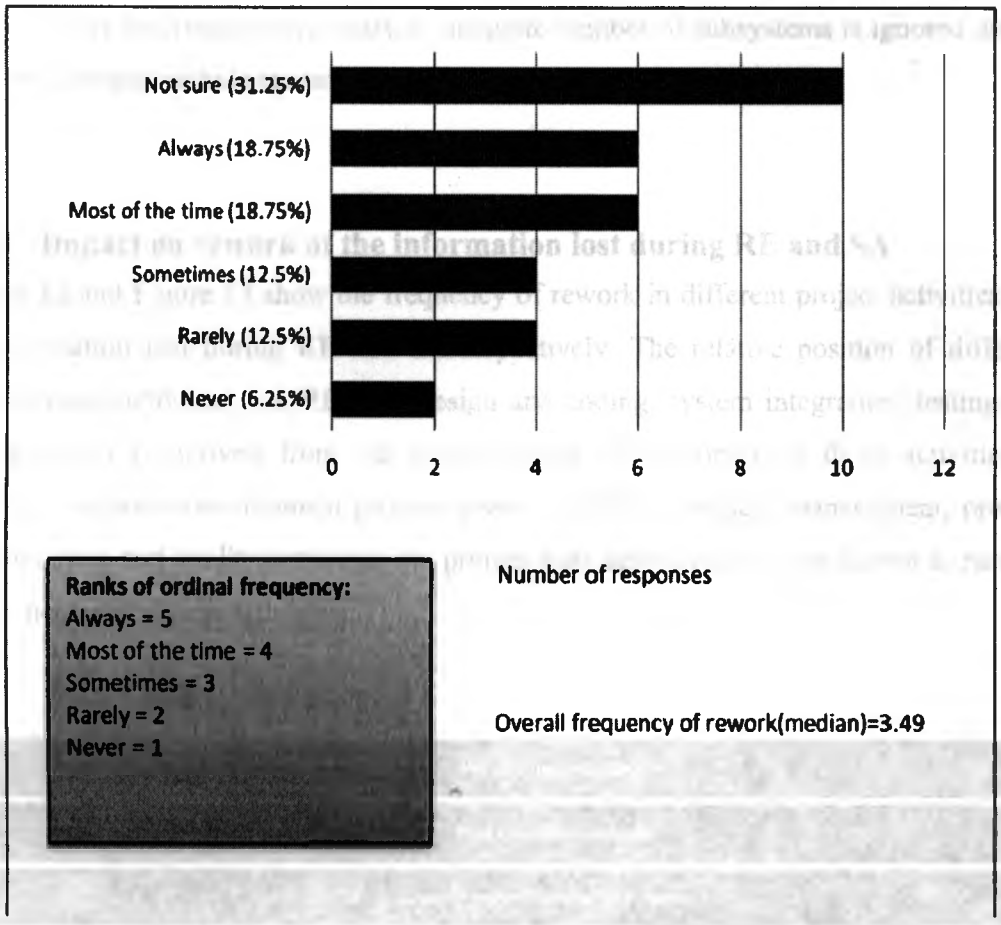

Figure 11: Frequency of rework introduced in *system integration* due to the information lost during SA

Figure 9 shows that only 6% (approximately) respondents said that they never face the situation where rework is introduced in *system integration* due to the information lost during SA whereas 63% (approximately) of the respondents agreed that at least some rework is introduced in *system integration* from SA. Among these 63% of respondents, the frequency of introducing rework is not as severe in 25%, who reported that they face the situation where rework is introduced in *system integration* "rarely" or "sometimes".

For the remaining 38% (approximately), introducing rework in *system integration* by SA is "Always" or "Most of the time". Some of the architectural problems (such as, interface incompatibility, disagreement between two components about which one invokes the other etc.) can usually be found during *system integration* instead of design and coding [4]. So if any information necessary to integrate number of subsystems is ignored during SA then it might include rework in *system integration.*

4.2.3 Impact on rework of the information lost during RE and SA

Figure 12 and Figure 13 show the frequency of rework in different project activities due to information lost during RE and SA respectively. The relative position of different development activities (i.e, RE, SA, design and coding, system integration, testing and maintenance) is derived from the generalization of the order of these activities in different software development process given in [29]. As project management, process improvement and quality assurance are project wide activities, they are shown as parallel to all the development activities.

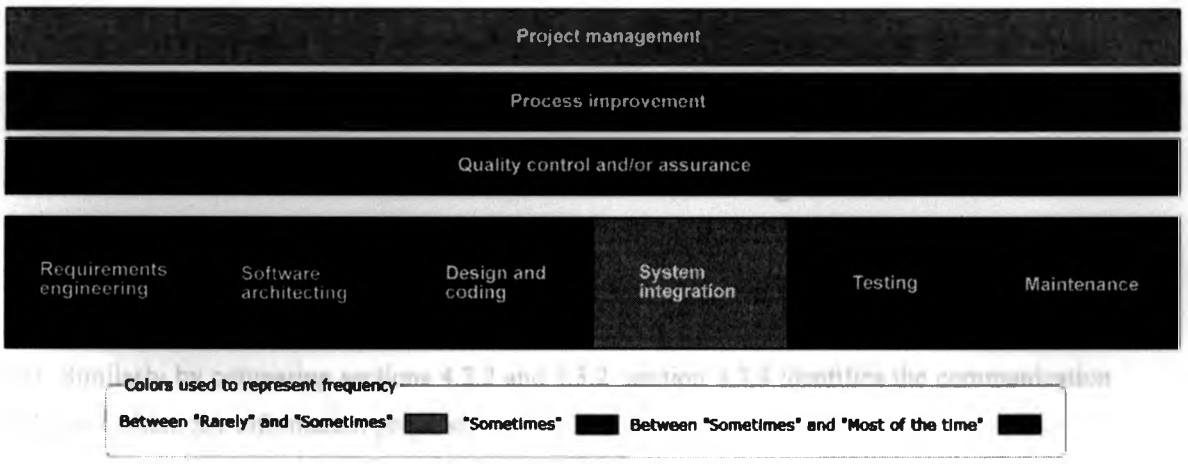

Figure 12: Frequency of rework introduced in different activities by the information lost during RE

As we can see from Figure 12 that the information lost during RE introduces rework in *SA* and *RE* most frequently. Similarly in Figure 13 the information lost during SA introduces rework in *design and coding* and *system integration* most frequently. It is important to note that in both the cases the impact is more significant in activities which are immediate to the activity during which information are lost. One plausible reason behind this can be that the importance of any information from a particular activity are observed in immediate activities and mitigated through rework so that the impact is not directly significant in later activities.

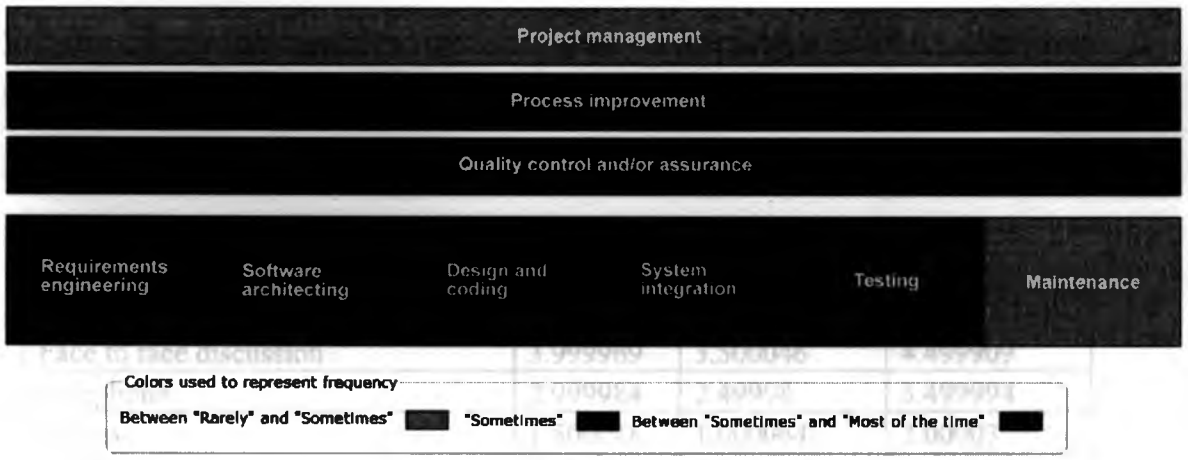

Figure 13: Frequency of rework introduced in different activities by the information lost during SA

4.3 Communication mediums where information gets lost

Section 4.3.1 and 4.3.2 discusses the frequency of using different communication mediums for RE and SA discussion respectively. Based on the comparison between findings of sections 4.2.1 and 4.3.1 and section 4.3.3 identifies the communication mediums where RE information gets lost. Similarly by comparing sections 4.2.2 and 4.3.2 section 4.3.4 identifies the communication mediums where SA information gets lost.

4.3.1 Communication mediums used for RE discussion

Table 25 shows the medians of frequency of using different communication mediums for RE discussion. The first column in Table 25 shows the different communication mediums. The median frequency of using different communication mediums for RE discussion is shown in the second column, which was calculated using the one sample Wilcoxon signed rank test. The next two columns show the lower and upper bounds of the 95% confidence interval (CI) for the median.

In Table 25, the 95% confidence interval in none of the rows include zero. So, all the medians are statistically reliable (please see section 3.2.3).

Table 25 Frequency of using different communication mediums for RE discussion

Communication mediums	Median	95% CI(-)	95% CI(+)
Face to face discussion	3.999969	3.500046	4.499909
Telephone	2.999984	2.49998	3.499994
Survey	1.500048	1.000064	2.00003
Email	3.999986	3.499993	4.000037
Teleconference	3.000026	2.500001	3.499932
Text chat	3.499958	2.999978	3.500019
Wiki, blog, forum, etc.	2.500058	2.000011	3.000045

The three most frequently used communication mediums for RE discussions are:

- Email (median 3.999986; which is very close to "Most of the time" in the ordinal scale).
- Face to face discussion (median 3.999969; which is very close to "Most of the time" in the ordinal scale).
- Text chat (median 3.499958; which is in between "Sometimes" and "Most of the time" in the ordinal scale).

4.3.2 Communication mediums used for SA discussion

Table 26 shows the medians of frequency of using different communication mediums for SA discussion. The first column in Table 25 shows the different communication mediums. The median frequency of using different communication mediums for SA discussion is shown in the second column, which was calculated using the one sample Wilcoxon signed rank test. The next two columns show the lower and upper bounds of the 95% confidence interval (CI) for the median.

In Table 26, the 95% confidence interval in none of the rows include zero. So, all the medians are statistically reliable (please see section 3.2.3).

Table 26: Frequency of using different communication mediums for SA discussion

Communication mediums	Median	95% CI(-)	95% CI(+)
Face to face discussion	3.000011	2.500023	3.500017
Telephone	2.499948	1.999989	2.999994
Survey	1.49998	1	1.999985
Email	3.00001	2.500018	3.99996
Teleconference	2.500058	2.000061	3.000061
Text chat	2.999998	2.000048	3.000021
Wiki, blog, forum, etc.	2.000037	1.500015	2.500008

The three most frequently used communication used for SA discussions are:

- Face to face discussion (median 3.000011 which is very close to "Sometimes" in the ordinal scale).
- Email (median 3.00001 which is very close to "Sometimes" in the ordinal scale).
- Text chat (median 2.999998 which is in between "Sometimes" in the ordinal scale).

4.3.3 Communication mediums where RE information gets lost

We found in 4.2.1 that *RE* and *SA* are the two activities where most frequently reworks are introduced by the information lost during RE. So to find the communication mediums where RE information are lost very frequently in Table 27 we calculated the Spearman's correlation coefficient (ρ) and the associated p-value to see whether there is any correlation between the frequency of rework introduced in these activities by the information lost during RE and the frequency of using any particular communication mediums for RE discussion.

The first column in Table 27 shows the different communication mediums used for RE. The Spearman's correlation coefficient (ρ) between the frequency of using these communication mediums for RE discussion and the frequency of rework in later RE introduced by information lost during RE, and associated p-values are shown in the next two columns respectively. Similarly the ρ between the frequency of using these communication mediums and frequency of rework in SA, and associated p-value are shown in the last two columns.

For a particular row (i.e., communication medium, **c**) and particular activity (**a**) if the p-value is less than 0.05, then it shows that the result is statistically significant (please see section 3.2.3) and for that particular communication medium, we can reject the null hypothesis:

$H(M8)_0$: *There is no correlation between how frequently communication medium* **c** *is used for RE and how frequently rework is introduced in project activity* **a** *by information lost during RE (i.e.,* **c** *is not related to information loss).*

If the null hypothesis for a particular communication medium can be rejected then we have to examine the ρ value (please see section 3.2.3). If the absolute value of ρ is greater than 0.7 then **c** is highly correlated to information loss. If the absolute value of ρ is less than or equal to 0.7 but greater than 0.4 then **c** is moderately correlated to information loss. If the absolute value of ρ is less than or equal 0.4 then the correlation is week. The sign of ρ indicates whether the correlation is positive or negative. In this case positive correlation implies that the communication medium is (most likely) the reason behind information loss (considering all other factors are constant) and negative correlation implies that the communication medium is (most likely) preventing information loss (considering all other factors are constant).

Table 27 Correlation coefficient between the frequency of rework introduced in RE and SA by the information lost during RE and the frequency of using any particular communication medium for RE

	Rework introduced during			
	RE		SA	
Mediums used in RE	**ρ**	p-value	**ρ**	p-value
Face to face discussion	0.523016	0.005123	0.619578	0.000338
Telephone	0.5687	0.001966	0.502116	0.005511
Survey	-0.08137	0.6866	0.194386	0.3123
Email	0.098608	0.6246	0.1992	0.3002
Teleconference	0.482113	0.01088	0.505616	0.005142
Text chat	0.456615	0.01666	0.402876	0.03024
Wiki, blog, forum etc.	-0.47377	0.01255	-0.50797	0.004905

As we can see, face to face discussion, telephone, teleconference and text chat shows moderate positive correlation with both *RE* and *SA*. As the associated p-values are also less than 0.05 in all these four cases we can reject the null hypothesis for face to face discussion, telephone, teleconference and text chat and say that face to face discussion, telephone, teleconference and text chat has a moderate chance of losing information if used as communication mediums for RE discussion. Whereas Wiki, blog, forum has moderate negative correlation with both *RE* and *SA* and also the associated p-value is less

than 0.05. So, we can we can reject the null hypothesis for wiki, blog and forum and say that wiki, blog and forum has moderate chance of not losing information if used as communication mediums for RE discussion.

Because of high p-value (0.6866 for *RE* and 0.3123 for *SA*) we cannot take any conclusive decision about using survey for RE discussion. Similarly for high p-value (0.6246 for *RE* and 0.3002 for *SA*) we cannot take any conclusive decision about using email for RE discussion.

4.3.4 Communication mediums where SA information gets lost

We found in 4.2.2 that *design and coding* and *system integration* are the two activities where most frequently reworks are introduced by the information lost during SA. So to find the communication mediums where SA information are lost very frequently in Table 28 we calculated the Spearman's correlation coefficient (ρ) and associated p-value to see whether there is any correlation between the frequency of rework introduced in these activities by the information lost during SA and the frequency of using any particular communication mediums for SA discussion.

The first column in Table 28 shows the different communication mediums used for SA. The Spearman's correlation coefficient (ρ) between the frequency of using these communication mediums for SA discussion and the frequency of rework in *design and coding* introduced by information lost during SA, and associated p-value are shown in next two columns respectively. Similarly the ρ between the frequency of using these communication mediums and frequency of rework in *system integration*, and associated p-value are shown in the last two columns.

For a particular row (i.e., communication medium, **c**) and particular activity (**a**) if the p-value is less than 0.05, then it shows that the result is statistically significant (please see section 3.2.3) and for that particular communication medium, we can reject the null hypothesis:

H(M9)$_0$: *There is no correlation between how frequently communication medium* **c** *is used for SA and how frequently rework is introduced in project activity* **a** *by information lost during SA (i.e.,* **c** *is not related to information loss).*

If the null hypothesis for a particular communication medium can be rejected then we have to examine the ρ value. The interpretations for the ρ value are already discussed in section 4.3.3.

Table 28 Correlation coefficient between the frequency of rework introduced in design and coding and system integration by the information lost during SA and the frequency of using any particular communication medium for SA

	Rework introduced during			
	Design and coding		System integration	
Mediums used in SA	ρ	p-value	ρ	p-value
Face to face discussion	0.160443	0.4757	0.160514	0.4755
Telephone	0.762911	0.0000365	0.803741	0.00000662
Survey	-0.21807	0.3423	-0.15592	0.4997
Email	0.405414	0.06826	0.369283	0.09946
Teleconference	0.692217	0.000358	0.704359	0.000253
Text chat	0.593548	0.003591	0.651064	0.001032
Wiki, blog, forum etc.	-0.6258	0.001838	-0.65682	0.000898

As we can see, telephone shows high positive correlation (more than 0.7) with both *design and coding* and *system integration*. Teleconference shows high positive correlation

with *system integration* but moderate positive correlation with *design and coding*. Text chat shows moderate positive correlation with both *design and coding* and *system integration*. As the associated p-values are also <0.05 in all these four cases we can reject the null hypothesis for telephone, teleconference and text chat, and say that telephone has high chance but teleconference and text chat have moderate chance of losing information if used as communication mediums for SA discussion. Whereas wiki, blog, forum have moderate negative correlation with both *design and coding* and *system integration* and also the associated p-value is < 0.05. So, we can reject the null hypothesis for wiki, blog and forum and say that wiki, blog and forum have moderate chance of not losing information if used as communication mediums for SA discussion.

Because of high p-value (0.4757 for *design and coding* and 0.4755 for *system integration*) we cannot take any conclusive decision about using face to face discussion for SA. Similarly for high p-value we cannot take any conclusive decision about using survey (p-value=0.3423 for *design and coding* and 0.4997 for *system integration*) and email (p-value=0.06826 for *design and coding* and 0.09946 for *system integration*) for SA discussion.

4.4 Summary of the findings

Our result shows that all types of information are not documented as frequently as they are discussed in RE and SA meetings. As a result different types of information get lost. The types of information that are lost most frequently during RE and SA due to lack of documentation are:

- Issues.
- Rationale, Priority, Source and Assumptions behind Requirements.
- Tactics.

Lost information during RE and SA introduces rework in different project activities. Information lost during RE introduces rework in *SA* and *RE* most frequently whereas information lost during SA introduces rework in *design and coding* and *system integration* most frequently.

Also, different types of communication mediums are used for RE and SA discussions. The three most frequent communication mediums used for RE and SA discussions are Face to face discussion, Email and Text chat. Face to face discussion, telephone, teleconference and text chat have moderate chance of losing information if used as communication mediums for RE discussion Telephone has high chance but teleconference and text chat have moderate chance of losing information if used as communication mediums for SA discussion. On the other hand Wiki, blog and forum have moderate chance of not losing information if used as communication mediums for RE or SA discussions.

So far there was no scientific data available on the above issues. Our findings can therefore be considered as an important step toward building knowledge on characteristics and impact of information lost during RE and SA.

Chapter 5. Implications

This chapter discusses implications of our results. Sections 5.1, 5.2 and 5.3 discuss the implications on process, tools and empirical research respectively.

5.1 Implications on industry

Our result identifies the communication mediums where information are lost most frequently during RE and SA. This result could be helpful for choosing communication mediums for RE and SA discussion based on the project type. For example, during the development of a safety critical system, or a system which includes regulatory requirements from the customer, the communication should be chosen carefully. In such a situation a detrimental (in terms of information loss) communication medium should be avoided or used with caution.

5.2 Implications on tools

There are already some research tools that are available (e.g., [14] and EGRET [38]) and tools used in industry (e.g., Rational Team Concert [20]), which provide traceability between software artefacts and communication artefacts (e.g., meeting videos, email, chat, etc.). Our findings could be useful while further developing such tools. For example, our result identifies the types of information that are lost most frequently. Also, the communication mediums where information is lost most frequently are identified. So while further developing such tools these software artefacts and communication artefacts could be targeted.

5.3 Implications on empirical research

Based on the findings of this study, we raise the following hypotheses which might be tested through further empirical investigation.

H1: *Information lost during a particular project activity (e.g., RE, SA, design and coding, testing etc.) will have more impact on the immediate activities compared to later activities.*

Section 4.2.3 shows that the information lost during RE introduces rework most frequently on *SA*, and the information lost during SA introduces rework most frequently on *design and coding*. Here *SA* and *design coding* are the two immediate activities after RE. Also, later activities such as *system integration* (in the case RE) and *testing* (in the case of SA) are less impacted in terms of rework due to information loss. These findings motivate us to state the above hypothesis.

H2: *If a particular communication medium (e.g., face to face discussion, email, chat etc.) is detrimental (in terms of information loss) to use for RE then it is also detrimental to use for SA.*

This hypothesis emerges from the findings in sections 4.3.3 and 4.3.4. Section 4.3.3 shows that *telephone*, *teleconference* and *text chat* has moderate chance of losing information if used as communication mediums for RE discussion. Whereas section 4.3.4 shows that *telephone* has high chance but *teleconference* and *text chat* has moderate chance of losing information if used as communication mediums for SA discussion.

H3: *If a particular communication medium (e.g., face to face discussion, email, chat etc.) is safe (in terms of information loss) to use for RE then it is also safe to use for SA.*

This hypothesis is rooted in the findings in sections 4.3.3 and 4.3.4. These sections show that *wiki, blog and forum* has moderate chance of not losing information if used as communication mediums for RE or SA discussions.

To test these hypotheses further empirical investigations would need to be designed. Also, our questionnaire and data analysis method can be considered as a primary template for investigating other areas of software engineering (such as design, coding, testing, maintenance, etc.) from the point of view of information lost due to lack of documentation.

Chapter 6. Limitations, Future Work and Conclusions

This chapter discusses the limitations of the study, future and ongoing research in sections 6.1 and 6.2 respectively. Finally section 6.3 concludes the thesis.

6.1 Limitations

To best of our knowledge, the following is a list of the limitations of the study:

- As a research strategy survey has some inherent limitations. For example, data collected through survey reflects the participants' perception of the situation rather than the actual situation. Because we used survey as our research strategy our study has the limitations of survey research.
- Availability sampling and snowball sampling was used as sampling technique for the study rather than random sampling. This is a limitation towards generalizing the result to a larger population.
- While measuring the impact of the information lost due to lack of documentation, we only focused on impact on rework and excluded other aspects such as cost, quality etc.
- We could not obtain the perception of the participants and other practitioners towards the findings of this study.
- We could not make a comparison between the values of our metrics in the back drop of contextual information such as development models followed, participants' background experience, geographical location etc.

All these limitations are mainly due to the resource and time constraints. We intend to overcome these limitations in our future work. We also encourage other researchers to conduct confirmatory and complementary studies in other domains and contexts to help build grounded theory on the characteristics and impact of information lost during RE and SA due to lack of documentation.

6.2 Ongoing and future work

Fifty five hours of audio and video data were collected from four projects where the subjects are eliciting requirements and developing a software architecture based on the elicited requirements. There were in total 17 subjects (13 developers working as requirements engineers and software architect; 4 customers) in those studies. Two of the projects focus on a banking system and the remaining focus on a garage door controller. Also the software artefacts (i.e., requirements and architectural artefacts) developed by the participants were collected. This data will be used in the future to compare the findings of the survey study by answering research question 1 and 2, including cost, quality and time issues. Also the following emerging research questions will be investigated from those projects data:

Q1: *What fraction of the information lost due to documentation during RE and SA are needed in later RE and SA work?*

Q2: *What is the difference between the lost information and the recreated information (if needed as mentioned in Q1 in terms of quality and focus?*

Also, a proof-of-concept prototype tool was developed which provides traceability between meeting videos and software artefacts (i.e., requirements and architectural

artefacts) based on the developer's (whoever was using the tool during meeting) action in the tool. The tool also provides facility to develop traceability between different software artefacts through tagging (e.g., image tagging). Further enhancement of the tool will be done based on the findings of the analysis of project data mentioned above. The comparison of return on investment in terms of cost, quality and effort of using the tool for projects, through further empirical investigation is also part of future research plans.

6.3 Conclusions

While the evidence of information loss during RE and SA can be found from literature (e.g., [35],[15]), no scientific studies focusing on the characteristics and impact of the information lost during RE and SA has ever been conducted. In this thesis, we described an industrial survey, involving 32 software professional from 23 different companies having 1 to 15 years of experience. Our study investigated the characteristics and impact on rework of the information lost during RE and SA due to lack of documentation.

We found the types of information that are lost most frequently during RE and SA due to lack of documentation are "issues", "rationale, priority, source and assumptions behind requirements" and "tactics" (please see section 4.1.3). Our results show that information lost during RE introduces rework in SA and RE most frequently whereas information lost during SA introduces rework in design and coding and system integration most frequently (please see section 4.2.3). We also found that face to face discussion, telephone, teleconference and text chat have moderate chance of losing information if used as communication mediums for RE discussion (please see section 4.3.3). Telephone has high chance but teleconference and text chat have moderate chance of losing information if used as communication mediums for SA discussion (please see section 4.3.4). On the other hand Wiki, blog and forum have moderate chance of not losing information if used as communication mediums for RE or SA discussions (please see sections 4.3.3and 4.3.4).

Our results have implications in the industry as it could help the practitioners to decide which communication mediums to avoid or to use with caution based on project type (please see section 5.1). But we advise caution when making business decision based on the results of this fundamental study alone.

Our results also have implication in research as number of new hypothesis emerge from it (please see section 5.3). We encourage other researchers to conduct confirmatory and complementary studies in other domains and contexts to help build grounded theory on characteristics and impact of information lost due to lack of documentation.

References:

[1] Ambler, S. W. (n.d.). *Agile Architecture: Strategies for Scaling Agile Development.* Retrieved November 2009, from Agile Modeling: http://www.agilemodeling.com/essays/agileArchitecture.htm

[2] Ambler, S. W. (n.d.). *Introduction to User Stories*. Retrieved November 2009, from Agile Modeling: http://www.agilemodeling.com/artifacts/userStory.htm

[3] Basili, V. R., Caldiera, G., & Rombach, H. (1994). The Goal Question Metric Approach. In J. J. Marciniak (Ed.), *Encyclopedia of Software Engineering* (2nd ed., Vol. 2, pp. 528-532). John Wiley & Sons, Inc.

[4] Bass, L., Clements, P., & Kazman, R. (2003). *Software Architecture in Practice* (2nd ed.). Addison Wesley.

[5] Beecham, S., Hall, T., & Rainer, A. (2003). Software Process Improvement Problems in Twelve Software Companies: An Empirical Analysis. *Empirical Software Engineering , 8* (1), 7-42.

[6] Bell, D. (2004). *UML basics: The class diagram.* Retrieved November 2009, from IBM developerWorks : Rational : Technical library view: http://www.ibm.com/developerworks/rational/library/content/RationalEdge/sep04/bell/

[7] Berenbach, B. (2006). Impact of Organizational Structure on Distributed Requirements Engineering Processes: Lessons Learned. *International workshop on Global software development for the practitioner, 28th International Conference on Software Engineering (ICSE 06)* (pp. 15-19). Shanghai, China: ACM.

[8] Cao, L., & Ramesh, B. (2008). Agile Requirements Engineering Practices: An Empirical Study. *IEEE Software , 25* (1), 60-67.

[9] Clements, P. C. (1996). A Survey of Architecture Description Languages. *8th International Workshop on Software Specification and Design* (pp. 16-25). Schloss Velen, Germany: IEEE Computer Society.

[10] Dalgaard, P. (2002). *Introductory Statistics with R* (1st ed.). Springer.

[11] Damian, D., Izquierdo, L., Singer, J., & Kwan, I. (2007). Awareness in the Wild: Why Communication Breakdowns Occur. *International Conference on Global Software Engineering* (pp. 81-90). Munich: IEEE Computer Society.

[12] Dekleva, S. (1992). Delphi Study of Software Maintenance Problems. *IEEE Conference on Software Maintenance, ICSM 1992*, (pp. 10-17). Orlando, Florida, USA.

[13] Forward, A., & Lethbridge, T. C. (2002). The Relevance of Software Documentation, Tools and Technologies: A Survey. *The ACM Symposium on Document Engineering*, (pp. 26-33). McLean, Vancouver, Canada.

[14] Gall, M., Bruegge, B., & Berenbach, B. (2006). Towards a Framework for Real Time Requirements Elicitation. *First International Workshop on Multimedia Requirements Engineering, 2006. MERE '06.* (pp. 4-4). Minneapolis/St. Paul, Minnesota: IEEE Computer Society.

[15] George, B., Bohner, S. A., & Prieto-Diaz, R. (2004). Software information leaks: A complexity perspective. *Ninth IEEE International Conference on Engineering Complex Computer Systems (ICECCS'04)* (pp. 239-248). Florence, Italy: IEEE Computer Society.

[16] Grant, T., & West, D. (n.d.). *Agile Adoption In The Real World.* Retrieved November 2009, from Forrester Research: http://www.forrester.com/rb/teleconference/agile_adoption_in_real_world/q/id/5881/t/1

[17] Grinnell, R., & Unrau, Y. (2008). *Social Work Research and Evaluation: Foundations of Evidence-Based Practice* (8th ed.). Oxford University Press.

[18] Hofmeister, C., Nord, R. L., & Soni, D. (1999). Describing Software Architecture with UML. *TC2 First Working IFIP Conference on Software Architecture (WICSA1)*, (pp. 145-160). San Antonio, Texus, USA.

[19] *IBM - Rational Software Architect.* (n.d.). Retrieved November 2009, from IBM - United States: http://www-01.ibm.com/software/awdtools/architect/swarchitect/

[20] *IBM Software - Rational Team Concert.* (n.d.). Retrieved November 2009, from IBM - United States: http://www-01.ibm.com/software/awdtools/rtc/

[21] IEEE-std-1471-2000: Recommended Practice for Architectural Description of Software-Intensive Systems. IEEE, http://standards.ieee.org/.

[22] IEEE-std-830-1998: Recommended practice for software requirements specifications. IEEE, http://standards.ieee.org/.

[23] Kitchenham, B. A., & Pfleeger, S. L. (2002). Principles of survey research: part 3: constructing a survey instrument. *ACM SIGSOFT Software Engineering Notes* , 27 (2), 20-24.

[24] Kitchenham, B. A., Pfleeger, L. S., Pickard, L. M., Jones, P. W., Hoaglin, D. C., Emam, K. E., et al. (2002). Preliminary Guidelines for Empirical Research in Software Engineering. *IEEE Transactions on Software Engineering* , 28 (8), 721-734.

[25] Kotonya, G., & Sommerville, I. (1998). *Requirements Engineering: Processes and Techniques.* John Wiley & Sons.

[26] LaLonde, J. (2008). *China becomes world's 4th largest Software producer.* Retrieved November 2009, from China Bits: http://bits.typepad.com/chinabits/2008/06/china-becomes-w.html

[27] Lethbridge, T. C., Singer, J., & Forward, A. (2003). How Software Engineers Use Documentation: The State of the Practice. *IEEE Software* , *20* (6), 35-39.

[28] *MySQL :: The world's most popular open source database.* (n.d.). Retrieved November 2009, from MySQL :: The world's most popular open source database: http://www.mysql.com/

[29] Pfleeger, S. L., & Atlee, J. M. (2005). *Software engineering: theory and practice* (3rd ed.). Prentice Hall.

[30] *PHP: Hypertext Preprocessor.* (n.d.). Retrieved November 2009, from PHP: Hypertext Preprocessor: http://www.php.net/

[31] *Requirements Tools.* (n.d.). Retrieved November 2009, from Volere Requirements Resources: http://www.volere.co.uk/tools.htm

[32] *Research Methods Knowledge Base.* (n.d.). Retrieved November 2009, from Social Research Methods: http://www.socialresearchmethods.net/kb/

[33] Rowntree, D. (1981). *Statistics without tears: A primer for non-mathematicians.* Penguin.

[34] Runeson, P., & Höst, M. (2009). Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering* , 14 (2), 131-164.

[35] Schneider, K., Stapel, K., & Knauss, E. (2008). Beyond Documents: Visualizing Informal Communication. *Requirements Engineering Visualization, 2008. REV'08* (pp. 31-40). Barcelona, Spain: IEEE Computer Society.

[36] Sheppard, S. B., Kruesi, E., & Bailey, J. W. (1982). An empirical evaluation of software documentation formats. *Conference on Human Factors in Computing Systems*, (pp. 121-124). Gaithersburg, Maryland, USA.

[37] Singer, J., Lethbridge, T., Vinson, N., & Anquetil, N. (1997). An Examination of Software Engineering Work Practices. *CASCON '97 - IBM Centre for Advanced Studies Conference*, (pp. 209-223). Toronto, Canada.

[38] Sinha, V., Sengupta, B., & Chandra, S. (2006). Enabling Collaboration in Distributed Requirements Management. *IEEE Software , 23* (5), 52-61.

[39] Smolander, K., & Päivärinta, T. (2002). Describing and Communicating Software Architecture in Practice: Observations on Stakeholders and Rationale. *CAiSE'02 - The Fourteenth International Conference on Advanced Information Systems Engineering* (pp. 117-133). Toronto, Canada: Springer Berlin / Heidelberg.

[40] *Software Architecture Design, Visual UML & Business Process Modeling – from Borland.* (n.d.). Retrieved November 2009, from Borland Software Solutions for Change Management, Asset Management, Test Automation, SDLC and more – Borland: http://www.borland.com/us/products/together/index.html

[41] Solemon, B., Sahibuddin, S., & Ghani, A. A. (2008). Requirements engineering problems in 63 software companies in Malaysia. *International Symposium on Information Technology, 2008. ITSim 2008, 4*, pp. 1-6. Kuala Lumpur, Malaysia.

[42] Sommerville, I. Software Documentation. In R. H. Thayer, & M. I. Christensen (Eds.), *Software Engineering, Vol 2: The Supporting Processes*. Wiley-IEEE Press.

[43] Sommerville, I., & Ransom, J. (2005). An Empirical Study of Industrial Requirements Engineering Process Assessment and Improvement. *ACM Transactions on Software Engineering and Methodology (TOSEM)* , 14 (1), 85-117.

[44] *StatisticalHelp from StatsDirect*. (n.d.). Retrieved November 2009, from StatsDirect: Software to improve statistical practice: http://www.statsdirect.com/help/

[45] Survey on Documentation and Feed forward information in Software Projects (http://survey.obviousdesign.net/).

[46] Tamhane, A. C., & Dunlop, D. D. (2000). *Statistics and Data Analysis: From Elementary to Intermediate*. Prentice-Hall.

[47] Tang, A., Babar, M. A., Gorton, I., & Han, J. (2006). A survey of architecture design rationale. *Journal of Systems and Software , 79* (12), 1792-1804.

[48] *The R Project for Statistical Computing*. (n.d.). Retrieved November 2009, from The R Project for Statistical Computing: http://www.r-project.org/

[49] Tigris.org Open Source Software Engineering Tools. (n.d.). Retrieved November 2009, from argouml.tigris.org: http://argouml.tigris.org/

[50] *Unified Modeling Language™*. (n.d.). Retrieved November 2009, from Object Management Group - UML: http://www.uml.org/

[51] Valtanen, A., Ahonen, J. J., & Savolainen, P. (2009). Improving the Product Documentation Process of a Small Software Company. *10th International Conference on Product-Focused Software Process Improvement, PROFES 2009*, (pp. 303-316). Oulu, Finland.

[52] Visconti, M., & Cook, C. R. (2004). Assessing the State of Software Documentation Practices. *5th International Conference on Product Focused*

Software Process Improvement, PROFES 2004 (pp. 485-496). Kansai Science City, Japan: Springer Berlin / Heidelberg.

[53] Wang, Z. (2005). *Characterising Architectural Concerns Using "Concern Traceability Maps"*. Master Thesis, University of Western Ontario, Department of Computer Science.

[54] Wheeler, J. (2009). *Overcoming the Software Developer Experience Gap in China.* Retrieved November 2009, from China software outsourcing and Chinese offshore development blog: http://www.daoofoutsourcing.com/software-developer-experience-in-china/

[55] Yin, R. K. (2003). *Case Study Research: Design and Methods* (3rd ed.). Sage Publications.

Appendix A: Survey Questionnaire

The first section of the survey consisted of six initial questions that were intended to determine the background of the respondents. The questions in the second section were meant to collect data for measuring the metrics for this study. This appendix focuses on questions of the second section. The complete questionnaire can be found at [45].

Question 7: How often are the following type(s) of project information discussed and explicitly documented (in any form or medium)?

	Discussed?					
Types of Information	Never	Rarely	Sometimes	Most of the time	Always	Not sure
Requirements						
Rationale, Priority, Source and Assumptions behind Requirements						
Architectural relevance of Requirements						
Relationship between Requirements						
Quality Attributes						
Quality Scenarios						
Use case scenarios						
Domain related information						
Issues						
Design decisions and rationale						
Architectural driver						
Tactics						
Patterns						
Other architectural artefacts						
Others (Please specify in the rows below)						

Types of Information	Documented?					
	Never	Rarely	Sometimes	Most of the time	Always	Not sure
Requirements						
Rationale, Priority, Source and Assumptions behind Requirements						
Architectural relevance of Requirements						
Relationship between Requirements						
Quality Attributes						
Quality Scenarios						
Use case scenarios						
Domain related information						
Issues						
Design decisions and rationale						
Architectural driver						
Tactics						
Patterns						
Other architectural artefacts						
Others (Please specify in the rows below)						

Question 8: How often do you use the following medium(s) for project discussion?

Discussion Mediums	Requirements discussion					
	Never	Rarely	Sometimes	Most of the time	Always	Not sure
Face to face discussion						
Telephone						
Survey						
Email						
Teleconference						

Text chat						
Wiki, blog, forum, etc.						
Others (Please specify in the rows below)						

	Architecture discussion					
Discussion Mediums	Never	Rarely	Sometimes	Most of the time	Always	Not sure
Face to face discussion						
Telephone						
Survey						
Email						
Teleconference						
Text chat						
Wiki, blog, forum, etc.						
Others (Please specify in the rows below)						

Question 9: For each of the following activities, how often do you find that the project documentation and related knowledge are not adequate and that you need to go back and check the context (talk to the stakeholders again, check old mail/chat/meeting-minutes/etc.) for additional information?

	Requirements document and related knowledge are not adequate?					
Project Activities	Never	Rarely	Sometimes	Most of the time	Always	Not sure
Requirements Engineering						

Software Architecting						
Design & Coding						
Testing						
System Integration						
Maintenance						
Project Management						
Quality Control and/or Assurance						
Process Improvement						
Others (Please specify in the rows below)						

	Architecture document and related knowledge are not adequate?					
Project Activities	Never	Rarely	Sometimes	Most of the time	Always	Not sure
Requirements Engineering						
Software Architecting						
Design & Coding						
Testing						
System Integration						
Maintenance						
Project Management						
Quality Control and/or Assurance						
Process Improvement						
Others (Please specify in the rows below)						

Appendix B: Condensed Survey Results

B 1. Types of Information discussed:

Types of Information	Never	Rarely	Sometimes	Most of the time	Always	Not sure
Requirements	0	0	4	8	20	0
Rationale, Priority, Source and Assumptions behind Requirements	0	5	5	12	10	0
Architectural relevance of Requirements	1	8	10	6	7	0
Relationship between Requirements	1	4	7	12	8	0
Quality Attributes	1	6	11	7	6	1
Quality Scenarios	5	5	7	8	6	1
Use case scenarios	3	7	10	5	6	1
Domain related information	0	7	8	10	6	1
Issues	1	0	7	13	9	2
Design decisions and rationale	1	2	7	15	7	0
Architectural driver	3	10	8	6	5	0
Tactics	5	7	9	9	1	1
Patterns	5	11	9	3	3	1
Other architectural artefacts	8	7	4	3	2	8

B 2. Types of Information documented

Types of Information	Never	Rarely	Sometimes	Most of the time	Always	Not sure
Requirements	0	4	4	8	14	2
Rationale, Priority, Source and Assumptions behind Requirements	3	7	7	5	7	3
Architectural relevance of Requirements	4	9	6	6	3	4
Relationship between Requirements	3	7	10	1	8	3
Quality Attributes	4	5	10	3	4	6
Quality Scenarios	8	4	10	4	3	3
Use case scenarios	7	5	5	6	5	4
Domain related information	5	5	10	1	6	5
Issues	3	7	6	7	6	3
Design decisions and rationale	3	5	12	6	3	3
Architectural driver	3	10	8	6	5	0

Tactics	7	11	6	3	1	4
Patterns	7	15	3	2	2	3
Other architectural artefacts	8	8	3	2	2	9

B 3. Mediums used for RE discussion

Discussion Mediums	Never	Rarely	Sometimes	Most of the time	Always	Not sure
Face to face discussion	1	0	10	10	11	0
Telephone	3	8	13	3	5	0
Survey	19	6	5	1	1	0
Email	1	2	10	10	9	0
Teleconference	3	5	15	5	4	0
Text chat	3	4	11	9	5	0
Wiki, blog, forum, etc.	9	5	11	3	4	0

B 4. Mediums used for SA discussion

Discussion Mediums	Never	Rarely	Sometimes	Most of the time	Always	Not sure
Face to face discussion	4	4	9	8	5	2
Telephone	9	9	8	2	3	1
Survey	20	5	4	0	1	2
Email	5	5	10	2	9	1
Teleconference	7	7	9	5	2	2
Text chat	9	4	10	2	6	1
Wiki, blog, forum, etc.	13	5	10	0	3	1

B 5. Impact of information lost during RE

Project Activities	Never	Rarely	Sometimes	Most of the time	Always	Not sure
Requirements Engineering	0	4	13	8	2	5
Software Architecting	2	5	6	9	7	3
Design & Coding	2	9	7	8	2	4
Testing	1	10	10	4	2	5
Maintenance	1	8	13	5	0	5

Project Management	7	7	6	5	1	6
Quality Control and/or Assurance	3	6	9	5	2	7
Process Improvement	2	6	7	7	2	8
System Integration	1	11	8	6	0	6

B 6. Impact of information lost during SA

Project Activities	Never	Rarely	Sometimes	Most of the time	Always	Not sure
Requirements Engineering	2	7	6	6	1	10
Software Architecting	3	5	8	3	2	11
Design & Coding	1	5	3	6	7	10
Testing	1	7	8	5	1	10
Maintenance	2	7	9	2	1	11
Project Management	1	12	4	4	0	11
Quality Control and/or Assurance	1	6	6	6	1	12
Process Improvement	1	6	7	5	1	12
System Integration	2	4	4	6	6	10