Requirements Engineering Process Improvement in Health IT Projects

Master’s Thesis

By

ANUPAMA CHITTI

STUDENT ID: 1369958

CHITTA 04 @ WAIRAKA.COM
Fill the brain with high thoughts, highest ideals, place them day and night before you, and out of that will come great work.

Swami Vivekananda
Abstract

Research has proved that requirements engineering in e-health projects is a challenge. Many of the e-health software end-users do not believe in investment in this process and e-health vendors find it challenging to deliver what client wants without having defined requirements accurately.

Healthcare is an industry where patient information is very crucial not just for the patient but also for healthcare providers. Appropriate health related information is one of the keys to quality care of the patient. This has been the reason for computerising business processes in healthcare.

Given that many privacy models can be applied to protect patient’s health information and complex business models and processes in healthcare, designing and developing such IT processes needs to be meticulously analysed and implemented.

This thesis deliberates on requirements engineering effort required for e-health vendors in delivering complex IT projects for e-health.

The aim of this thesis is to study approaches taken by software engineers in an e-health vendor company in accomplishing such complex projects. The approach of literature review, survey and interviews has been adopted to investigate the requirements engineering effort carried out in such projects.

An optimised solution has been presented with the help of quantitative and qualitative analysis of the collected data, conglomeration of different approaches defined by contemporary researchers and usage of Microsoft Visual Studio Team Foundation Server 2012 to improve the requirement engineering process in the e-health vendor organisation.
I am grateful to my advisor and initial primary supervisor Dr. Guan Yue Hong for her initial advice and support during this study and research. I am thankful for her patience and the time she has spared to bring me up to speed with the understanding of software engineering subject matter.

I am grateful to my current primary supervisor Dr. Chandimal Jayawardena for his timely guidance in health informatics and completion of this thesis.

I am also grateful to my associate supervisor Dr. Xiaosong Li for her continuous support and guidance in the different aspects of the subject and doing the thesis document.

I would like to thank my current employer and my managers who have been flexible to accommodate my study requirements.

I am thankful to my current employer who has consented for me to carry out the survey and interviews for this study in the organization.

I am very thankful to all my ever supportive respondents who have shared their experiences and knowledge in various e-health projects in their responses. This thesis would not have been possible without their timely help, support and guidance.

I would like to thank all my friends and my children’s care giver for the continuous support they have extended. Thanks to all of them for the countless number of times they have offered to take care of my children and help me complete this thesis.

I would like to thank my mother and my brothers who have continuously supported me throughout my Masters program.

Last but not the least; I would like to thank my lovely children and my husband for their continuous support for over the last couple of years and the encouragement they have given over this period. I am ever grateful to my husband who encouraged me to start studying again without whose support it would have never been possible for me to start and complete this study.
# Contents

Contents .......................................................................................................................... 5  

Chapter 1 ........................................................................................................................... 1  
1. Introduction .................................................................................................................. 1  
   1.1 The Stimulus ........................................................................................................... 1  
   1.2 Problem Statements ............................................................................................. 2  
   1.3 Main Contribution ................................................................................................. 4  
   1.4 Theme of the thesis ............................................................................................... 4  
   1.5 Research Methodology ......................................................................................... 5  
   1.6 The context of the thesis ....................................................................................... 6  
   1.7 Thesis Structure .................................................................................................... 6  

Chapter 2 .......................................................................................................................... 8  
2. Literature Review ........................................................................................................ 8  
   2.1 ICT in Healthcare ................................................................................................... 8  
   2.2 E-health in New Zealand ....................................................................................... 10  
   2.3 Software Development Life Cycle models ............................................................ 14  
   2.4 Requirements Engineering (RE) and its presence in e-health ................................ 28  
   2.5 Hypothesis ........................................................................................................... 56  

Chapter 3 .......................................................................................................................... 58  
3. Survey and interview Results ...................................................................................... 58  
   3.1 Introduction ........................................................................................................... 58  
   3.2 Data Collection and Outcomes .............................................................................. 59  
   3.3 Summary ................................................................................................................ 85  

Chapter 4 .......................................................................................................................... 86  
4. Data Analysis and Discussion ...................................................................................... 86  
   4.1 Introduction ........................................................................................................... 86  
   4.2 Integrated Analysis (of the survey and interviews) ............................................... 86  
   4.3 My observation from data collected in the interviews .......................................... 101  
   4.4 The synthesized methodology .............................................................................. 104
4.5 Summary ........................................................................................................... 125

Chapter 5 ............................................................................................................. 127

5. Conclusion ....................................................................................................... 127

5.1 The research ................................................................................................. 127

5.3 Future Research Direction .......................................................................... 127

5.4 Limitations of this research ......................................................................... 128

References .......................................................................................................... 129

Appendix 1 ........................................................................................................... i

Participant consent form ....................................................................................... i

Information sheet .................................................................................................. ii

Privacy statement ................................................................................................ iii
# Table of Figures

Figure 1: Research method followed ................................................................. 6  
Figure 2: Integrated Healthcare Model of IT Health Board (Ministry of Health, 2012) .......... 11  
Figure 3: The Waterfall Model of Software Development (Royce, 1987) .......................... 16  
Figure 4: Incremental software Development ..................................................... 18  
Figure 5: Evolution of software development standards in military domain (Codur & Dogru, 2012). 20  
Figure 6: Reuse Oriented Software Engineering Model (Sommerville, 2011) ..................... 22  
Figure 7: XP Process Life Cycle (Ramyakrishna, Phanikanth., Phanikrishna & Vamsikrishna, 2011) 26  
Figure 8: Scrum Process (Schwaber, 2011) ....................................................... 27  
Figure 9: Requirements Elicitation Process based on BABOK definitions ..................... 36  
Figure 10: Requirements Analysis Process (Blaise, 2012) ....................................... 38  
Figure 11: Domain Diagram Model (Blaise, 2012) ............................................... 39  
Figure 12: User Stories in Agile Life Cycle (Shafer, 2010) ...................................... 44  
Figure 13: Peer review process (Chemuturi, 2012) ............................................. 46  
Figure 14: Verification vs. Validation (Samaras & Horst, 2005) ............................... 47  
Figure 15: Pre and Post Traceability of requirements (Gandhi, Lee & Park, 2010) .......... 49  
Figure 16: Respondents’ roles in e-Health Projects .......................................... 60  
Figure 17: Requirements documents the respondents refer to .................................. 61  
Figure 18: Requirements documents - input for project planning .............................. 62  
Figure 19: Aspects in e-health requirements documents ......................................... 63  
Figure 20: Level of importance to confirmed requirements ..................................... 64  
Figure 21: Percentage of confirmed requirements before development starts .......... 65  
Figure 22: Practicing structured requirements engineering results in better planning ...... 66  
Figure 23: Quality of the software and RE effort .................................................. 67  
Figure 24: RE process leads to better delivery management .................................... 68  
Figure 25: Negative consequences of not having RE Process ................................ 69  
Figure 26: Rework because of unclear requirements .............................................. 70  
Figure 27: Requirements must be prioritised ....................................................... 71  
Figure 28: Who prioritises the requirements ....................................................... 72  
Figure 29: Prioritisation of requirements and quality of delivery management of e-health projects .... 73  
Figure 30: Factors influencing requirements prioritisation ..................................... 74  
Figure 31: SDLC in e-health projects ................................................................... 75  
Figure 32: Rework after requirements are developed and deployed ......................... 76  
Figure 33: impact analysis on e-health projects .................................................... 77  
Figure 34: Factors to be considered when analysing the impact of rework .................. 78
Tables

Table 1: Respondents’ roles in e-Health Projects ........................................................................60
Table 2: Requirements documents the respondents refer to .........................................................61
Table 3: Requirements documents - input for project planning ..................................................62
Table 4: Aspects in e-health requirements documents ................................................................63
Table 5: Level of importance to confirmed requirements .............................................................64
Table 6: Percentage of confirmed requirements before development starts ...............................65
Table 7: Practicing structured requirements engineering results in better planning ....................66
Table 8: Quality of the software and RE effort ...........................................................................67
Table 9: RE process leads to better delivery management ............................................................68
Table 10: Negative consequences of not having RE Process .......................................................69
Table 11: Rework because of unclear requirements ....................................................................70
Table 12: Requirements must be prioritised ..............................................................................71
Table 13: Who prioritises the requirements ..............................................................................72
Table 14: Prioritisation of requirements and quality of delivery management of e-health projects ....73
Table 15: Factors influencing requirements prioritisation .............................................................74
Table 16: SDLC in e-health projects ............................................................................................75
Table 17: Rework after requirements are developed and deployed .............................................76
Table 18: impact analysis on e-health projects ..........................................................................77
Table 19: Factors to be considered when analysing the impact of rework .................................78
Table 20: Does Customer spend enough time on requirements ................................................79
Table 21: Requirements notation techniques in e-health projects ..............................................81
Table 22: Requirements notation techniques in e-health projects – Other options ....................81
Table 23: RE effort in e-health projects .......................................................................................82
Table 24: Requirements management tools in e-health projects ................................................83
Table 25: Ratings of different RE aspects ....................................................................................85
## Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABC</td>
<td>Activity Based Computing</td>
</tr>
<tr>
<td>ACM</td>
<td>Association of Computer Machinery</td>
</tr>
<tr>
<td>BA</td>
<td>Business Analysis / Business Analyst</td>
</tr>
<tr>
<td>BAA</td>
<td>Business Analysis Approach</td>
</tr>
<tr>
<td>BABOK</td>
<td>Business Analysis Body Of Knowledge</td>
</tr>
<tr>
<td>BPML</td>
<td>Business Process Modelling Language</td>
</tr>
<tr>
<td>BRS</td>
<td>Business Requirements Specification</td>
</tr>
<tr>
<td>CDA</td>
<td>Clinical Documentation Architecture</td>
</tr>
<tr>
<td>CGS</td>
<td>Clinical Guidelines System</td>
</tr>
<tr>
<td>CIS</td>
<td>Clinical Information System</td>
</tr>
<tr>
<td>COTS</td>
<td>Commercially Off The Shelf</td>
</tr>
<tr>
<td>CPSA</td>
<td>Community Pharmacy Service Agreement</td>
</tr>
<tr>
<td>CRM</td>
<td>Customer Relationship Management</td>
</tr>
<tr>
<td>CSCW</td>
<td>Computer Support Co-operative Work</td>
</tr>
<tr>
<td>DBA</td>
<td>Database Administrator</td>
</tr>
<tr>
<td>DHB</td>
<td>District Health Board</td>
</tr>
<tr>
<td>ECG</td>
<td>Electrocardiogram</td>
</tr>
<tr>
<td>EHR</td>
<td>Electronic Health Record</td>
</tr>
<tr>
<td>EMR</td>
<td>Electronic Medical Record</td>
</tr>
<tr>
<td>eMR</td>
<td>e-Medicines Reconciliation</td>
</tr>
<tr>
<td>EPA</td>
<td>e-Prescribing and Administration</td>
</tr>
<tr>
<td>ERP</td>
<td>Enterprise Resource Planning</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>GOe</td>
<td>Global Observatory for e-Health</td>
</tr>
<tr>
<td>HCI</td>
<td>Human and Computer interaction</td>
</tr>
<tr>
<td>HCT</td>
<td>Home Care Technology</td>
</tr>
<tr>
<td>HINZ</td>
<td>Health Informatics New Zealand</td>
</tr>
<tr>
<td>HIS</td>
<td>Health Information System</td>
</tr>
<tr>
<td>HL7</td>
<td>Health Level Seven International</td>
</tr>
<tr>
<td>HPI</td>
<td>Health Practitioner Index</td>
</tr>
<tr>
<td>IBM</td>
<td>International Business Machines</td>
</tr>
<tr>
<td>ICT</td>
<td>Information and Communication Technology</td>
</tr>
<tr>
<td>ID</td>
<td>Identity</td>
</tr>
<tr>
<td>IEEE</td>
<td>Institute of Electrical and Electronics Engineers</td>
</tr>
<tr>
<td>IGP</td>
<td>Information Gathering Plan</td>
</tr>
<tr>
<td>IREB</td>
<td>International Requirements Engineering Board</td>
</tr>
<tr>
<td>IS</td>
<td>Information Systems</td>
</tr>
<tr>
<td>IT</td>
<td>Information Technology</td>
</tr>
<tr>
<td>LTC</td>
<td>Long Term Condition</td>
</tr>
<tr>
<td>MRI</td>
<td>Magnetic resonance imaging</td>
</tr>
<tr>
<td>MZULM</td>
<td>New Zealand Universal List of Medicines</td>
</tr>
<tr>
<td>NASA</td>
<td>National Aeronautics and Space Administration</td>
</tr>
<tr>
<td>NATO</td>
<td>North Atlantic Treaty Organization</td>
</tr>
<tr>
<td>NEHTA</td>
<td>National Electronic Health Transition Authority</td>
</tr>
<tr>
<td>NHI</td>
<td>National Health Index</td>
</tr>
<tr>
<td>NZ</td>
<td>New Zealand</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>NZePS</td>
<td>New Zealand e Prescribing Service</td>
</tr>
<tr>
<td>NZMF</td>
<td>New Zealand Medicines Formulary</td>
</tr>
<tr>
<td>PAS</td>
<td>Patient Administration Systems</td>
</tr>
<tr>
<td>PCEHR</td>
<td>Patient Centric Electronic Health Records</td>
</tr>
<tr>
<td>RE</td>
<td>Requirements Engineering</td>
</tr>
<tr>
<td>RFID</td>
<td>Radio Frequency Identifier</td>
</tr>
<tr>
<td>RFP</td>
<td>Request For Proposal</td>
</tr>
<tr>
<td>RMP</td>
<td>Requirements Management Plan</td>
</tr>
<tr>
<td>RMT</td>
<td>Requirements Management Tool</td>
</tr>
<tr>
<td>RT</td>
<td>Requirements Traceability</td>
</tr>
<tr>
<td>SAD</td>
<td>System Analysis and Design</td>
</tr>
<tr>
<td>SAP</td>
<td>SAP Enterprise Resource Planning software</td>
</tr>
<tr>
<td>SDLC</td>
<td>Software Development Life Cycle</td>
</tr>
<tr>
<td>SME</td>
<td>Subject Matter Expert</td>
</tr>
<tr>
<td>SOA</td>
<td>Service Oriented Architecture</td>
</tr>
<tr>
<td>SRS</td>
<td>System Requirements Specifications</td>
</tr>
<tr>
<td>SWEBOK</td>
<td>Software Engineering Book Of Knowledge</td>
</tr>
<tr>
<td>T &amp; M</td>
<td>Time and Money</td>
</tr>
<tr>
<td>TDD</td>
<td>Test Driven Development</td>
</tr>
<tr>
<td>TFS</td>
<td>Team Foundation Server</td>
</tr>
<tr>
<td>UAT</td>
<td>User Acceptance Test</td>
</tr>
<tr>
<td>UI</td>
<td>User Interface</td>
</tr>
<tr>
<td>UML</td>
<td>Unified Modelling Language</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>URS</td>
<td>User Requirements Specification</td>
</tr>
<tr>
<td>VAR</td>
<td>Value Added Resellers</td>
</tr>
<tr>
<td>WCF</td>
<td>Microsoft’s Windows Communication Foundation</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Mark-up Language</td>
</tr>
<tr>
<td>XP</td>
<td>Extreme Programming</td>
</tr>
</tbody>
</table>
1. Introduction

1.1 The Stimulus

Healthcare industry’s focus is to provide quality healthcare services to its consumers i.e., their patients. Information and communication technology is being used by healthcare industry to enhance the quality of services. This industry is dynamic in nature. The dynamics increases complexity with respect to technology, medical needs and organisational needs. Identifying those dynamic features and its interactions leads to definition, development and implementation of systems. Integration of computer and information technology with such systems in healthcare has given rise to a new area of study i.e. Health Informatics and e-Health.
The health informatics is associated with activities like communication, structuring, questioning, searching and making decisions with health information. The intervention of informatics in health helps reducing human errors in healthcare. Many information technology companies in NZ under the umbrella of health IT board today have built healthcare software commonly known as e-health software. These are many information systems where data is processed and used within and across the systems (Ministry of Health, 2012). This dynamics calls for better definition of models (prototypes) from which the information systems are developed. The understanding and engineering of requirements of such models has been attempted by many software companies.

The requirements for such systems are expected to be unambiguous. The requirements should cater to not just a group of people but all the stakeholders (Holt, 2012). The requirements engineering framework includes feasibility study of the project, requirements elicitation, gathering, analysing and specifying with respect to different stakeholders; classifying them as functional, non-functional, user, technical, operational and transactional requirements; interpreting and documenting; modelling and in the end designing the requirements.

If e-health softwares are not as per the requirements of its users, this may lead to severe damages to the patients’ health, healthcare providers’ career, healthcare organisations, healthcare governance, e-health software providers and wide range of other stakeholders in the health and IT industry. As mentioned by Ebert (2010) in his work, many surveys and researches since 1970 have suggested directly proportional relationship between the Requirements Engineering (RE) and project success. Insufficient RE of both the development and change management projects is one of the top standing factors contributing to the project failures. RE is a discipline in software engineering is the initial factor that determines the success or failure of the project or the product (Ebert, 2010).

This thesis is an attempt to study and deliberate on the RE practices that software development teams undertake in e-health projects.

1.2 Problem Statements

In today’s e-health scenarios, the notion of completeness of health information systems and computer, communication and information technology is questioned by few healthcare professionals. There have been many implementations of IT software in the healthcare sector. However, in every project, there is some change of requirements that finally directs to unfinished/incomplete requirements. Heeks (2006) argues that this is because of the difference between the current functionality in the software/ business process / information processing to the required functionality. The required functionality is perceived by differently by different stakeholders. The difference is reflected in two terms - an intended design and a reality. The gap between the design and the reality (required) can be seen differently by different user groups or stakeholders in the
- Information processes like data storage and data flows
- Technology being used including hardware, software, the networks and difference automated gadgets
- The process of communication and sequence of activities
- The objectives with which different groups behave
- The different qualitative and quantitative competencies required by the staff using the software
- The management systems associated with the requirements
- Other project dimensions like time and money

As studied by Pagliari (2007), the design methods applied by medical and software engineering teams are tremendously linked and can be jointly applied. This calls for appropriately investing the time and effort in the initial stages of requirements analysis and designing and thus the engineering of requirements.

In any e-health software the information processing is at all times patient health centric or patient information centric. Security and privacy of patient information are the other factors that require attention. All these factors call for not just analysing the requirements but also communicating them rigorously between the development team and user team during the project.

When there are changes in requirements, the need of effective communication is required all the more which leads to a research direction of *Identifying the different approaches to enhance the RE effort to accommodate the dynamically changing requirements.*

In many projects, there are many requirements for different users. Each user has a different perception of the priority of the requirements. However, the sufficient correctness of the software is required to be identified and optimised. Shaw (2002) argues that the sufficient correctness is

> “the degree to which a system must be dependable in order to serve the purpose its user intends, and to do so well enough to satisfy the current need and expectation of the users.”

Cheng and Atlee (2007) suggest that if software is in sufficient correctness, then it is optimised for the current requirements specified and so is tolerable. This concept leads to further research to tackle many requirements defined by many users. But within the scope of a project (time, money and resources) these requirements should prioritised and documented. This thorough study has led to the identification of different attributes of RE and has led to my first research question:

1. **Does attributes like identifying gaps, prioritisation and documentation of requirements enhance the RE effort on managing the changes in requirements in e-health projects?**
Cheng and Atlee’s study has raised some questions about how to determine the healthiness / un-healthiness of software with reference to requirements (features) in a highly complex and dynamic healthcare environments.

An investigation in this direction has further clarified that the impact of change and communication of requirements are the other aspects of requirements management which are very important to help to determine the healthiness of the software. The other research questions that have been identified are:

2. How is the impact of change in requirements analysed and used in the delivery of project
3. How can the requirements management effort be enhanced through effective communication.

This thesis is an attempt to answer these questions.

1.3 Main Contribution

In the process of finding answers to the above mentioned questions, the journey of this study has led to ample knowledge gained. The process of research followed has helped me to draw hypothesis from the literature review and the analysis of collected data to verify the hypothesis. There were some major observations that have further helped me to design a requirements engineering process specific to small and medium size e-health vendor organisations.

An attempt has been made to create a model by conglomerating two different approaches - one for designing a requirements road map and the other about different factors that can lead to successful requirements planning. This study is further synergised by adding the feasibility of managing the requirements and achieving traceability in agile projects using Microsoft Visual Studio Team Foundation Server 2012 with Microsoft Test Manager which is the main contribution presented in this thesis.

1.4 Theme of the thesis

RE is within itself a very vast section of the software engineering field. The challenges identified in the software development of e-health projects with regards to RE and the quest for the answers have been outlined to the software development life cycles(SDLC) and relevant RE practices that are currently applied for these e-health projects. Within each SDLC applied in e-health projects, the RE process with reference to the documentation, prioritization, change management, impact analysis of changes, the delivery management and the use of required / relevant RE techniques have been deliberated in this thesis.
1.5 Research Methodology

An insight into RE through study and day to day experiences in SDLCs for e-health projects in the IT industry has prompted an epistemological approach to this study of RE. The research method proposed in the thesis is:

- My knowledge of study and relevant experience in the IT industry has been used to identify the area of interest and appreciate the different challenges currently existing in this area of RE.
- A further insight has been gained by reading different publications like journals, magazines, blogs by relevant experienced people in the industry, research papers and white papers from the others in the industry has clarified thoughts and has helped me to identify the research problem and design the appropriate research questions.
- A contemporary insight has been gained thorough intensive literature review. The sources for this has been books, peer reviewed journal articles, conference proceedings, technical reports, website publications, blogs of experience professionals in the industry. A thorough study has led to drawing a few hypothesis
- An e-health software development company was approached for conducting surveys and interviews.
- After gaining the consent to conduct the survey and interviews of the staff of the software company, the survey questionnaire was accordingly designed. The questions for the survey were all close ended multiple choices. These questions consisted of leading, importance, likert, dichotomous, bipolar and rating scale questions of different aspects of RE. All these questions can be seen in Chapter 3 of this thesis.
- A quantitative analysis of the results was done and then followed by a few questions in the interviews. The purpose of this is process was get an ethnographic understanding of the RE process carried out in the software company under observation. Both the results were analysed to get to an understanding of how different aspects of RE are handled in the software company. This analysis has helped in proving the hypothesis and outlining some observations in the study.
- These observations have been used to define my perception of a possible solution in the conclusion chapter. Further scope of research and limitations of the current research have also been identified.

This research method has been adapted as this was best suited for the kind of study that has been carried out. However, there are some limitations of this method.
• The survey has been conducted on a sample of only 50 people. This is because the size of the organisation.

• The study has been limited to only one software development company and does not necessarily represent the complete industry. However, as this is an e-health software vendor company, its relevance to the study cannot be challenged.

Below is the pictorial representation of the research method followed:

![Research Method Diagram]

**Figure 1: Research method followed**

### 1.6 The context of the thesis

I have been working on different e-health projects in New Zealand for the past 7 years. During the development and implementation of these projects, I have observed that their requirements have changed at every stage of the development life cycle. I have also observed that RE process were ambiguous at some stage of the project and ended up in having requirements changed because of missing requirements, requirements gaps, additional functionality required etc. This has motivated me to investigate and analyse the reasons behind such scenarios in this thesis.

This thesis has been submitted as a partial fulfilment of the requirement for Masters of Computing.

### 1.7 Thesis Structure

The thesis is structured in the following manner:

**Chapter 2: Literature Review:**
It has a thorough consideration of the information and communication technology in healthcare globally and in New Zealand. A critical analysis of the different software development life cycles (SDLCs) prevalent in the e-health software development industry as of today is presented followed by the different RE practices within different SDLCs. The different strengths, weaknesses, advantages and disadvantages of all these approaches have been deliberated. This chapter’s purpose in the thesis has been to identify and draw the hypotheses out of this literature review for further investigation in this research process.

**Chapter 3: Survey and Interview Results:**

This section of the thesis contains the different questions asked in the survey and interviews conducted as a part of the study and the results.

**Chapter 4: Data Analysis**

The analysis of the results from these survey and interviews has been deliberated in this section. This chapter also captures my view points of the data collected and different findings of the research and presents my proposed approach to the solution.

**Chapter 5: Conclusion**

This chapter presents my approach to the solution and the possible future research directions from this research. The limitations of the conducted research have also been identified in this section of the thesis.

The different references and the *appendix* containing the information sheets for the participants, consent forms etc. have also been attached at the end of the document.
Chapter 2

2. Literature Review

This chapter is the knowledge base on which this thesis is written. The review of literature starts with an introduction to the ICT in healthcare followed by its presence in the New Zealand e-Health industry. Then, an intensive understanding of different software development life cycle models is presented and is followed by rigorous appreciation of respective RE processes. Further, investigation has been undertaken to justify how RE is different in e-health software development projects. By the end of this chapter, a few hypotheses have been identified.

2.1 ICT in Healthcare

Healthcare is one of the most important aspects of mankind that helps to keep the race existing on earth. Healthcare industry is one of the most information critical and process intense industries where the mankind’s wellbeing is at stake. Increasing costs, complexity of the business processes within healthcare sector, incompetent management of such business processing leading to variability in quality of care, limitations to physical access to the patient records etc have been some of the intrinsic characteristics of healthcare industry in the past. Because of this nature of criticality, the industry has been integrating with information intense advancements (Popescu, 2010). Information technologists have been working towards helping the healthcare professionals to access relevant critical patient information timely, accurately, completely and reliably (Wager, Lee & Glaser 2009).

The IT contributions (to name a few) have been towards

All differences in this world are of degree, and not of kind, because oneness is the secret of everything.

Swami Vivekananda
• Digitising patient medical results like different reports, ECGs, MRIs and other records which have been hand written in the past
• Helping the medical staff in the surgical processes
• Communications between different tasks of the patient care ensuring better care
• Starting from the patient entry to the hospital, all administrative tasks go through the clinical/health information systems and are centrally viewable to the administrative, Para medical, medical staff which can further flow through the care givers.
• E-prescribing and tracking the medication given to the patients
• Telemedicine
• Mobile health
• EMRs, EHRs, PCEHRs
• Mobile electronic devices used for diagnoses, medical tests etc
• Decision support systems facilitating consulting for the healthcare professions
• E-learning and online tools for the healthcare professional(students, nurses, doctors, caregivers etc)(Gerber, Olazabal, Brown & Pablos-Mendez, 2010)

This integration of IT with healthcare has resulted in healthcare informatics tools which have increased efficiencies of the clinical and Para-medical staff reduces medical blunders, reduced paper work and increased quality of care and affordable medical care. This has paved way for more research on the requirements of e-health models and platforms and preventive care.

With the complex nature of the business processes with e-health applications and the data processing requirements, there has been an emergence of a market for designing clinical information systems like computerised physician order entry, simulation systems for educational purposes, medication administration, electronic medical records, telemedicine, tele-health, electronic health records, decision support systems, healthcare accounting information systems, healthcare administrative and financial information and management systems. These include infrastructure requirements implementation like the barcode usage, RFIDs, networking systems for electronic communication of data (Wager, Lee & Glaser, 2009)

This application of information technology in healthcare industry has catered to information needs of a wide range of stakeholders like healthcare professionals, researchers, healthcare consumers, infrastructure architects/developers, ministry of health, healthcare funding organisations, healthcare IT professionals etc (Kerr, Cullen, Duk, Holt, Kirk, Komisarczuk, Warren & Wilson, 2006). This requirement has further lead to the developing interoperability between different information systems, business processes and workflows. Thus, Analysing, Designing, Estimating, Engineering and Developing on the appropriate requirements is deemed to be the need of the hour.
Using Information Technology in health sector is now termed as ‘e-Health’ in the market. Today, it is one of the most promising aspects of healthcare. The World Health Organisation has initiated the GOe to study the growth of e-Health and its impact on the health industry in different nations. Not just the developed nations but the developing and the under developed nations also have open to this concept (World Health Organisation, 2013).

2.2 E-health in New Zealand

Healthcare in NZ is publicly funded through taxes earned by the government of NZ. The healthcare is provided by the DHBs. The central focus of which is the community. There has not been any strategic footing to develop the HIS in NZ. However, about 21 DHBs, since 2001 have been working towards incorporating clinical information systems that can facilitate information sharing through dynamic web-based / network-based technologies. The purpose has been to manage the patient information across different interoperable systems effectively and efficiently (Kerr, 2004).

The Health Information Strategy was introduced by the Ministry of Health in 2005 and was directed towards implementation of ICT in healthcare. The existence of legacy systems within each DHB and the need to maintain and interoperate between these systems has opened opportunities for many software vendors in NZ. This calls for the software vendors and the healthcare provider organisations (DHBs) to have cross discipline skills that can help them to understand the business and the priorities of the health industry better, communicate and work towards better products that can yield to better service and better understanding of the data structures within the industry.

Since then, there have been more number of stakeholders in the health informatics in NZ like the healthcare providers, consumers, the non-government organisations of healthcare providers, the ministry of health, HINZ, ancillary care providers (like the social workers), educational institutions, health IT companies, the Health IT Cluster, the IT Health Board etc. (Kerr, Cullen, Duk, Holt, Kirk, Komisarczuk, Warren & Wilson, 2006)

HINZ is a non-profit organisation with focus facilitating all round improvements in the health sector to provide better patient care and service which is achieved through the implementation of information technologies in the industry (HINZ, 2013).

NZ Health IT Cluster is a dynamic alliance of all software solution providers, consultants, IT professionals, policy makers, funders, and educational institutions etc that have the collaborated vision towards e-Health (New Zealand Health IT, 2013)

The IT Health Board plays a leadership role in conglomerating all these different stakeholders to bring a synergised effect in the e-Health industry. It is a sub set of the National Health Board which ensures
that appropriate IT implementation is done in the health sector. The vision of the Health IT Board is to facilitate an integrated healthcare model that

- can provide patient centric health information to the healthcare providers, care givers
- places clinicians as an integral part to ensure ongoing development and use of patient health information solutions
- assists the communities to access and use appropriate patient information that is electronically shared through authentic repositories

Below is the pictorial representation of the vision of the IT Health Board

![Figure 2: Integrated Healthcare Model of IT Health Board (Ministry of Health, 2012)](image)

The Board conducts conferences, holds together the health IT companies specialising in system integrations and consulting, communicates and works with the academic institutions. The e-projects initiated by the board, to name a few are e-Labs, e-Pharmacy, e-Prescribing. Microsoft collaborated e-health solutions (Ministry of Health, 2012).

The different programmes the board is currently undertaking as a priority till 2014 in NZ are

- **e-Medicines programme**: focusing on reducing the medication errors and the unfavourable drug administration errors by implementation of electronic systems that can aid secure and valuable use of the drugs. The goal is
o To reduce the damage caused to the patient because of inappropriate medication events
o To improve the patient care supported by appropriate medication information
o To manage and improve productivity by reducing the overheads caused because of medication error

This is being done through

a. Pharmacy Clinical Integration as NZePS and CPSA where the GPs and Pharmacies are brought under one information exchange
b. ePA where the focus is implementing the electronic prescribing and administration in hospitals
c. eMR where patient medication history is collected from different sources and reconciled for any differences before the administration plan.
d. NZ Universal List of Medicines (NZULM) and NZ Medicines Formulary (NZMF) are Decision Support Systems that helps the healthcare professionals with prescribing and point of care assistance

• National Solutions: for specialities like cardiac health, oncology, renal services etc, the governance is required for all purposes. This is done through:
  a. Two cardiac health projects ANZACS-Q1registry and cardiothoracic surgery project
  b. Common data business process, data description and messaging standards within information systems that provide oncology information throughout the nation
  c. interRAI project where the goal is to provide wide-ranging clinical support to the aged
  d. A common supply chain and finance system where the standards are defined and followed across the nation the different functions in the healthcare industry
  e. NHI and HPI systems that identifies the patient and healthcare provider and enables communicating between all existing healthcare systems.

• Regional Information Platforms within the DHBs where patient information is shared across different clinical information systems across different DHBs. The current implementations that are of priority till 2014 are
  a. eReferrals and eDischarges a system where the patient referrals can be made electronically and from GPs to DHBs, DHBs to DHBs, DHBs to care givers facilitating even patient flow and integrated service across the platform
  b. Clinical Data Repositories where the patient information and other medical support information is stored and access across the platforms
c. Imaging and Picture Archiving Communications Systems (PACS): the radiology, MRI, ultrasound and such other medical images are stored with standard and agreed formats so that they can be accessed across the platforms.
d.
Pharmacy and Laboratory clinical support: through a ePharmacy system throughout the region.
e. Patient Administration Systems (PAS) helping the medical and the paramedical staff in the hospital environments in patient administration activities like scheduling the appointments, tracking the patients, coding/billing, medication records within and across the DHBs.

- Clinical Integration programme: is an attempt to integrate the existing clinical practises like healthcare planning and shared care.
  a. Shared Care: Under this, the patient and the caregiver jointly develop healthcare plans.

  Currently, this is being envisioned through the innovation of Patient Portals were the patients will be able to view their health records online.

  Long Term Conditions is another such program where the co-ordination of healthcare is being managed by the community pharmacies and the care givers avoiding re-admissions for certain conditions where long term care is required (for example: diabetes). This has already been implemented in 3 DHBs in and around Auckland.

  Maternity Information System Programme where the complete maternity process starting from the first trimester of the pregnancy to the prenatal and post natal care across different care givers is managed and the sharing of the service and the required information is also facilitated. This information is shared across different stakeholders of the process i.e., GPs, midwives, doctors, nurses, hospitals. This new system is currently being implemented in Tairawhiti, Wanganui DHB and will be followed by Counties Manukau and Capital and Coast DHBs later in next year.

  Child Health Platform: the Midlands Health Network and the ministry of health are working towards delivering effective child health service where the information is shared between the paediatricians / hospitals / plunket nurses through the well child program.

b. Primary/Community Care: the ministry is working towards providing primary health information of the patient to be reachable to the emergency departments and such other primary care centres. This work is still at a very initial stage.
These initiatives have been worked upon since almost a decade. However, this is only the partial implementation and a national rollout of many programmes is still due and would require immense customisation depending upon the region where it is being implemented and the legacy information systems they are working on.

The purpose of all these initiatives is to make healthcare information accessible not just to the providers but also to the consumers. This has paved ways for many software vendors to specialise in delivering the required IT solutions to these initiatives.

Today there are many software applications assisting the healthcare practitioners like GP2GP for clinicians, Healthsoft, Toniq for pharmacists, world’s first implementation of voice to text dictation system for clinicians accessible over the internet, world’s first practice management system for ophthalmologist that can interface with 3rd party ophthalmology software, patient information systems for mobile dental clinicians accessible on wireless networks, breast screening program tacking system, clinical and centralised disease management projects, electronic whiteboard for ER patient Tracking etc are being implemented in NZ. However, this accounts for only 50% of web portal implementations and about 60% of EMR healthcare systems in the hospitals in New Zealand. There are still lot more business process to be automated within the healthcare industry in New Zealand (Ministry of Health, 2012).

2.3 Software Development Life Cycle models

In the initial stages of computing era, the engineering outlook and focus was only on the hardware aspect of computing. Only when the programmers started facing the difficulties in designing, recounting, ciphering and delivering complex systems did they appreciate the requirement of a systematic and structured approach to inscribe programs and thus evolved the engineering discipline for software development. The software engineering arena is built on the concept of developing programs (a set of lines of code that can be logically co-ordinated to complete a task) developed with a structured approach that can be executed by a computer.

According to Grier (2010), in 1966, the term software engineering started to stabilise in the computer industry. The science committee of the North Atlantic Treaty Organisation (NATO) organised the first conference in 1968 and since then there have been many deliberations given to several approaches. Since then the fundamental concepts of the software engineering discipline have been revolving around the systems specification, design, programming, debugging and maintenance. A formal code of ethics SWEBOK for this discipline has been produced and Association of Computer Machinery (ACM) and Computer Society of Institute of Electrical and Electronic Engineers (IEEE-CS) have defined the standards for software engineering.
Sommerville (2011) argues that today software engineering is a conglomeration of computer science and systems engineering disciplines. The fundamental process activities now revolve around software specification, development, validation and evolution (where the software changes to the customers’ and market’s requirements). The primary notions about Software Engineering is to manage software processes, dependability, security, RE and software reusability. Thus today, Software Engineering is being argued to be a systematic approach to apply these fundamental ideas in creating software which cater to a diverse varieties of applications like the stand alone applications, transaction based interactive applications, embedded control applications, batch processing applications, gaming and entertainment applications, data collection applications and systems that co-ordinate, inter-operate and control these applications or in other words systems of systems.

In healthcare industry, there is a lot of requirement for such systems and the interactions between them. These systems are used by healthcare providers, biotechnologists, care givers, administrators etc who are all aiming at providing better service to the patients and the healthcare industry. Applications like medical equipment supply management systems, medical records information systems, computerised physician order entry systems, clinical point of care systems, medical imaging systems, clinical decision support systems, artificial intelligence implemented systems like heart beat analysis system, diagnostic systems, radiology information systems, Health information message exchanging systems, mobile and tele-health systems are a few examples that are currently being used in the e-health industry today.

Weston (2005) advocates in his work that effective bioinformatics applications can be delivered through incorporating good listening activities in software engineering process.

- Listen to what the end user wants
- Listen to what the users are not saying i.e., understand what is being assumed
- Listen to what the user is saying or not saying when the end product is shown
- Listen to inner voices about the end product (your intuition)

These lead to the fact that like in any other projects, even in e-health projects, the understanding of the requirements of solution is the starting point in the software development.

There are certain activities involved in the software process and stream lining such activities in different ways has led to the deliberation of different software process models. The basic activities in any process model are

1. **RE** where the requirements are elicited, specified, documented, communicated and agreed upon with the customers/end users
2. **Designing and Implementing** where the requirements are transformed into executable software systems
Requirements Engineering Process Improvement in Health IT Projects

3. **Validating** if the built software is as per the requirements of the user
4. **Software Evolution** where the changes to the requirements (end-user requirements, technology enhancements etc) are incorporated.

Exclusive models were defined and practised in the 1950s and the 60s. These definitions were based on the rational approaches in handling the large scale software projects. Scacchi (2001) has classified them as the *traditional models*.

### 2.3.1 Traditional Models

The identified 4 models were:

1. **Waterfall Model**: where all the activities of the project are defined and executed in the linear sequential process. The activities of this model are as shown in the figure shown below.

![Waterfall Model of Software Development](image)

*Figure 3: The Waterfall Model of Software Development (Royce, 1987)*

The important **features** of this model include:

- The outcome of each step will feed in as an input to the next step.
- None of the stages overlap each other
- The model starts with establishing the feasibility of the project from where the requirements are drawn and agreed upon with the client.

The **advantages** of this vintage model have been identified by many as follows:
• It is suitable for well defined projects
• It can accommodate weak teams
• Documentation specific at each stage of the project
• Clearly defined deliverables at each stage
• Structured enforcement of activities like understanding before defining, defining before designing, designing before coding, coding before testing and testing before implementing
• Perceiving and implementing the model is not complicated
• Widely know

The disadvantages of using such a model can be

• It is not best suited for all/many projects
• It does not accommodate changes
• A clearly defined requirement is never a real-time scenario. So there is always a change aspect that needs to be handled.
• Errors in the requirements cannot be identified early there by always prone to the risk
• Mitigating risks/uncertainties is not practically possible in the model
• Operational overheads are high for error rectification (Munassar & Govardhan, 2010)

This classic model has been addressed as Pure Waterfall model because of the strengths like

• Not much input is required for planning, thereby reducing planning costs
• Structural implementation of the process minimises wasted effort

However, the weaknesses of such models because of it is being considered as obsolete are

• All processes are inflexible. So the scope of managing change will be very less
• Addressing any flaws in the project will be a challenge

This model is well suited for projects whose requirements are well understood with clearly defined architecture, technology, tools and infrastructure. However, this model fails to handle when changes are to be incorporated.

This model was then introduced with Stepwise Refinement. This model comprises the phases as the pure waterfall model. However, the different phases are not discontinuous. This creates the feasibility for the phases to overlap each other. This strengthens the whole model by incorporating

• More flexibility
Requirements Engineering Process Improvement in Health IT Projects

- Because of continuity in phases, the documentation effort is reduced
- Because of the overlapping nature between the phases, the one that are clearly defined and less ambiguous need not wait till the sequential implementation of other components

However, the weakness are identified as

- Because of the overlapping nature of the phases, there is ambiguity of the milestones and the deliverables
- As the activities can be parallelly performed, there can be some dependencies that can possibly be over looked
- A huge scope for miscommunication when activities are parallelly performed

2. **Incremental Development Model**: the software is developed in increments where the initial goal of the project in the model is to give the basic functionality and then providing improvements/enhancements and more functionalities at regular intervals or iterations. This model also supports regular maintenance of the software through support contracts. This fundamental approach of this model differentiates it from the classic waterfall model. This is the elementary feature of the agile approaches. Prototyping technique supports incremental development and iterative releases. The feedback from the user in each iterative release is fed back into the software at mutually agreed next release. This technique is most suited for large software systems and is very popular with organisations like NASA and IBM (Scacchi, 2001). This process can be either plan driven or agile. In the plan driven approach, the increments are planned in advance and in agile, they are identified but developed once the customer approved it for the iteration. Below is the representation of the process.

![Figure 4: Incremental software Development](image)

Figure 4: Incremental software Development
The SWOT analysis of this model points out the strengths and opportunities associated with this model are that the process:

- Accommodates the changes in requirements at reduced costs as the cost of analysis, design and documentation is way less than that of change in the waterfall model
- Accommodates customer feedback
- Ability for the users to give feedback as they can see the software
- The users can give better feedback from what they see on the software than on what they see in the design documents
- Continuous quality control through the process control and feedback from the customers/users
- Ensures intellectual control and system integration at every iteration (Trammell, Pleszkoch, Linger & Hevner, 1996)
- Faster delivery and deployment feasibilities ensure earlier access to the basic features of the software than as seen in the waterfall model

The weakness and threats of this model are:

- Even if the users and developers know the deliverables, the process deliverables may not be visible to the managers as the process is not documentation specific
- The system architecture may not be foolproof as the future deliverables (and its associated requirements) are never in sight. This could possibly degrade the system in terms of quality

The advantages which have enabled the model to be used widely are:

- The earlier versions of the releases always can be used as prototypes by the users and developers
- The waiting time for the users to use the system is less as the basic required functionality is delivered first and increments as add-ons
- Scope for change is always accommodated
- Important functionalities are identified and prioritised as initial releases so, the analysis, design and testing is to the maximum on the important features

However, the disadvantages of such a model are:

- Most systems requirements are different for different components that are used by different users. So, even if delivered, all users may not be able to use the systems immediately
- If replacement products are being used, then iterative development/deployment cannot be compatible for such requirements
The system designers will not be able to design the complete solution as this model incorporates a lot of changes. The system model incorporates a lot of changes. The system specifications can never be complete in such scenarios. As the technical specification totally relies on the system specification, this can lead to an incomplete technical specification and design (Sommerville, 2011).

3. Military Standards Models: Military domain uses a wide range of software applications. However, these applications are designer pieces and the only consumers/users are the armed forces. These applications are to work in the closed and controlled domain. These applications are a part of many mission critical systems like the missiles, submarines, rockets, fighter planes, aircrafts and many such other order and guarded systems. The regulatory authorities tend to have full control over such projects. Further, these applications average utilisation period is expected to be long. So, all software projects in the armed forces are planned and executed and are followed by the standards defined for them. NATO recognised the need of standards in software engineering of military projects in 1968. The first standard i.e. MIL-STD-1679 was defined in 1978. The timeline below specifies the evolution of the standards for software engineering processes over the last 4 decades.

![Figure 5: Evolution of software development standards in military domain (Codur & Dogru, 2012)](image-url)

IEEE defined the introductory standards in 1998. The latest being IEEE12207-2008. As software development has evolved and improved in the civilian domain, so has the standards in the military domain (Codur & Dogru, 2012)

2.3.2 Alternatives to the traditional models

There are a few alternatives to the traditional models where the focus is on the products, production process and the production environments. The alternative models came into being because
of the revisions over the traditional models. This alternative emerged because of the techniques like prototyping, environments and languages, reusable software, documentation efforts etc.

1. **Software rapid prototyping** is a technique where a limited functionality of the product is developed to get the feedback from the customer. In this technique, more importance is given to the initial stages activities like requirements analysis and specification. This helps to accommodate any changes in the initial stages of the project. The prototyping can be mock-ups, throw away prototypes, demonstration pieces or evolutionary incremental models.

2. **Joint Application Development (JAD)** is another technique where all the software engineers i.e. the developers, testers; end-users all work together to design and develop the user required system. This technique demands intensive group interaction and dexterity. The approach advocates collective behaviour as the development process is based on the idea that:
   - The one who works on the job is the one who is more knowledgeable of the job
   - People trained in software development have a better understanding of how to do the job
   - Best products are designed when everybody inputs equal participation

JAD is used where the project is well defined and has a timeline between 3 to 6 months. This model is best suited for softwares constructed from the commercially available softwares in the market like ERP packages.

3. **Reuse Oriented Software Engineering Process**: Organisations have already invested in a spectrum of software applications that cater to specific needs of the users in wide range of industries. In each industry, the requirement of similar softwares is common. So, reusing such software and customising it to the users is the main purpose of the reuse oriented software engineering process model. This approach works in the direction of reducing the effort on software development process and increase the quality of the software product to be delivered. There are a few known approaches to this:
   - Component based reuse: where the only relevant components of software are reused and not the complete system as a whole. The best examples for these are the current enterprise resource planning software modules of SAP, CRM, JDEdwards etc. The financial module or purchases module etc of the software are used as per the requirements. Implementation of medical software likes GP2GP, off the shelf available clinical information systems are the best examples in New Zealand.
   - Generation based reuse: where the high-level specifications and design of the software is re-used (Lockemann & Ateveh, 2006). For example, an e-prescription
health information exchange design can be reused to implement in different countries. However, as the government regulations, privacy constraints, consent models etc are different for different geographical boundaries, there will be some changes to the code accordingly.

- Service oriented Architecture (SOA) reuse: is another approach where the software services act as building blocks for building software. Services may be for general business purposes, a specific purpose interface of the business process or co-ordination of services. The process model can include identifying the candidate services for implementation, defining the service implementation, testing and deployment. In some cases, the services interfaces that are defined for the existing legacy business process and applications can be reused for new applications.

In the last couple of decades, the reuse of existing software design and code is widely being practised. The process model is similar to the other process except for that for the required specifications the component analysis is done and modified/customised requirements are generated for which the system design is reused to develop, integrate and validate the system. A standard reuse model can be as below:

![Reuse Oriented Software Engineering Model](image)

Figure 6: Reuse Oriented Software Engineering Model (Sommerville, 2011)

Typically, 3 types of component reuse is seen in the software development industry today:

- Web services
- Collection of objects designed using .NET or J2EE frameworks
- Commercial off the shelf software

This process model is very popular because of the obvious advantages:

- Reduces the amount of software to be developed. So, the cost is reduced
- The associated unknown risk is very little as the software is already in use
- As the software is readily available, the delivery is usually faster
The disadvantages however can be that:

- As the software is ready, some requirements of the users may be compromised. This may lead to not serving the purpose completely for the users
- Control over system evolution is lost as there is no much scope for change

The software development industry has evolved over the last few decades. With the emergence of the internet and the World Wide Web, there has been another niche created by many developers for the open source softwares. Organisations like Microsoft, Apple have started giving software updates to fiercely compete in the industry. By incrementally enhancing, releasing, debugging in an iterative manner. The time line for such incremental releases has been seen to be weeks, days or hourly. This revolution has given rise to Rapid increments, Incremental Evolution and Evolutionary Deliveries. This is a fresh area of research for the researches in the new millennium in the software engineering industry (Scacchi, 2001)

2.3.3 Agile Models

Change in requirements and the need to have the changes implemented within short period of time has been the main reason for high dynamics in the software engineering. Rapid software development because of such dynamics has been recognised by organisations like IBM, Microsoft, and Apple etc. The purpose of such rapid software development is to deliver the required software as quickly as possible. The different approaches for rapid software development have been popularly known as agile models. Many approaches like eXtreme Programming (XP), Scrum, Crystal Methods, Adaptive Software Development, Feature Driven Development, Dynamic Systems Development Method were seen between 1990s and now. The basic characteristics of the agile models across all the approaches is that the requirements specification, design and implementation are all interwoven, documentation is lightweight, the solution is delivered in increments in short intervals of time. The principles of agile methods are:

- To involve the customers throughout the development of the software
- To produce the software in incremental releases
- The process is people centric
- To be open to change in the requirements
- Maintaining simplicity in the software and the development process(Sommerville, 2011)

The Agile Manifesto was an initiative by a few representatives from eXtreme Programming, Scrum, Crystal Methods, Adaptive Software Development, Feature Driven Development and other like-minded methodologies in 2001 that discovered the need to identify and acknowledge an alternative to
document driven, heavyweight and plan driven software development. The Agile Manifesto advocates agile characteristics as

- Individual and team interactions against processes and tools
- Rapid delivery of working software over comprehensive documentation
- Customer partnership over customer contracts negotiation
- Change acceptance and management over following a plan

The manifesto further defined 12 principles with which agile software development teams work as

1. Highest priority is customer satisfaction achieved through quick and continuous delivery of valued software.
2. Accommodate change in every/any aspect and stage of the project.
3. Frequent delivery of working software. Frequency can be of any range from days, weeks and months where the time scale is very short.
4. The complete team including project managers, business/requirements analysts, developers, testers etc. to work together throughout the project
5. Building of the projects is around dynamic and motivated people.
6. Active and well-organized communication through face to face and continuous communication.
7. Progress is measured in terms of working software.
8. Sustainable development is promoted where the sponsors, developers and users can maintain constant pace through the process.
9. Ensuring virtuous design and excellence throughout the project and process
10. Simplicity in the processes
11. Best architects, requirements and design to evolve from the self-organised teams
12. Regular introspections by the team to become more efficient and adjust to the changes accordingly (Beck et al., 2001)

Many approaches have evolved and are further surfacing since the Agile Manifesto’s initiative.

A couple of Agile approaches currently being practised and very prominent in software development industry are:

1. **Extreme Programming** (XP) is one such process formulated by Kent Beck (who has been a part of Agile Manifesto). XP takes a lightweight approach where there are no large-scale requirements analysis and design phases. User stories are defined by the customers and the developers which have a short description of what the software should do. These user stories serve as requirements on which estimates are done and they are also used to define the acceptance tests. The development team is
usually of 5-20 team players. The rationale behind the size being that small teams can yield highly efficient output as all activities and communications can be managed effectively. Early prototyping is one of the by-products of XP. It involves iterative development, pair programming, simultaneous code writes and reviews, onsite client feedback and continuous testing. Small flexible groups, de-fragmented modules, continuous testability, frequent and effective releases are some of the features of XP. The process involves 5 phases:

- **Exploration: User Stories** are collected through meetings with client/ user and other stakeholders.
- **Planning:** When the user stories are prioritised and agreed upon with the client to further decide on the iterations and management
- **Iterations to Release:** Analysis, design and testing activities are carried out for each user story through pair programming. Customer feedback is also obtained by delivering prototypes
- **Production:** The completed product from the previous phases is then released into sandpit test environment where the user gets to play with the piece of software and verify against the user story. Once approved by the user, it is deployed into the production environment.
- **Maintenance:** Any enhancement for the already released piece of software (functionality as per the user stories) is done and the same cycle is followed through from the exploration or planning phase as required. This phase is worked and delivered as per the terms of the support contract agreed between the development team and the client.
- **Death:** This phase occurs when the support agreement is ended and the final release of the project takes place and handed over to the client (RamyaKrishna, Phanikanth., Phanikrishna & Vamsikrishna, 2011)

Below is the diagram for this process.
Advantages of XP

→ Planning of iterations ensure that the cost is not wasted on unwanted/low priority features for the customer.
→ Planning involves very less guess work.
→ As small releases are carried out, every release would incorporate customer feedback. This reduces slippage of requirements and tracking and tracing requirements is uncomplicated.
→ Simplicity in requirements analysis and design ensures that
  o time is not wasted on superfluous functionality
  o less confusion for developers
- easier understanding of the product by the users
- Continuous testing is possible through test-driven development. Automation of regression testing is possible through series of unit tests.
- Continuous revisit of the code by the developers ensures proactively improving the product. This also helps the developer to understand the product better.
- Pair programming increases focus as 2 heads are better than one. It further ensures coding standards and simultaneous code reviews.
- Collective effort and ownership ensures mitigating the risk of loss of leaving team member.
- This also promotes developers to take responsibility for the product quality.
- Small and compact releases ensure continuous integration.
- Continuous feedback from the customer helps answer many questions for the developer during development.

Another such eXtreme programming development methodology is the Test Driven Development where the software development is done in small iterations and is backed by the test first concept. Designing the test scripts is the integral part of the programming and is implemented even before the programs are written. (Tort, Olive & Sancho, 2011; Erdogmus, Jeffries & Melnik, 2010)

2. Scrum: is another agile model. It is an iterative incremental process skeleton. The figure below is a typical cycle of scrum. All requirements are collected as product backlogs. The iteration is the heart of this scrum process. The requirements are selected for the iteration. During the development, continuous monitoring and management is carried out through daily inspections. The team collectively determines how to build the functionality and figures out the optimum way to deliver the end products.

![Figure 8: Scrum Process (Schwaber, 2011)]
The scrum process involves 3 major roles:

→ The **Product Owner**
→ **Development team**
→ **Scrum Master**

The scrum process starts with the product owner envisaging the required product, anticipating the return on investment and understanding the different functionalities required and the milestones to be achieved. Then the product backlog list is defined through identifying the requirements and prioritising the backlog. The product owner then presents the prioritised backlog to the development team. The development team accordingly chooses the product backlog for development. This is when the backlog sprint/iteration starts. During the iteration, the team self-organises and manages the development process. The scrum master manages the team by getting daily updates from the team through a 15 minutes daily scrum/stand-ups session. The outcome of such typical scrum process is the incremental functionality of the product (Schwaber, 2011).

Many other models like the Crystal Methods, Adaptive software development, Feature Driven Development; Dynamic Systems Development Methods have evolved over the last two decades which follow the similar iterative processes. In all of them, the development is based on the immediate requirements of the customer like it being based on the feature the customer requires or the dynamics in the customers’ business or the market adaptability of the customers’ needs (Dingsøyr, Nerur, Balijepally & Moe, 2012; Qureshi & Hussain, 2008). Further tools like the Kanban charts (used for delivering the feature based outputs) have evolved that can be used as a methodology itself.

Thus software development process has evolved in the last few years to incorporate the incremental approach with one or few agile models through iterative process. Thus some organisations now use Agile/incremental models where deliveries are done every week or bi monthly and Agile/Iterative models were the deliverables are set to be delivered within anywhere between a month to 3 months.

### 2.4 Requirements Engineering (RE) and its presence in e-health

A requirement of a system is the description of what a system should do to accomplish a task. As defined by IEEE (IEEE, 2004),

“A *requirement is*

1. A condition or capability needed by the end user to achieve a goal
2. A condition or capability that is to be passed by a system to satisfy a specification standard or other formally imposed documents
3. A document representation of a condition or capability as in 1 and 2 above."

A requirement helps in defining the ‘WHAT’ factor of the problems and a better definition of WHAT can help in designing the HOW to solve aspect of the problem. The user never has the requirements. The users and stakeholders have information pertaining to the problem.

Engineering is the study and research of a field that can be applied scientifically for accomplishing a purpose. Technically, it is the science of converting the required specifications into tangible end products.

A process can be defined as a set of activities carried out as per certain procedures and standards using required checklists, formats, templates and guidelines for accomplishing a task (Chemuturi, 2012).

The main purpose of the requirements process in SDLC is to identify all the different processes within the proposed software and detail these processes so that further stages of SDLC can be carried out with client feedback. Each process has 4 different attributes which are the key factors with which requirements are defined.

1. **Inputs:** these can be external to the software product/process but is the starting point for the software to work as required. The requirements definition starts with identifying such inputs.
2. **Outputs:** these are the deliverables of the software product/process and the inputs serve as the feeds for the desired outputs. The required output is identified/defined during the requirements stage.
3. **Transformation Process:** the software uses the inputs and processes it to transform it into the required output. This transformation is another key process defined in the requirements stage.
4. **Triggers:** each process required some triggers which initiates the process where inputs are used to transform and deliver the outputs. Such triggers are also identified in the requirements stage.

### 2.4.1 Evolution of Requirements in different SDLCs

Chemuturi (2012) advocates that Requirements start as a single idea but evolve into a full set of business rules and specifications that further get processed into a software component/product/process/service/platform. There are different phases of evolution of requirements and depends on the type of idea.

#### 2.4.2.1. A new idea (product) which does not exist in the market:

For a new product idea, the requirements phases are:-

- The requirements starts as a **single idea**
• Goes through a series of brainstorming to deliver the initial requirements of the idea
• These are then triangulated by research through customer surveys, market surveys, consultant surveys
• The finalised requirements at this stage are then verified through personal interviews with experts like the marketers, users, educationists, designers etc. Any further inputs from them is further fed into the requirements
• The finalised requirements are converted into prototypes and presented to the users for trials. Further feedback from the users are taken and fed back as requirements. This is the final stage in the requirements process
• Once optimised and user is happy about the functionality or the user is not interested in further enhancements, the requirements are frozen and are documented. These requirements are confirmed and subjected to the inflexibility of change management. The full product is developed to this set of requirements.

This is a spiral model which can be embedded into a spiral software development life cycle. In e-health, this kind of prototyping is followed for applications like electronic health records systems, e-prescription applications, medical records systems etc.

2.4.2.2. A new product with the idea already existing in the market: This scenario exists if a product already is available in the market and the vendors intend to compete with the products.

• In such cases, the requirements start with the idea of competing with an existing product. Identifying the features that the existing product is not currently possessing is the key to this scenario
• Market Surveys are done accordingly to understand what the users are expecting out of another product of the similar kind. This consolidates the requirements from the idea generation
• Interviews are conducted to validate the requirements consolidated in the previous phase. Any feedback is further fed into the requirements document.
• Brainstorming is done by the designers, developers to enhance to enhance the requirements for the current market(user) expectations
• Prototypes may or may not be generated for the existing product but are developed for the enhanced requirements and presented in the market to get the feedback
• The requirements are frozen and documented and the development of the product takes place.

This process is compatible to the traditional models of software development.
2.4.2.3. **Product Upgrade**: This is about a product already available in the market for long and the existing needs improvements. These are identified from the feedback of the users, support team and the VARs

- The starting point of requirements is to get the feedback from the **user surveys** and identify the upgrade requirements
- Further triangulating the results from user survey by conducting **independent surveys** from the **support team** and **VARs** and analysing the results
- Conducting **personal interviews** to authenticate the identified upgrade requirements and identify any new requirements from the feedback
- Consolidating the survey results and feedback and analysing the requirements, documenting and **freezing the requirements** for proceeding with development

The above 3 are exclusively product based requirements methodologies. Product development and project development are distinguished by the outcome of the development. The output or end product of product development is targeted to be sold in open market. These products are usually known as COTS products. On the contrary, the outcome of the project development is a customised product that can be used by one customer. This is the reason requirements process is different for product based software development and project based software development. In the project development, the customer preference is given utmost importance. The scenarios in which project based software development is carried out are in house projects and out sourced projects.

2.4.2.4. **In-house-new project**: The following is the process of evolution of requirements process in the in-house new projects.

- **Proposal by the functional team**: specifying the new need like computerising a new process is submitted to the IS department
- **User Requirements**: business analysts elicit and gather information about the new requirements by using techniques like interviews, studying manuals. They use different templates and forms to collect the information. All this information is collected and analysed to document the user requirements
- The document is then shared with the functional team for their feedback. A meeting between the business analysts and the functional team helps in getting feedback and negotiating the requirements. Once agreed, this feedback is incorporated into the document and the **requirements are finalised** and sent to development team

2.4.2.5. **In-house-upgrade project**: These projects also perform requirements process as defined in the in house new projects. However, the purpose is to scale up existing software with new technology or functionality like moving a standalone application to be a web based application.
2.4.2.5. Out-Sourced-new project: this is a scenario where an organisation has a new requirement and decides to outsource to a specialised software development company. The evolution of requirements could be:

- **Project Acquisition**: where the vendor company acquires the project from the organisation that is outsourcing. The organisation may outsource all activities of the SDLC or a few of them.
- If requirements activities are outsourced, then the next phase is to **elicit and gather the requirements** from the outsourcing organisation through techniques like interviews, reading existing manuals, studying the existing process models, the dependencies of the processes.
- **Analyse the requirements** to ensure the feasibility of the project, classify the requirements and prioritise them.
- **Finalise the requirements** by documenting them and ensuring that the document is as per the internal standards of the organisation.
- Agreeing the requirements with the clients by sending the requirements document to the outsourcing organisation and getting their **approval**. Once approved, the requirements document acts as a reference document throughout the other activities in the SDLC.

2.4.2.6. Out-Sourced-upgrade project: Usually upgrade of source code is not preferred by many organisations to be outsourced. If upgrade is to have the same functionality but move to another platform, such kind of projects are usually outsourced. The possible evolution of requirements in such a scenario could be:

- **Acquisition of proposal**: The vendor company understands/ lists the requirements from the RFP and these serves as initial requirements. The vendor company responds to the RFP and gets to an agreement with the outsourcing company to deliver the project.
- Once the response to RFP has been approved by the outsourcing organisation and the vendor agrees to deliver the project, then the **requirements are further elicited and gathered** by the BA of the vendor company. The tools used by the BAs for this are usually interviews, surveys, existing manual study, existing process study etc.
- **Requirements are analysed** for its completeness, identify any gaps in requirements and feasibility to implement.
- The requirements are documented and presented to the outsourcing organisation and meetings are arranged for getting any feedback from the customer. The feedback is fed into the document and the **requirements are then finalised**.
- Once the **customer approval** is obtained, the requirements document is used as a reference document during the SDLC.
2.4.2.6. Requirements in Agile Projects:

RE process in agile projects is diverse to the traditional software development projects. The accommodation and adoption of rapid changes in stakeholder requirements and preferences, competitive environments, market pressures, development technologies is the important aspect of agile software projects. In many agile projects, the development starts without the formal requirements analysis and design phase. The requirements evolve and mature through continuous feedback from the customer throughout the development of the project. Agile methodologies like XP and Scrum advocate that RE process is carried out throughout the life cycle of the project for every small iteration/stage. However, authenticating the suitability, purposefulness and reliability is a challenge as such process does not look well into the future. However, these can be mitigated by incorporating unambiguous negotiation of requirements, establishing and ensuring traceability with aspect oriented approach to the requirements. Cao and Ramesh (2008) conducted a study and identified seven important aspects of RE in Agile projects.

1. **Face to Face communication** is more preferred over the **written documents** of the specifications.

2. **RE** is done **at each iteration**’s development cycle. Further, RE is done collectively with the design. The advantages of this approach is that
   - There is continuous interaction with the customer and this helps build customer relations
   - The requirements are done for the iterations so only those relevant and agreed with be worked on there by controlling the cost incurred

3. The high level requirements are prioritised initially. However, **prioritisation of requirements is done at each iteration** and the prioritisation is revisited. The prioritisation is done at the starting of each development cycle. Further, the prioritisation is continuously based on the current business value defined by the customer.

4. The **changes in requirements are managed with constant planning** throughout the different iterations of the project. Two types of changes that can be possible are adding and removing features, changing the existing and implemented features. This also involves constant refactoring of the code to continuously improve the performance of the features. The study showed that the reason for change is the constant interaction between the customer and the developers. However, in some cases such changes can require change in the architecture of the solution which could mean an expensive overhead to the customer. Further, refactoring can call for series of intensive development and regression testing that could add to the overheads.
5. **Prototyping** is an intrinsic part of the iterative process. The end product in each iteration acts as a prototype for the next stage. So, these prototypes, in many organisations, are deployed because of market pressure and subjected to innovative version deployments depending on the built features.

6. **TDD** is possible in Agile projects as before the new functionality is finalised, the developers start writing unit tests. So, writing tests at early stages is an intrinsic feature of RE in agile projects. This can help in tracing the requirements at an early stage and the tests can ensure through feedback from the developers that the requirement is defined as required. So TDD acts as a verification phase required for RE. TDD could be a challenge at times because the developers’ do not get accustomed to the idea of writing tests at the early stage. Further, writing the tests will require extensive understanding of the current features.

7. Incorporating **UAT and review meeting** in the Agile RE process helps the development team and the customer to define the acceptance criteria. The UATs suggest if the project is going in the right direction at the end of each iteration and review meetings to review requirements are the verification technique used during beginning of each development cycle/iteration.

All these features does authenticate that RE process in Agile projects are different to the traditional RE process. The agility in RE is seen in projects where the development is done in ambiguous environments, defining complete requirements before development is not possible and continuous customer feedback is required/available. The agile RE process is dynamic and is not centralised to one phase of the development lifecycle.

Hamley, Kamel & Hegazy (2012) identified that a few software architectural challenges that arise in agile RE processes are because of incomplete elicitation of requirements, inappropriate prioritisation of user stories and deficient insight into the non functional requirements. They suggested that along with the requirements identification, grouping, prioritising; the identification of non functional requirements and envisaging the architecture should also be incorporated into the RE process before the tasks are identified and developed.

### 2.4.2 Requirements Engineering

Going by these descriptions above, RE in software development is a methodical approach to the specification and management of requirements. RE discipline focuses on both the problem and the solution required. RE interferes with system engineering to analyse and define the problem and suggests the required different functionalities out of the software (Ebert, 2010).

This is the initial process in software engineering. The RE process encompasses the following main activities

- Requirements Elicitation
RE is a collateral process which needs to be carried out throughout the project in agile environments. The RE processes are influenced by people, stakeholders, domain factors, organisational approaches towards it. The focus of RE is on the information provided about the problem by the stakeholders and not the stakeholders themselves.

2.4.2.1 Requirements Elicitation:

In its glossary, IREB defines requirements elicitation as

“The process of seeking, capturing and consolidating requirements from available requirements sources. May include the re-construction or creation of requirements.”

(IREB, 2013)

According to the i.e. IREB (2012), the purpose of this activity is to identify the conscious, subconscious and the unconscious requirements of different stakeholders. So, the different viewpoints of all the stakeholders are to be gathered. Different sources of requirements are to be identified and requirements are to be gathered. A conscious effort has to be made to not disregard any important sources as it could lead to incomplete requirements or requirements with gaps. The sources could be the stakeholders, relevant existing documents and legacy systems. Different techniques of requirements elicitation are required at different stages and types of projects. The following are a few techniques listed in BABOK that can be used.

- Survey techniques like questionnaires and interviews for getting the initial information from the stakeholders
- Brainstorming for getting the analogical perspective and the paradoxical view of different stakeholders
- System archaeological techniques like studying the existing processes, legacy systems, existing use cases etc
- Document based techniques like reading existing documents, requirements reuse feasibilities
- On the floor / field observations
- Workshops, audio recordings of the meetings, video recordings, process mapping techniques like prototypes, use cases modelling etc
- Document and Interface analysis (IIBA, 2009)
Choosing the appropriate mix of techniques is a competency critical to the project. IREB further advocates that elicitation should cater to the satisfactory needs of the stakeholders by categorising the requirements into different categories by the basic factors, performance factors and delighters.

Blais (2012) suggests that having an IGP can help in envisioning the problem as defined/required by the stakeholder, reduce the time and cost by setting less information gathering sessions, extract more usable information from each information gathering session. The IGP focuses on what information is to be extracted, where is it found, how the information should be extracted and in what order should it be acquired. The information can be the actual functional information, the dependencies and the supporting information. The ‘where’ defines the source of information which can be the documents, current and legacy processes, and the different stakeholders. The ‘how’ refers to the different
methods of gathering the information. Apart from techniques suggested in BABOK, methods like business and use case analysis, workflow and technical analysis of the current/required process can also be incorporated. In the real world scenarios there are many issues that the requirements engineers face during this phase. The issues can be

- The users may not know what they want
- The users may not be able to effectively communicate what they want
- The clients do not want the requirements. This means the user does not want to spend money on the requirements engineer creating the document. However, the requirements should be granulated and listed for the designers to further work on them
- Some users can get a bit greedy and want all the requirements and would prioritise them all as important and tag them and ‘wanted NOW’

Durugbo and Riebel (2013) suggest that requirements elicitation activity is carried out parallel with the other activities like analysis, validation and negotiation of requirements. This concurrency helps in lowering costs, identifying any gaps in requirements at early stages and implementing relevant required features in the software. Many researchers have worked in this direction and have proposed different models. (Aranda, Vizczi’no & Piattini (2010); Coughlan & Macredie (2002); Hickey & Davis (2004)). Aranda, Vizczi’no & Piattini (2010) proposed a framework for requirements elicitation in GSD projects where the focus was on predicting future problems and strategies and solutions that can reduce the impact of such problems in the long run in the software. Another framework was proposed by Coughlan and Macredie (2002) where the focus was on the communications model in requirements elicitation. They suggest that effective communication is the key for identifying and eliciting the requirements. Hickey and Davis (2004) argue that an improved understanding of requirements elicitation process helps the analyst’s ability to elicit quality requirements by selecting appropriate elicitation techniques. This further leads to getting things right initially and an outcome that the customer is expecting finally. They presented a unified model of requirements elicitation where the analysts take a requirement from the current state to a better understood state with appropriate elicitation techniques. This helps the software development team to develop software that matches to what the customer needs.

**2.4.2.2 Requirements Analysis:**

Analysis is a thorough inspection of all the essentials of something for interpretation in simple language. The purpose of analysis is usually to deliberate and interpret something that is complex.

Bastani (2007) suggested that intangible nature of software at the RE stage makes requirements analysis a very complex area of engineering work.
A RE framework that should be able to incorporate

- flexibility to incorporate requirements, architectural changes at any phase of the project
- flexibility in creating, updating or removing entities/components and process at any phase

Abstraction Oriented Frames which is a hybrid of 3 models where:

- a framework for the defining the problem using questions like who, what, how and when
- a use case approach of requirements model and

Abstraction based classification of the requirements as an analysis model.

Once elicited and gathered, requirements are need to be catalogued as from here, the requirements are traced by the reference numbers listed to them.

A thorough understanding of the problem by analysing the problem domain is the initial step towards this. The broader problem domain is analysed, the better it is for identifying the different departments/areas that it is scattered across. This kind of analysis helps in completely determining the source of the problem, its symptoms, the areas it impacts and the point where the solution is required. This can be best achieved through the tool - Domain Diagram Model where the problem domain is diagrammed with the different work flows. The main purpose of requirements analysis phase is to analyse the information gathered to determine the solution. This is done by determining the problem and the envisioned solution, elicit information to determine the problem domain and then analyse the information to define the optimum solution.

![Figure 10: Requirements Analysis Process (Blaise, 2012)](image-url)
The purpose is to identify/define the business process for the process workers and the activities associated with the process. This also helps in discovering any gaps and contradictions in the requirements elicitation process. There are usually some components identified like:

- The **goal** that is to be achieved
- The **generation** of results that leads to achieving of those goals.
- The **requestor** who uses certain **setup** that can **trigger** the process under some defined **pre-conditions**.
- The **Beneficiary** who generates the **results** from the business process for some **post-condition** activities
- The **supporting information** and **resources** that supply some input to the business process that are being analysed

Once these components are defined, the inside process gets self defined. Understanding the boundaries of the system context and its relevance to the environments and defining the irrelevant environments is the first step in RE. (Pohl & Rubb, 2011).

The other activities in analysis are

1. the categorisation of the information into
   - the scope of the project (and not in scope)
   - Functional and Non-functional requirements
2.4.2.2.1 Requirements Prioritisation: The purpose of requirements prioritisation is to ensure that the analysis and implementation effort is focused on most critical requirements for the iteration. It is a process to establish the comparative importance of a requirement by its value, implementation feasibility and the risk associated with it and many such other criteria. Prioritisation helps in planning as to which requirement should be analysed and implemented next in the iteration. It is dependent on the key goals defined in the business case/business need and is determined by the stakeholder’s role and responsibility and their annotation of the level of authority they exercise in the project. The prioritisation is based on

- the cost-benefit analysis and the business value the requirement provides
- the risk it carries from the business and technical point of view
- implementation feasibility and the possibility of the success
- dependencies on the other prioritised requirements
- demands associated with the policies and regulatory frameworks of the business
- the importance/need/urgency and the agreement with the stakeholders

The issues that the requirements engineers face during prioritisation can come in the form of unrealistic expectations by the stakeholders by rating all the requirements as high in priority, multiple stakeholders’ exercising the authority on different requirements at the same time. This could be because of the behavioural issues within the stakeholders. Domain specialist, implementation specialists, the users along with the commercial stakeholders and the project managers decide on the priorities of the requirements.

There are many techniques currently prevalent in the market today. BABOK recommends a few
• general techniques like decision analysis through decision tables and decision trees, risk analysis

• MoSCow Analysis where the requirements are categorised on the basis of Must, Should, Could and Won’t. The key thing here is during prioritisation is if the “Won’t” requirements are not identified then this could potentially lead to not identifying the negative requirements in the implementation plan and could lead to commercial chaos

• Time Boxing Analysis where the priorities are analysed and set based on the amount of work the team can deliver within a given period of time

• Budgeting Based Analysis where the project is set for a fixed budget and time, the deadlines are to be met within the stipulation.

• Voting where a fixed amount of resources are provided to be distributed within the requirements for the participants to decide on the delivery schedules (IIBA, 2009)

Another method the case based ranking (CBRank) is an iterative prioritisation framework that can handle multiple stakeholders and decision makers with different criteria of ordering. Perini, Susi and Avesani (2013) suggest that this method exploits machine learning technique and can handle up to 100 requirements. The process is based on the history of prioritisation of similar requirements in the past for similar projects. The new requirements are paired with the past implemented requirements and the associated prioritisation criteria are implemented accordingly.

Daneva, Veen, Amrit, Ghaisas, Sikkel, Kumar, Ajmeri, Ramteerthkar, & Wieringa (2013) have found that in agile projects, the requirements prioritisation is based on not just what has been deliberated above, but it encompasses the following:

• Requirements dependencies are key factors to be considered in prioritisation

• Most important aspect next to business value is the risk associated with the requirement. So, this should be the primary consideration in the prioritisation process

• “Delivery stories” are deliberations on time and effort estimates, associated risks and technical implications of the user stories (requirements in agile projects). So these should also be taken into consideration when prioritising requirements

• SME and vendor’s domain knowledge are vital assets for the collaboration with the development team. So, availability of SMEs also should be considered during the prioritisation process.

The requirements are then checked for its completeness. This is done by identifying the primary and secondary stakeholders and ensuring that each stakeholder’s need has been addressed through the requirements. Further, every requirement is assessed for its technical, financial and timeline feasibilities within the defined scope of the project. This process is not complete within itself. This
further follows a sequence of prioritisation and agreeing with the primary and secondary stakeholders within the bounds of technical, financial and timeline feasibilities. Once this is done, the requirements are grouped as core functional requirements and supplementary functional requirements. This further helps in sanitising the requirements by eliminating the duplicates in the requirements, identifying contradictory requirements, identifying and resolving any gaps in the requirements. All these will determine if further re-engineering of requirements is to be done. (Chemuturi, 2012)

Once all these activities are achieved the next phase in RE is the documentation phase.

2.4.2.3 Requirements Documentation

At this stage of RE, the requirements would already have been classified based on its functionality, software/product construct and by the source. Functional requirements can further be seen as core and ancillary functional requirements. The core functionality defines what the software should do. The ancillary requirements elaborate on the esteem, reliability, fault tolerance, statutory, safety, usability, data integrity, response time and such features. Product construct based requirements deliberates on issues like maintainability, flexibility, efficiency, reusability, portability etc of the software being delivered. The source based requirements is classified by the end user, management of the customer organisation, domain experts, project team, software designers, software developers and programmers etc. All these have to be captured in the document for different purposes. Different requirements documents serve different purposes.

Classifying requirements into different categories like the functional, non-functional and crosscutting requirements benefits system centric analysis. This helps the requirements to be understood better because they tend to be comprehensive, scalable, traceable, reusable, and evolvable and thus flexible (Ranjan & Misra, 2006).

The initial set of requirements in an ideal requirements documentation process should be the **Business Requirements Specification (BRS)**. The purpose of this document is to clearly define the business need from the perspective of the different sources from where the requirements are elicited. The document gives high level requirements of the business as anticipated by the end users, the stakeholders, business process owners etc. The rationale behind BRS is to visualise and capture the intangible business requirements. The document clearly describes

- the business need or the problem
- high level business process impact (stopping certain process and start some other new ones)
- high level justification of the business need and how it can help enhance the business process (Smith, 2008)
Effective definition and documentation of BRS is the key factor as this serves as an instrument on which the vendor companies and customer organisations get into agreement. The clearer the BRS, the better it is for the project in terms of the quality and meeting the business purpose (Chandra, Ravishankar & Sharma, 2008).

The BRS forms a basis for the **User Requirements Specification (URS)**. In majority of the cases, the URS deliberates on the different use cases that will be affected from the required new functionality. The target audience for this document is the end user and the software development team. The end user uses this document as a reference to interact with the development team. The requirements should be grouped and documented in a logical manner – as it has been analysed. The nomenclature of the requirements should be consistent with the business requirements and other document sources to reduce any ambiguity within the development team and the stakeholders. The general outline of a URS document follows a structure that includes the title pages, the contents pages, and the requirements definitions in the form of use cases and the concluding pages in the form of annotations / addendums/appendixes.

In Agile projects, the user requirements are usually in the form of **user stories** and not elaborate documents. There is no standardised way of writing user stories. Organisations write their own customised user stories as required by the team. Shafer (2010) advocates that user stories serve two purposes as requirements and test scenarios. As suggested earlier, user stories eliminate the need of the requirements document. Below is the perception of a user story in the Agile projects as seen in his work.
Figure 12: User Stories in Agile Life Cycle (Shafer, 2010)

However, the user stories cannot incorporate the non-functional requirements specifications in them. As such these are very useful for the users and the functional analysts and are a challenge to the development team.

The URS feeds as an input for the **System Requirements Specifications (SRS)** document. The purpose of the SRS is to categorise requirements as the functional and non-functional requirements. IEEE has developed the standard for SRS and specified that the characteristics of a good SRS should contain correct, unambiguous, consistent, complete, modifiable, traceable, verifiable and prioritised requirements. The SRS contains the technical specifications required for the designers. So, the purpose of generating this document is for internal reference by the software development team. This document need not be seen by the customer.

In Agile projects, prototypes are built based on the URS and these serve as SRS for the designers. Organisations enduring agile projects document the URS and create prototypes instead of SRS.
document creation. This helps the end users to perceive the technical aspect of the software better. In some cases, the end users do not approve the URS documents also. They would like to just see the end product. In such cases, the process followed in Agile projects is to derive the prototypes based on the user stories and get the customer feedback about the functionality and the designer’s inputs into the technical aspect of the project (Chemuturi, 2012).

A standard SRS usually contains specific sections for hardware and software specifications, interface specifications, core functionality and ancillary functionality specifications.

2.4.2.4 Verification, Validation and Management of Requirements:

**Verification** of the requirements is done to ensure that right things are done as per the specifications. This does not include any testing of the requirements. Verification ensures that what is being built is right. This is done by peer reviews and management reviews.

Peer reviews can be done independently or through guidance as an individual effort or a group effort. The purpose of peer reviews is to ensure that

- the technical content is accurate and comprehensive
- the organisational standards are as per the defined guidelines
- the end product is clear and unambiguous

The process of a standard peer review process can be seen as follows depending on whether it is an independent peer review or a group review:
Once the requirements are peer reviewed, the management reviews the requirements. The management team consists of the management team from the customers and the project managers and project sponsors etc. The purpose of this is to have a bird’s eye view and agree on the final requirements of the project/iteration.

IIBA (2009) suggests in BABOK that the analysts verify the requirements for its cohesiveness, completeness, consistency, correctness and feasibility and ensures that the requirements are unambiguous and testable. General techniques like acceptance and evaluation criteria definition, problem tracking and walkthroughs and checklists can be used for verification of completeness of the requirements. The possible tools that can be used are e.g., brainstorming, story boarding, prototyping, and reviewing.
**Validation** on contrary to verification is demonstrating that the software does what it is intended to do and aligns with the requirements. This is usually done throughout the project/product life cycle. Test driven development is one of the best examples of continuous validation throughout the life cycle of the project. Testing plays a major role in the validating the output.

Samaras & Horst (2005) suggested the difference between validation and verification as solving the correct problem is validation and solving the problem correctly is verification and can be illustrated as below:

![Verification vs. Validation](image)

**Figure 14: Verification vs. Validation (Samaras & Horst, 2005)**

IIBA (2009) suggest that business cases, stakeholder requirements are the inputs and helps the stakeholders to validate the software and determine the business value, the dependencies and evaluate the alignment of the end product with the business case and the opportunity gain or loss from the software. Thus this phase is very vital in the RE process

**Requirements Management** is a planning and implementing consolidated activities of analysing, monitoring, communicating, verifying, validating and tracing of requirements. The outputs from the planning are the business analysis approach and the requirements management plan. The **Business Analysis Approach (BAA)** deliberates on the fundamentals like the framework of business analysis, the different methodologies that will be used to elicit, gather and analyse the requirements, the prioritisation criteria, the different tools and techniques that will be used etc. The purpose of this is to communicate to the appropriate stakeholders the approach taken for RE. This sets a baseline for the stakeholders’ understanding of the process and they will know what to expect with the requirements. The **Requirements Management Plan (RMP)** describes how the baseline of the requirements will be set, documented, communicated, monitored, traced and changed when required. This serves as a
roadmap for the project managers to further plan the iterations/releases/phases of the project. This can act as a subsidiary to the project plan. This could also help in the billing and other financial activities of the project (Larson & Larson, 2010a).

**Requirements Traceability**: the ability to follow the requirement backward and forward through from its generation to the development, verification, validation, deployment, use and change across the iteration or different phases of the project. Traceability of requirements is required for different reasons:

- **As a practice, it helps the end users**
  - To understand the requirements decomposition, allocation and dependencies.
  - In Meticulous compliance verification
  - Controlling the change
  - Asset management
  - Compliance across the process/product dimensions
  - Measuring the progress and monitoring the requirement

- **As supported analysis, it helps in**
  - Analysing the impact of the change
  - Analysing the coverage of the requirement in the software engineering, management and reporting
  - Analysing the financial advantage of the requirement

- **As a quality attribute**
  - To improve the usability, safety, maintainability, security
  - To regulate any deviations and provide software assurance
  - For financial accountability

Unfortunately, tracing of requirements is not as easy as it sounds. Finkelstein (1991) suggested that there are many problems encountered during the process of tracing requirements. Jarke (1998) deliberated that establishing the pre-traceability links forward and backward of requirements is challenging. Gandhi, Lee and Park (2010) further examined that the pre-traceability links are from multi-dimensional perspectives like business goals, business vision, low level end-users, the domain knowledge owners and subject matter experts, the different environments the requirement impacts and the different business & legal regulations that the requirements is to be bound with. Any ambiguity in these links leads to problems in traceability. The identification of these links is the key to forward and backward traceability as they help in capturing the diversified views of the requirements.
Sultanov (2011) applied swarm intelligence to establish the links between the words and terms in the documents. Each individual swarm member (document) exhibits some actions (with reference to the keywords in the document) and collectively exposes some relationship and patterns. This was applied to trace links from high-level to low-level elements of the document. Each pheromone left by the member is taken into consideration as a pheromone deposit by the other member and thus the links are established.

**Requirements Communication:** One of the main causes for project failures or project overrun by time and cost are the ineffective communication of the requirements. This could be a failure because of cost or time overrun. Good RE practises contribute significantly to the success of the project.

- Defects identified at the RE phase are very easy to fix
- Defects identified at UAT cost 20 times more than when detected at RE Stage
- Calls for higher levels of quality for RE process

Today the Requirements Engineers are expected to have domain knowledge, the RE process knowledge and multitude of soft skills. Effective communication in the form of

- **Verbal Communication:** relieves the users from heavy documentation
- **Technical Communication**: information is transmitted with minimum redundancy and feedback

In agile projects, these communications are the key to managing the congregation, evolution and illustration of requirements. Abdullah, Honiden, Sharp, Nuseibeh & Notkin (2011) identified that these communications flow in a pattern within the RE activities of the team. The shared conceptualisation of the project by the development team and customers is the result of effective communication and collaboration during the RE activities and this is the bases of the effectiveness of the RE process.

**Requirements Management Tools** (RMT) are used to simplify the RE process and establish systematic approach to handling and tracing requirements and managing change. The tools can be helpful and can

- Support elicitation by storing and managing elicitation templates, checklists, prioritisation forms etc. The requirements can be grouped by objects of **Requirements Interchange Format** (ReqIF) this can be used across the same project or across other projects
- Support verification and validation by generating exception reports for verifications and validations. The tools can provide standard interfacing formats that can be fed into different verification and validation tools. They can also implement checks to confirm if the requirements are complete, consistent, unambiguous, traceable, prioritised, verifiable and understandable
- Support modelling of business processes, business goals, systems, process flows and workflows
- Support specifications with the ability to inspect the document using spell check, grammar check, dictionary support etc. These tools can also feed on granular requirements to generate the specifications document that has been properly indented, page marked, tabulated etc. Further, any changes in the requirements are instantly incorporated into the requirements document
- Support traceability by
  - Generating comparison reports of different versions of the document
  - Tracing the requirements objects through the boundaries of different tools of verification and validation
  - Tracing text to graphics, graphics to graphics, tables to cells of tables
  - Generating traceability matrixes

Today there are many RMTs in the market. Carrillo de Gea, Nicolás, Alemán, Toval, Ebert & Vizcaíno (2011) have established that there is no single product that incorporates all these features in
one tool. They described that the requirements management needs are different for different organisations. The development projects where identifying the user needs is very critical by the client has very little time to spare, the need is to ensure that the output or final product or output is to be as per the specification. In such situations, the RMT should be able to help in elicitation and verification and validation. So, selecting the tool which has strength in these features is required. On the other hand, if a project is design and specification intense, then the RMT with strong specification defining capabilities and modelling features is to be chosen. Further, if a project’s nature demands strong control and quality assurance, then accordingly, the RMT should have strong traceability features.

2.4.3 Requirement Engineering in e-health

In today’s healthcare industry, electronic health records, e-prescribing, broker managed message exchanges are the wished-for ways of managing the technical and organisational complexities of the day to day activities of the healthcare professionals. Today, the approach is to customise and simplify the complexities to streamline healthcare activities and shift focus from process centric to patient centric healthcare. There have been many approaches in this direction. Below is the deliberation on this aspect with reference to different RE frameworks for e-health projects:

1. **Mobility Work**: where the analysis focuses on the movement of system users, resources, information and spaces in hospitals. Activities before ‘going mobile’ like information seeking, transfer of information to mobile artefacts are analysed in the RE process. Further, post mobilisation, certain activities like transferring of data from mobile artefacts to the fixed repository (like EMR, EHR etc) are also analysed. Bardram and Bossen (2005b) prototyped requirements considering these kinds of activities to provision EHR systems, mobile information access systems, content based information access (Decision Support System) and for information and knowledge based systems.

2. **Common Information Space (CIS)**: This concept was introduced by Bannon and Bodker (1997). They proposed two important aspects of CIS
   - CIS is open, flexible and easily understandable space where there are a number of highly relevant work activities
   - CIS also defined the creating of closures which punctuates and defines the boundaries of the space which helps in relevant information being available and used for various tasks within the different groups of the workspace. This is the reason for the dynamic behaviour of CIS in many environments.

One of the best examples of CIS in a hospital is an intranet website that caters to all different users where each user requirement is unique and different. Further, such intranet can be accessible over the internet for authorised users. This concept of CIS has been researched by many Computer Support Co-operative Work (CSCW) researchers like
Reddy, Dourish & Pratt (2001) to analyse the collective work of different groups of people in the intensive care ward of a hospital
→ Bossen (2002) to analyse the work patterns in hospital wards

As stated by Reddy, Dourish & Pratt (2001), the focus of CIS within an intensive care unit could be to share information system artefacts like EHR, Whiteboards, and Clinical Decision Support Systems etc. So the analysis and requirements design could centrally focus on

- Consistency of data across users incorporating changes in real times
- Access feasibilities to multiple users
- Multiple representations/interpretations of the same data for different users
- Possible data leaks during information exchange and information change and how this can be mitigated by use of technology
- Information review and update support tasks and how and to what degree such information is needed by different users/stakeholders

3. **Temporal rhythms:** involves looking for information, providing information and managing activities that healthcare providers should follow to accomplish tasks in CSCW. They are recurring patterns of work. Typical temporal rhythms in a hospital environment could be rounds by doctors/nurses, shift change by the care givers, patient medication administration, scheduled catch-up of healthcare providers, educational intents like interns recruitments etc.

The fundamental characteristics of such temporal rhythms are:

- Temporal trajectories: that defines the timelines of the temporal rhythms like defining the time healthcare professionals use for accessing patient information (Reddy, Dourish & Pratt, 2006).
- Temporal Horizons: that defines the frequency or the time period between the re-occurrence of the activity.

This concept motivates in designing information systems that can accommodate such rhythms. The analyst should be able to identify these temporal trajectories and horizons for the information system. The identification and computerisation of these can support the users to accomplish tasks better (Doherty, McNight & Luz, 2010).

4. **Cognitive and Coordinative Artefacts:** Doherty, McNight & Luz, (2010) argue that Human and Computer interaction (HCI) can be successful through artefact analysis where Distributed Cognition (DCog) and Activity Theory of the objects are identified. Two basic forms of analysis can be cognitive and coordinative. The concept of DCog advocates that the knowledge lies not just in the individuals but also in the physical and social environments in which the individual operates. According to Nemeth, O’Conner, Klock and Cook (2005), the
automation in healthcare fails in many cases to improve the clinical routine and performance because the analysis and design fails to incorporate all aspects of DCog. The cognitive artefacts analysis in healthcare projects focus on only the artefacts created and used by healthcare professionals to accomplish some goals and tasks. The requirements of such systems should be to maintain the affordance of the artefacts in the existing systems and designing effective workflows that can help to process the data perceptually within the artefacts. Automation of such artefacts should be able to enhance coordination between the different artefacts. So, the focus of analysis has to include maintenance of such coordination. This can be done by identifying the dependencies and ensuring to incorporate those connections within the artefacts in the development and maintenance of existing systems (Bardram & Bossen, 2005a).

5. **Activity Based Computing (ABC):** Bradram and Christensen (2007) discovered Activity Based Computing as an alternative to document and application centred systems in healthcare environments. This approach is activity centric and incorporates the following:

- Activity discovery: where activity is identified based on the context
- Activity suspend-resume: where the user can deal with interruptions
- Activity roaming: tolerate activities to be mobile and ubiquitous
- Activity sharing: which can incorporate coordination as discussed previously in this section

Requirements gathered for such ABC systems can help activity management by allowing the users to suspend and resume activities as required in the workflow, maintain the activity when itinerant (possibly changing the user interface devices like Smartphone’s, tablets etc) and coordinate activities by sharing the current activity in different environments and contexts. A typical example of this is where the healthcare professional is examining the patients and updating the patient clinical information on handheld device when moving from one patient to the next. A possible coordination of activities is when the healthcare professional would want to update the accounting department of the billing of the services to a patient at the hospital.

All these above mentioned frameworks call for tremendous fieldwork to be incorporated in the analysis and design of the requirements. Doherty, McKnight and Luz (2010) used these frameworks to study lung cancer patient review process in a large scale hospital. Each concept helped in generating individual and some overlapping, related and challenging requirements. However, these concepts could not incorporate the patient’s perceptive in the review process which is a major component. Further, the extent to which the frameworks influence the requirements generation is still
to be assessed. However, they argue that the application of domain knowledge is beneficial during gathering the requirements.

McGee-Lennon (2008) analysed another aspect of e-health i.e. the home care technology by consolidating the existing research in this area. HCT is a consolidation of services like social services and healthcare. The characteristics of home care technology are multi-user, multi-stakeholder, distributed and dynamic. The interactions within each service and across services could be multimodal. The complex and dynamic nature of HCT is because of its complex care conditions, relationships, shared interactive spaces, accountability and multiple changing requirements. The RE process can get challenging as different users and different stakeholders has different priorities for the same requirements. McGee consolidated many researchers’ works and came up with an ideal RE process for home care to contain the following:

- Participated and distributed requirements elicitation and negotiation
- Iterative elicitation and validation of requirements
- Identifying and engaging relevant stakeholders to elicit requirements
- Categorising and prioritising the requirements
- Resolving any conflicts in requirements
- Retaining and tracing requirements
- Numbering of requirements to facilitate future referencing
- Correlating the requirements with other practises

The process seems complete within itself. The process advocates including stakeholders at the design stage but does not clarify how the change in requirements is handled.

Salini and Kanmani (2011) designed a model based security RE framework for online trading system, improvised it to suite any web application (Salini & Kanmani, 2012a) and extended its usage to e-health system (Salini & Kanmani, 2012b). Patient medical information is very vital in medical diagnosis and decisions and is very sensitive in nature. Using Model Oriented Security RE (MOSRE) framework for eliciting requirements for e-health system and considering these requirements as a part of functional requirements helps to identify

- Business assets like patients and doctors details, medical documents, treatment plans
- System assets like the existing software technology, databases, servers, hardware and network
- Threats
- Weaknesses
associated with them at requirements elicitation stage. This helps in defining the high-level architecture. The business requirements gathered can feed into the use case modelling. This helps in identifying the threats, vulnerabilities of the e-health system. Further, categorising and prioritising these threats and vulnerabilities will help defining the misuse cases. The requirements representation through use cases and sequence diagrams will further facilitate development and testing. This can also help in defining the possible negative aspects of the requirements. However, this is granular requirements and handles only the security aspect of e-health systems.

In HIS, incorporating analysis of incremental hazard analysis, preventive and post-deployment corrective actions serves the purpose of having the user interfaces to be human centric. Samaras and Horst (2005) view this to be a vital point in Systems Engineering of HIS. They suggest that the requirements are the basis on which validation process of any software is carried out. The incorporation of human centric ergonomics like hardware and software ergonomics, environmental and organisational ergonomics in analysis and formulation of proper requirements based on this is the key factor in development and implementation of e-health projects. This will be the basis of the design and further activities in the SDLC. Many researchers (Gould & Lewis (1985); Sibley, Mantei, Teorey, (1988); Nielsen (1992); Shneiderman (1998); Mayhew (1999) and Endsley(2002)) have studied the SE models but have not emphasised on the incremental hazard analysis in their iterative designs. Analysing and incorporating the Corrective And Preventive Actions (CAPA) through such risk analysis helps in engineering the reliability aspect in the software at each stage of SDLC by the iteration. As the iterations progress, the risk keeps reducing and thus results in a quality HIS. Thus, incorporating a framework of ergonomics engineering knowledge in RE can be important as it tackles the safety critical aspect of the HIS.

CGSs are ancillary systems that support clinicians to carry on the day to day activities and patient care. This CGS are to work on interoperable platforms along with the existing clinical information systems like the EHRs, EMRs, inventory, billing, scheduling and resource utilization systems. The theoretical benefits of CGS can be translated into better patient care and help to the clinicians by analysing not just the requirements of the CGS but its interoperable requirements and the process flows within and with other clinical information systems. Shah, Allard, Enberg, Krishnan, Williams & Nadkarni(2012) established in their research work that notable high level requirements are defined well in many cases however, no established model has been implemented for assisting the patient management for the healthcare professionals on a day to day basis. They advocate that careful definition of system requirements to a granular level helps in better design of the software. Analysing business work flows for its extensibility, integration capability, scalability, audit ability, error recoverability with reference to the environments which the system interacts with can provide assistance to the developers of CGS. This dimension of the requirements analysis can guide the developers in effective designing and implementation strategies.
Human Centric Distributed Information Design (HCDID) includes distributed cognition and multilevel analysis of requirements. With a human centric computing as primary focus, Zhang, Patel, Johnson, Smith & Malin (2002) presenting the HCDID model in designing Electronic Medical Records (EMRs). They suggest that at multiple stages of analysis of User analysis, functional analysis, task analysis and representational analysis, different cognitive aspects are also considered. In many of the HCC designs, the human centric principles are implemented only at the representational level. However, in HCDID, this is implemented at multiple stages of analysis which helps in defining the requirements better.

Zhang, Jetley, Jones & Ray (2011) studied the generic insulin infusion pump (GIIP) model and appreciate that there are many hazardous situations in which the insulin infusion can be life threatening for the patients. So, identifying the safety requirements from such hazardous situations and their causes, developing those requirements, formalising, implementing them in the design and testing them in the SDLC is required when developing such critical products in healthcare.

2.5 Hypothesis

**Hypothesis 1: Requirements elicitation, analysis process reflects on the quality of the end product in e-health projects.**

The RE process is the initial stage of any software development life cycle. The first steps need to be set right for the complete development cycle to deliver the required final product. Within the RE process of the e-health projects, each step/ phase is equally important. The different approaches in each phase serve as a basis for RE and thus quality certification of the end product. The requirements elicitation is the crucial first step in RE process where listening to the stakeholders, understanding their needs and getting their feedback to confirm that the need or the business problems are well understood. So, this would invariably call for effective communication between the requirements engineer and the stakeholders of the e-health projects. The clarity in understanding and defining problems will further facilitate in granular analysis of the requirements which will further go a long way in developing quality e-health products.

**Hypothesis 2: Requirements Prioritisation should be done considering many important aspects associated with requirements in e-health projects.**

The requirements prioritisation in e-health should be done only after having a complete knowledge of the business value associated with the requirement, the risks associated with it and the vital dependencies like cost of the solution, sense of urgency of the requirement, the end users need of the solution etc. In agile e-health projects, the delivery stories compliment the user stories as they deliberate on the technical implications, risk associated and the time and effort estimation on the requirement. So, they should be considered as vital inputs. Further, vendor’s domain knowledge
(healthcare knowledge) is a key asset that helps in setting up collaboration between a development team and customers to understand the priority of the requirement in e-health projects.

**Hypothesis 3: Impact analysis is one of the key aspects of change management and reflects on the delivery of e-health solutions.**

The dynamic nature of the agile e-health projects results in change in requirements at any stage of the SDLC. This nature of the projects calls for effective and efficient management of the change. The impact of the change reflects on the delivery, quality of the end product and can have implications on the customer relations and the future business opportunities. So, impact analysis is one of the key aspects of change management.

These three hypotheses will be investigated and verified in this thesis through survey and interviews conducted on the software professionals working on e-health projects.
3. Survey and interview Results

3.1 Introduction

This chapter holds the data collected during the survey. The hypothesis defined in the previous chapter has lined a clear understanding in the direction of defining the questions for the survey and the interview. The questions in the survey have been designed around different aspects of RE, such as,

- the documentation of requirements
- the different aspects of the requirements that should be deliberated in the document
- importance of having confirmed requirements before starting the development
- the extent to which and when the requirements should be confirmed
- if structured RE methodology is used during the project development
- if RE methodology helps in delivering a quality output
- adverse effect of not having a RE method defined for a project
- rework in the project because of unclear requirements
- prioritisation of requirements, its impact on the quality of the deliverable
- who defines the priority of the requirements
- the factors that influence the prioritisation of requirements
requirements engineering process improvement in health IT projects

- change in requirements and the resulting rework on the project
- the impact analysis of the change in the requirements
- the factors to be considered to access the impact of rework
- the different requirements representation techniques that are used

The survey was conducted electronically through the survey monkey website. All the respondents were personally contacted over phone and then through emails with the link to the survey. Complete information of the purpose of the survey and my study was deliberated individually to all the respondents. Consolidated survey responses have been presented in graphical and tabular forms in the next section.

The privacy aspect of the information collected has also been spoken thoroughly to the respondents. The respondents were informed that they had a right to withdraw from the survey if they did not want their responses to be included in the study within two weeks of answering the survey and interview questions.

The interview questions were also around the same issues. The purpose of the interviews was to seek rationalization on the responses of the survey. The survey respondents were used in this process. This was more a triangulation effort on the data that was collected.

### 3.2 Data Collection and Outcomes

Below is the data (represented in graphs and tables) that has been collected during the survey:
Q1 Your role/roles in e-health projects (please choose multiple options if you have handled multiple roles)

Answered: 52  Skipped: 0

Figure 16: Respondents' roles in e-Health Projects

<table>
<thead>
<tr>
<th>Answer Choices</th>
<th>Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solution Architect</td>
<td>23.08%</td>
</tr>
<tr>
<td>Software Developer</td>
<td>51.92%</td>
</tr>
<tr>
<td>Database Administrator</td>
<td>9.62%</td>
</tr>
<tr>
<td>Business Analyst</td>
<td>5.77%</td>
</tr>
<tr>
<td>Test Analyst</td>
<td>17.31%</td>
</tr>
<tr>
<td>Test Manager</td>
<td>11.54%</td>
</tr>
<tr>
<td>Support Team</td>
<td>17.31%</td>
</tr>
<tr>
<td>Delivery Manager</td>
<td>3.85%</td>
</tr>
<tr>
<td>Project Manager</td>
<td>18.23%</td>
</tr>
</tbody>
</table>

Table 1: Respondents' roles in e-Health Projects
This question has multiple answers. The purpose of the question was to identify the top 5 requirements documents that are used by the software development teams during the SDLCs. There
were some people who ticked the ‘other’ option and mentioned the following other documents that are used:

- High level requirements
- Business Cases
- Code Quality
- System Analysis and Design Document
- Technical Specifications
- User Manuals

Q3 Do you believe that the requirements documents/definition serve as an input for project planning?

<table>
<thead>
<tr>
<th>Answer Choice</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very High</td>
<td>59.62%</td>
</tr>
<tr>
<td>High</td>
<td>34.62%</td>
</tr>
<tr>
<td>Medium</td>
<td>3.85%</td>
</tr>
<tr>
<td>Low</td>
<td>1.92%</td>
</tr>
<tr>
<td>Very Low</td>
<td>0%</td>
</tr>
</tbody>
</table>

Figure 18: Requirements documents - input for project planning

Table 3: Requirements documents - input for project planning
Q4 What according to you are the top 5 aspects that should be included in the requirements documents of e-health projects

Answered: 52  Skipped: 0

Figure 19: Aspects in e-health requirements documents

Table 4: Aspects in e-health requirements documents
Q5 What level of importance do you give to 'Having confirmed requirements (requirements agreed with the client) before starting development in a e-health project'?

![Bar chart showing the level of importance to confirmed requirements](image)

**Figure 20: Level of importance to confirmed requirements**

<table>
<thead>
<tr>
<th>Answer Choices</th>
<th>Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very High</td>
<td>39.34%</td>
</tr>
<tr>
<td>High</td>
<td>50%</td>
</tr>
<tr>
<td>Medium</td>
<td>13.46%</td>
</tr>
<tr>
<td>Low</td>
<td>0%</td>
</tr>
<tr>
<td>Very Low</td>
<td>0%</td>
</tr>
<tr>
<td>Not Sure</td>
<td>0%</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
</tr>
</tbody>
</table>

**Table 5: Level of importance to confirmed requirements**
Q6 According to you, to what extend (in percentage terms) should the requirements of the project be confirmed before development starts

Answered: 52  Skipped: 0

![Percentage of confirmed requirements before development starts](image)

Figure 21: Percentage of confirmed requirements before development starts

<table>
<thead>
<tr>
<th>Answer Choices</th>
<th>Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-20</td>
<td>0%</td>
</tr>
<tr>
<td>21-40</td>
<td>1.92%</td>
</tr>
<tr>
<td>41-60</td>
<td>5.77%</td>
</tr>
<tr>
<td>61-80</td>
<td>51.92%</td>
</tr>
<tr>
<td>81-100</td>
<td>40.38%</td>
</tr>
<tr>
<td>Total</td>
<td></td>
</tr>
</tbody>
</table>

Table 6: Percentage of confirmed requirements before development starts
Q7 Do you believe that practicing structured Requirements Engineering methodology will result in better planning of the project?

Figure 22: Practicing structured requirements engineering results in better planning

Table 7: Practicing structured requirements engineering results in better planning
Q8 Do you agree that quality of the software is directly related to the requirements engineering effort on the project?

Answered: 52  Skipped: 0

Figure 23: Quality of the software and RE effort

<table>
<thead>
<tr>
<th>Answer Choices</th>
<th>Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strongly Agree</td>
<td>26.92%</td>
</tr>
<tr>
<td>Agree</td>
<td>48.88%</td>
</tr>
<tr>
<td>Disagree</td>
<td>17.31%</td>
</tr>
<tr>
<td>Strongly Disagree</td>
<td>3.85%</td>
</tr>
<tr>
<td>Not Sure</td>
<td>3.85%</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>52</strong></td>
</tr>
</tbody>
</table>

Table 8: Quality of the software and RE effort
Q9 Do you agree that structured requirements engineering process leads to better delivery management of the project?

Answered: 52    Skipped: 0

Figure 24: RE process leads to better delivery management

<table>
<thead>
<tr>
<th>Answer Choices</th>
<th>Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strongly Agree</td>
<td>28.85%</td>
</tr>
<tr>
<td>Agree</td>
<td>55.77%</td>
</tr>
<tr>
<td>Disagree</td>
<td>9.62%</td>
</tr>
<tr>
<td>Strongly Disagree</td>
<td>1.92%</td>
</tr>
<tr>
<td>Not Sure</td>
<td>3.05%</td>
</tr>
<tr>
<td>Total</td>
<td></td>
</tr>
</tbody>
</table>

Table 9: RE process leads to better delivery management
Q10 *Do you believe that negative consequences of ad-hoc or less formal Requirements Engineering methods cannot be experienced until later in the software development life cycle (e.g., delivery stage, UAT)*

Answered: 52  Skipped: 0

![Bar chart showing the responses to the question.](chart.png)

Figure 25: Negative consequences of not having RE Process

<table>
<thead>
<tr>
<th>Answer Choices</th>
<th>Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very High</td>
<td>3.85%</td>
</tr>
<tr>
<td>High</td>
<td>21.15%</td>
</tr>
<tr>
<td>Medium</td>
<td>36.54%</td>
</tr>
<tr>
<td>Low</td>
<td>23.08%</td>
</tr>
<tr>
<td>Very Low</td>
<td>11.54%</td>
</tr>
<tr>
<td>Not Sure</td>
<td>3.85%</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>52</strong></td>
</tr>
</tbody>
</table>

Table 10: Negative consequences of not having RE Process
Q11 Do you believe that ‘not clearly defined requirements’ result in significant amounts of rework for the software engineers

Answered: 52  Skipped: 0

Figure 26: Rework because of unclear requirements

Table 11: Rework because of unclear requirements
Q12 Do you believe requirements must be prioritized

Answered: 52  Skipped: 0

Figure 27: Requirements must be prioritised

<table>
<thead>
<tr>
<th>Answer Choices</th>
<th>Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very Strongly</td>
<td>32.69%</td>
</tr>
<tr>
<td>Strongly</td>
<td>55.77%</td>
</tr>
<tr>
<td>Not Very Strongly</td>
<td>11.54%</td>
</tr>
<tr>
<td>Do Not Believe</td>
<td>0%</td>
</tr>
<tr>
<td>Not Sure</td>
<td>0%</td>
</tr>
<tr>
<td>Total</td>
<td>52</td>
</tr>
</tbody>
</table>

Table 12: Requirements must be prioritised
Q13 **Who according to you should be involved in requirements prioritisation**

Answered: 52  Skipped: 0

![Figure 28: Who prioritises the requirements](image)

<table>
<thead>
<tr>
<th>Answer Choices</th>
<th>Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>End Users</td>
<td>51.92%</td>
</tr>
<tr>
<td>Requirements Engineers</td>
<td>32.69%</td>
</tr>
<tr>
<td>Software Engineers</td>
<td>30.77%</td>
</tr>
<tr>
<td>Commercial Stakeholders</td>
<td>51.92%</td>
</tr>
<tr>
<td>Legal Stakeholders</td>
<td>30.77%</td>
</tr>
<tr>
<td>Project Sponsor</td>
<td>42.31%</td>
</tr>
<tr>
<td>Development Teams</td>
<td>25%</td>
</tr>
<tr>
<td>All of them</td>
<td>34.62%</td>
</tr>
</tbody>
</table>

**Table 13: Who prioritises the requirements**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Respondents</td>
<td>52</td>
</tr>
</tbody>
</table>
Q14 Do you believe that prioritising of requirements has a direct impact on the quality of delivery management of the project?

Answered: 52  Skipped: 0

![Figure 29: Prioritisation of requirements and quality of delivery management of e-health projects](image)

<table>
<thead>
<tr>
<th>Answer Choices</th>
<th>Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very High</td>
<td>19.23%</td>
</tr>
<tr>
<td>High</td>
<td>34.62%</td>
</tr>
<tr>
<td>Medium</td>
<td>30.77%</td>
</tr>
<tr>
<td>Low</td>
<td>9.62%</td>
</tr>
<tr>
<td>Very Low</td>
<td>0%</td>
</tr>
<tr>
<td>Not Sure</td>
<td>5.77%</td>
</tr>
<tr>
<td>Total</td>
<td></td>
</tr>
</tbody>
</table>

Table 14: Prioritisation of requirements and quality of delivery management of e-health projects
Q15 What are the top 5 factors that influence the requirements prioritization

Answered: 52  Skipped: 0

Figure 30: Factors influencing requirements prioritisation

Table 15: Factors influencing requirements prioritisation
Q16 Which SDLC has been used in the e-health projects you have worked on?

Answered: 52  Skipped: 0

Figure 31: SDLC in e-health projects

<table>
<thead>
<tr>
<th>Answer Choices</th>
<th>Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scrum</td>
<td>15.23%</td>
</tr>
<tr>
<td>Rapid Prototyping</td>
<td>28.85%</td>
</tr>
<tr>
<td>Evolutionary/Spiral</td>
<td>21.15%</td>
</tr>
<tr>
<td>Agile/Incremental</td>
<td>40.39%</td>
</tr>
<tr>
<td>Agile/Iterative</td>
<td>44.23%</td>
</tr>
<tr>
<td>Waterfall</td>
<td>61.54%</td>
</tr>
</tbody>
</table>

Table 16: SDLC in e-health projects
Q17 On any project you have worked, have you encountered a scenario where requirements were collected, developed and delivered to the client and the client changed the requirements leading to rework?

Answered: 52  Skipped: 0

Figure 32: Rework after requirements are developed and deployed

<table>
<thead>
<tr>
<th>Answer Choices</th>
<th>Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>82.69%</td>
</tr>
<tr>
<td>May be</td>
<td>7.92%</td>
</tr>
<tr>
<td>May not be</td>
<td>3.85%</td>
</tr>
<tr>
<td>No</td>
<td>0%</td>
</tr>
<tr>
<td>Not Sure</td>
<td>5.77%</td>
</tr>
</tbody>
</table>

Total 52

Table 17: Rework after requirements are developed and deployed.
Q18 Is impact analysis done on any e-health projects to understand the impact of the rework on the deliverables

Answered: 52  Skipped: 0

![Bar chart showing impact analysis results for e-health projects.]

**Figure 33: Impact analysis on e-health projects**

<table>
<thead>
<tr>
<th>Answer Choices</th>
<th>Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>46.15%</td>
</tr>
<tr>
<td>May be</td>
<td>21.15%</td>
</tr>
<tr>
<td>May not be</td>
<td>9.62%</td>
</tr>
<tr>
<td>No</td>
<td>5.77%</td>
</tr>
<tr>
<td>Not Sure</td>
<td>17.31%</td>
</tr>
<tr>
<td>Total</td>
<td></td>
</tr>
</tbody>
</table>

**Table 18: Impact analysis on e-health projects**
Q19 Please select top 5 factors that should be considered during analysing the impact of rework.

Answered: 52  Skipped: 0

![Bar chart showing factors to be considered when analysing the impact of rework.](chart.png)

**Figure 34: Factors to be considered when analysing the impact of rework**

<table>
<thead>
<tr>
<th>Factor</th>
<th>Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Future scalability of the change</td>
<td>40.38%</td>
</tr>
<tr>
<td>Development effort</td>
<td>88.46%</td>
</tr>
<tr>
<td>Testing effort</td>
<td>76.92%</td>
</tr>
<tr>
<td>Deployment in different environments</td>
<td>25%</td>
</tr>
<tr>
<td>Project time line</td>
<td>73.00%</td>
</tr>
<tr>
<td>Commercial implications</td>
<td>50%</td>
</tr>
<tr>
<td>Educating the customer</td>
<td>25%</td>
</tr>
<tr>
<td>Resource availability/ allocation</td>
<td>59.52%</td>
</tr>
<tr>
<td>Impact on other projects' deliverables</td>
<td>61.54%</td>
</tr>
</tbody>
</table>

Table 19: Factors to be considered when analysing the impact of rework
Q20 In your opinion, does the customer spend enough time on the requirements related activities in the e-health projects you have worked on

Answered: 52  Skipped: 0

Figure 35: Does Customer spend enough time on requirements

<table>
<thead>
<tr>
<th>Answer Choices</th>
<th>Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>7.69%</td>
</tr>
<tr>
<td>Maybe</td>
<td>42.31%</td>
</tr>
<tr>
<td>May not be</td>
<td>11.54%</td>
</tr>
<tr>
<td>No</td>
<td>32.69%</td>
</tr>
<tr>
<td>Not Sure</td>
<td>5.77%</td>
</tr>
<tr>
<td>Total</td>
<td></td>
</tr>
</tbody>
</table>

Table 20: Does Customer spend enough time on requirements
Figure 36: Requirements notation techniques in e-health projects
### Table 21: Requirements notation techniques in e-health projects

<table>
<thead>
<tr>
<th>#</th>
<th>Other (please specify)</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>N/A</td>
<td>4/10/2013 12:41 PM</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>4/14/2013 18:57 AM</td>
</tr>
<tr>
<td>3</td>
<td>Less than the 10 above</td>
<td>4/13/2013 2:55 PM</td>
</tr>
<tr>
<td>4</td>
<td>White Board Sessions</td>
<td>4/9/2013 3:57 PM</td>
</tr>
<tr>
<td>5</td>
<td>I only supported the final product</td>
<td>4/5/2013 10:13 AM</td>
</tr>
<tr>
<td>6</td>
<td>Architecture diagrams</td>
<td>4/8/2013 2:56 PM</td>
</tr>
</tbody>
</table>

### Table 22: Requirements notation techniques in e-health projects – Other options
Q22 According to you, how much of Requirements Engineering effort is done on the e-health projects you have worked on

Answered: 52  Skipped: 0

Figure 37: RE effort in e-health projects

<table>
<thead>
<tr>
<th>Answer Choices</th>
<th>Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Too much</td>
<td>3.85%</td>
</tr>
<tr>
<td>Just enough</td>
<td>42.31%</td>
</tr>
<tr>
<td>Little</td>
<td>21.15%</td>
</tr>
<tr>
<td>Too little</td>
<td>25%</td>
</tr>
<tr>
<td>Not Sure</td>
<td>7.69%</td>
</tr>
<tr>
<td>Total</td>
<td></td>
</tr>
</tbody>
</table>

Table 23: RE effort in e-health projects
Q23 Is any Requirements Management tool used for the e-health projects you have worked on?

Answered: 52  Skipped: 0

![Chart showing yes, no, and not sure responses]

Figure 38: Requirements management tools in e-health projects

<table>
<thead>
<tr>
<th>Answer Choices</th>
<th>Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>26.92% 14</td>
</tr>
<tr>
<td>No</td>
<td>48.08% 25</td>
</tr>
<tr>
<td>Not Sure</td>
<td>25% 13</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>52</strong></td>
</tr>
</tbody>
</table>

Table 24: Requirements management tools in e-health projects

Q24 Please select one option for the effort spent on each aspect of the Requirements Engineering processes in the e-health projects you have worked on
Figure 39: Ratings of different RE aspects
This chapter hosts the data that has been collected through the survey done on the respondents. These respondents are different software engineers who have been working as different/multiple roles in e-health software development projects. The different questions asked in the survey, their responses and the tabulated results are presented in this section of the thesis.

Table 25: Ratings of different RE aspects

<table>
<thead>
<tr>
<th>Aspect</th>
<th>Very Good</th>
<th>Good</th>
<th>OK</th>
<th>Bad</th>
<th>Very Bad</th>
<th>Not Sure</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correctness of Requirements</td>
<td>5.77%</td>
<td>28.62%</td>
<td>46.15%</td>
<td>9.52%</td>
<td>0%</td>
<td>11.54%</td>
<td>52</td>
</tr>
<tr>
<td>Completeness of Requirements</td>
<td>3.3%</td>
<td>15%</td>
<td>25%</td>
<td>15%</td>
<td>3.85%</td>
<td>11.54%</td>
<td>52</td>
</tr>
<tr>
<td>Requirements Prioritization</td>
<td>1.92%</td>
<td>15%</td>
<td>15%</td>
<td>15%</td>
<td>3.85%</td>
<td>11.54%</td>
<td>52</td>
</tr>
<tr>
<td>Functional Requirements Specification</td>
<td>3.85%</td>
<td>15%</td>
<td>15%</td>
<td>15%</td>
<td>3.85%</td>
<td>11.54%</td>
<td>52</td>
</tr>
<tr>
<td>Non-Functional Requirements Specification</td>
<td>1.92%</td>
<td>15%</td>
<td>15%</td>
<td>15%</td>
<td>3.85%</td>
<td>11.54%</td>
<td>52</td>
</tr>
<tr>
<td>Requirements documentation structure</td>
<td>1.92%</td>
<td>15%</td>
<td>15%</td>
<td>15%</td>
<td>3.85%</td>
<td>11.54%</td>
<td>52</td>
</tr>
<tr>
<td>Referral discovery of requirements</td>
<td>3.85%</td>
<td>15%</td>
<td>15%</td>
<td>15%</td>
<td>3.85%</td>
<td>11.54%</td>
<td>52</td>
</tr>
<tr>
<td>Requirements Engineering Process (gathering, elicitation, classification, documentation, verification and Implementation of Requirements)</td>
<td>7.06%</td>
<td>25%</td>
<td>30.77%</td>
<td>15%</td>
<td>13.46%</td>
<td>11.54%</td>
<td>52</td>
</tr>
<tr>
<td>Time spent on Requirements Engineering</td>
<td>1.92%</td>
<td>15%</td>
<td>15%</td>
<td>15%</td>
<td>3.85%</td>
<td>11.54%</td>
<td>52</td>
</tr>
</tbody>
</table>

3.3.3 Summary

This chapter hosts the data that has been collected through the survey done on the respondents. These respondents are different software engineers who have been working as different/multiple roles in e-health software development projects. The different questions asked in the survey, their responses and the tabulated results are presented in this section of the thesis.
4. Data Analysis and Discussion

4.1 Introduction

The data collected in the survey has been analysed and interpreted in this chapter. The purpose of this chapter is to see if the observations derived from responses of the survey prove the hypothesis drawn in chapter two.

4.2 Integrated Analysis (of the survey and interviews)

The respondents in this study have worked on many e-health projects. Below is the deliberation of a few of the projects.

- One of the projects is an e-Prescribing solution. There is no front end system developed. A Broker is developed which can read, perform some functions and store prescribing messages in XML formats. The design involves very complex business rules like creating new prescriptions as a prescriber, retrieving the prescriptions as prescriber or dispenser, dispensing prescriptions as a dispenser, amending the prescription by only the prescriber who has prescribed it, amending the dispensed details by the dispenser who has dispensed it earlier, cancellation of the prescription/dispense by the owner etc. A customised adapter has been designed that can transport these prescriptions through a .Net middle tier where the security authentication is done through WCF. These prescriptions then interact with the...
customised interface and scripting services from where it is stored in the database. This has been now implemented in two countries.

- In one of the countries, it has been customised to further register this prescription entry to a hospital database and extended in the PCEHR system. An industry accepted HL7 international healthcare informatics interoperability standard has been followed to save the records as CDA document. This has been developed abiding to the NEHTA.
- Another project is where two different e-prescribing systems interact with each other. This is now facilitating the prescribers of one e-prescribing system to view items that have been dispensed by clinical information systems of other e-prescribing system.
- Projects where a web based UI platform is developed to access the disability needs of the patients. Considerable numbers of web services are used to facilitate the clinical staff to get the information required to access the disability needs of the patients.
- Another billing platform for the ministry of health is also being developed which facilitates the ministry to pay the pharmacies for registering the patients for different types of long term conditions

The 3 major roles in the survey respondents were developers, architects and the project managers as they are on the front line for ensuring development of solutions. The other important ones were the analysts who do business analysis and testing.

4.2.1 Requirements documents:

Among all the respondents, about 94.24% very highly or highly believe that defined requirements documents serve as an input for planning the project. Further analysis by role has suggested that all project managers, delivery managers, test managers, business analysts, test analysts completely agree with this. As these are the roles responsible for majority of the planning in the project, it is confirming that

**Observation: requirements documents serve as an input for planning in e-health projects**

Different team members look at different documents at different stages of the SDLC. The responses have shown that the solutions architects refer to the functional requirements, business requirements, technical requirements, non-functional requirements and system requirements. These documents help in further analysis and design of the solution for these architects. On the other hand, the project managers would refer to the project scope, system requirements, business requirements, functional requirements, and technical/non-functional requirements documents. The interview responses have justified that these selection are used as they help the project managers to plan their deliverables and define the project plan. The results have shown that the business analysts also work with the similar perception as the project managers. They prefer to refer to the project scope, business requirements,
user requirements, and system requirements as their interaction with the client is more. Further, these documents help in future enhancements and change requests scoping apart from serving as instructive tools for knowledge transfer (through workshops) to the software developers/ testers etc by the business analysts. The delivery managers’ activities of the project revolve around organising the deployment of the deliverables in different environments and preparing the release notes and the transmittal reports. So, their responses in the survey and interviews have suggested that the documents they refer to are the system requirements, functional requirements and technical requirements. The study has shown that database administrators prefer to refer to the technical requirements, functional requirements, project scope and the user requirements. Further deliberation with them has shown that the database design is done after thorough understanding of the reporting requirements, functional requirements of the users. The database design is to ensure that the project is within the scope of the project with future agreed scalability. It has been observed in the responses of the survey that the software developers refer to functional, technical, business requirements and use cases in the order specified. However, they also refer to the user, system requirements and project scope. That led to an understanding that the developers refer to maximum number of types of requirements documents. However, the interview responses showed that majority of the work is done by talking to the solution designers/architects, tech leads, test analysts and peers and not really looking/referring to any requirements documents. A few of the developers claimed that they work with the definitions they get in the emails from the tech leads/architects.

**Observation:** Software developers do not refer to e-health requirements documents but rely on the deliberations with the tech leads, architects, analysts and other peers for understanding the requirements.

The testing team design, execute and manage the testing activities based on the requirements documentation used in the project so they would prefer to refer to the system requirements, functional requirements, user requirements, business requirements, technical requirements and use cases. The support team has rated technical requirements, functional and non-functional requirements. This was justified in the interviews saying that the support activity is on the current functionality in the production version of the software and so rest other requirements do not mean much.

Interview responses showed that in the current projects the teams are working in, iterative SDLC is being followed. The Agile approach taken in these projects is to do the requirements for each required functionality. There is no particular documentation template used. The requirements are done as 2 or 3 page documents. The doing of user stories is not given weight-age. However, the previous e-health projects that the teams have worked on did have the user requirements, functional requirements, and business requirements defined and approved with the customers. The 2 or 3 page requirements in the current project go through a communication channel and get approved by the
customer before the development work starts. The overall responses reflects that the top 5 requirements documents that the software engineers refer to during the SDLC are

1. Functional requirements
2. Technical requirements
3. Business requirements
4. System requirements and
5. Non-functional requirements

There were a few other documents that were mentioned in the ‘Other’ option of the question. They were the

- High level requirements
- Business cases
- SAD document
- Technical Specifications
- User Manuals

The purpose of these documents has been deliberated in the literature review. To verify this aspect for e-health projects, the responses to the survey question number 4 has been analysed to see that in general, the software engineers look for the following top 5 aspects:

1. Functional description,
2. Process flow/workflow of the solution,
3. Definition of client expectation
4. Specifications of how the system should work
5. Scope and future scalability of the solution

Further analysis of this result showed that all roles individually had similar outlook for the above aspects. Except for the solution designer who instead of scope and future scalability, wanted to see the explanation of why the product is needed. The interview responses to this showed that this was because; the explanation of why the e-health application is needed will help them to design the solution for the business problem. The other role who had a little deviation in opinion from the above was the business analysts. They argued that for the e-health projects, graphical mock-ups and definition of the user expectations is required as with e-health projects. They suggest that there are many aspects like the patient consent model, the privacy of data model, the medical information of the patient etc are to be integrated and incorporated within these e-health solution and a graphical representation is the better way of explaining all this. This helps the business analysts to analyse the problem and the requirement better and deliberate the same with the business stakeholders and the
development teams. The database administrators had similar preferences but had rated the specifications of how the system should work. The justification from the analysis of the collated data is that the different consent models, privacy models, medication information access by the roles can be incorporated in the database design through the specifications details. This is justified as this is the basis on which the database design is done. The project managers and the delivery managers also had similar preference but had rated scope of the project to be the top most aspect. They argue that as majority of the e-health projects are funded by the ministry of health and there is always a crunch in the budget, interference of ego hassles of some stakeholders and different political dynamics involved etc, planning and implementation should be based on the scope of the e-health project.

Another interview response suggested that there are certain non-tangible aspects of the projects that should also be included in the document like the e-health intricate processes and the core expectations of the end user. This can be best documented by the requirements engineers (possibly a business analyst) here. The stakeholders drive and push the priorities. However the solution is finally used by the end users so, users interpretation and expectations of the entire process has to be documented. The requirements engineer’s role is very crucial here and he/she is the one who can drive this. If done well, the developers can rely on the documentation and can assist them to create what is required

All these aspects logically lead to an argument that having confirmed requirements and documenting the requirements before starting the development and implementation phases of the e-health projects is a safer approach. Any ambiguity of the requirements may have to go through a churning process through layers of bureaucracy in the ministry of health and other normal project management processes. The survey results show that about 86.54% of the respondents believe that it is very highly or highly important that the requirements have to be agreed and confirmed by the client before the development starts in the e-health projects. The rest of the respondents gave it a medium scale rating. There have been no respondents who have rated low or very low for this.

**Observation: It is required to have high level requirements confirmed before starting development in e-health projects**

However, the survey results have shown that percentage of detail of each requirement that should be confirmed varies for different respondents. 40.38% of the respondents feel that about 81-100% requirements should be confirmed, but 51.92% of them feel that about 61-80% confirmed requirements are good enough before development starts in e-health projects. It is ideal to have the requirements confirmed before starting development. This is possible in waterfall model. However, the incorporation of change will not be possible in such cases. So, in real time, confirming the requirements 100% is not possible before development starts. Nevertheless, having requirements confirmed to a certain extent before the development starts can ease the pressure out of the development team. It is interesting to know that none of the respondents would expect the confirmed
requirements to be less than 20% of the total solution. Interview responses showed that this is because the analysis and design of the solution cannot be done with such less confirmed requirements. The interview responses suggested that having confirmed requirements is good but in majority of the real time situations this is not the case because the requirements phase is usually rushed through. This could be because of an urgency to win the contract or kick off the project or an urgency to finish the project within a stipulated timeline or during initial negotiations, the sales pitch would have been to deliver the solution at a particular time however, by the time the customers accepts it sometime would have been lost and the customer would still expect it to be delivered at the original negotiated timeline. This is a result of not having a good governance strategy over this phase by the leadership team.

4.2.2 Structured RE Process:
SWEBOK extensively describes software requirements knowledge areas (IEEE Computer Society Professional Practices Committee, 2004) and IEEE has defined international standards for RE (ISO/IEC/IEEE 29148) (IEEE, 2011). These standards are in place advocating the need for structured requirements methodology. The survey results has shown that 73.08% of the respondents very highly or highly believe that practising structured RE methodology will result in better planning of the project. The people who rated most were the project managers, test managers, delivery managers, business analysts and the test analysts. There are a few who rated this as medium (15.38%), low (3.85%) and very low (1.92%). The people who rated as medium, low, very low were the support staff, database administrators and test analysts. The interview responses have suggested that the ones who have rated it to be medium and low are the ones who are really not involved in the planning of the project. The analysts thought that this gives a structured work pattern for them. The architects justify this rating by saying it will make sure that the requirements engineers will not miss any requirements and helps them to define optimised requirements which will then flow into the designing of the solution. The project managers think it is a good to have structured process however; expressed apprehension saying may result in extra cost for the project in small and medium sized software vendor companies.

The study has further indicated that in e-health projects, where many requirements intricacies are to be considered for the development of the solution, the quality of the software is directly related to the RE effort on the project. The results of the survey substantiate this understanding. About 75% of the respondents strongly agree or agree that RE effort reflects on the quality of the solution (output). The majority of the strongly agreeing ones were the analysts. This justifies their job. However, it was interesting to know that the software developers and the architects also agree with this. There was good percentage of 17.31% disagreeing and another 3.85% strongly disagreeing to this. They included a few project managers, solutions architects who believed that the quality parameter is not
just about RE. They believe that apart from following structured RE process following standards of coding, understanding the requirements from the user’s perceptive, following some best practises were all required for ensuring better quality of the solution.

Another indication of the study was that about 84.62% either strong agree or agree that structured RE practices leads to better delivery management of the projects. The project managers’ argument was that the structured approach will lead to better understanding of different aspects of the e-health projects. It helps to identify the release candidates for the iteration and the different aspects that need to be considered for the delivery management of the iteration. To add to this, the delivery managers suggested that a structured RE process reduces the ambiguity of the release candidates and clarifies not just about the deliverable but also about the release environments and the different aspects that needs to be considered and taken care of while releasing/deploying the candidates into the different environments like the internal test environment, UAT and production environments. This surely leads to better delivery management of the e-health projects. One of the interview responses to this by the project managers was that RE process surely helps to plan the delivery better but does not necessarily mean that this leads to better delivery management. The argument was that this surely enhances and helps in better delivery management. However, the delivery management within itself needs to be planned organised and executed and just the RE process will not lead to better delivery management. There was a small percentage of 1.92% strongly disagreed and a 9.62% disagreed to this. They were the once involved in the delivery management like the delivery managers, the support guys and some developers who were involved in knowledge transfer to the operations team. Their argument was that delivery management is a specialised area where managing things around the delivery is handled and that RE process will help the developers and the others to make the deliverable ready.

**Observation:** Structured RE practises helps in enhancing planning, delivery management and the quality of the e-health projects.

Majority of them (36.54%) agreed on a medium scale that negative consequences of ad-hoc or less formal RE methods cannot be experienced until later in the software development life cycle. Majority of them were analysts. This is a very interesting point to note as this is directly related to their area of work. Some project managers also were agreeing on this scale. Majority of them who rated between medium and low were the software developers as their thought process is always inclined towards technical aspect of the project (coding, environment setup etc.). The interview responses suggested that if within the high-level requirements, the granular requirements were missed in the RE process because of not adhering to formal standards and RE process models, such gaps would be identified in the reviews, development, system testing and integration testing of all the components. Contradictory to this, there were about 25% of respondents who thought about it on a very high/high scale. These were the respondents from the support team, the database administrators, the solutions architects and
the projects managers. It has shown in the interviews that the majority of support call and the tickets that are generated are because of these requirements gaps which are not identified at the development stage. However, a good 34.62% of them rated it on a low and very low score. This fraction of the respondents included every role except for the business analysts. This part of the survey and interviews resulted in very interesting results. The consolidated results of ‘very high’ and ‘high’ scale were 25%, medium scale was 36.54% and low and very low scale was 34.62%. The very high and high scale percentage of the results cannot be ignored as these are the ones who experience the after effects of the negative consequences of ad-hoc or less formal RE methods after development complete and they were about quarter the size of the respondents.

According to Gous (2013), theoretically, it is efficient and perfect to develop software with minimum RE and have no rework. The survey and interviews have shown that 98.08% of the respondents believed that not clearly defined requirements result in significant amounts of rework for the software engineers. A mere 1.92% not very strongly believed this. On investigation of the survey results, the people who answered this were the support and maintenance team who do not have much to do with the rework/ development at the first place. This authenticates the research work already done by many on this aspect (Kumar & Kumar (2011a); Kumar & Kumar (2011b); Kulk & Verhoef, (2008); Phillip (2010); Ebert (2010); Jeremy (2010); Mohamed & Armin (2010).

4.2.3 Prioritisation of requirements:
88.46% either very strongly/ strongly agreed that the requirements should be prioritised. It was intriguing to know that about 11.54% not strongly agreed. On investigation, it came to light that these were the delivery and support teams and some programmers. The programmers came up with the justification that this is not their arena of work as they get to the stage of development only after all these processes have been passed through.

The top 5 people that the respondents thought should be involved in requirements prioritisation were:

- End-users and commercial stakeholders
- Project sponsors
- Requirements engineers
- Legal stakeholders and software engineers like the architects, analysts and the project managers

Some suggested that the development teams as a whole should be involved as well. About 34.62% suggested that all of them should be involved in prioritising the requirements. This is ideal. However, in the interviews, it was deliberated that considering every stakeholder’s input would lead
to a lot of confusion and is not realistic. The developer’s input is not relevant to the commercial, legal stakeholders and the end users. This will unnecessarily increase the cost of prioritisation task. The project manager’s and the analysts think that it is realistic to take the input of the end users, the commercial and legal stakeholders and the prioritisation should be achieved collaboratively. The developers should be surely involved during the estimations and not for the prioritisation.

About 53.85% of the respondents very highly/highly believed and about 30.77% believed on a medium scale that the prioritization of requirements has a direct impact on the quality of delivery management. Majority of them were the project managers, the deployment managers, architects and analysts. The argument was that prioritisation of requirements will give a good understanding as to what is going in the release, which components are affected, which components should be tested, defining the scope of regression testing will be clearer and with respect to the delivery, the deployment manager can accordingly plan as to which components should be delivered, what environments he should be planning delivery on and such other features. Apart from this it can facilitate resource planning as per the technical requirements i.e., the manager can book and allocate different resources with specific skill sets to deliver the requirement. For example, on an e-health project, if there are specific requirements to develop an InfoPath form that collects the data and processes it through the business rules engine and stores it in the database for a healthcare practitioner as new feature in the e-health project, then the resource planning can be comprehensible for the manager to book the InfoPath Specialist, the Database Administrator, technical analyst who can interpret the business requirement into technical terms and the technical testers who can navigate into all these different components to test in different required environments. Further, it also helps in scheduling the time of resources, deployment, UAT etc. for the managers and estimating the amount of effort required by the software engineers.

The survey results show that the top 5 factors that influence the requirements prioritization are:

- Customer’s need/wants
- The requirement’s influence on customer’s deliverables in the business
- Technical feasibility
- Customer’s business dependencies
- Availability of resources

Some did identify that customer’s intuition about the requirements is also important. However, the respondents who disagreed to this said that the top 5 are more important as this factor is only about customer’s intuition and the customer may not have the ability to define his intuition in practical/ real/ technical terms. One of the instances from the interview responses deliberated the following:
Currently, in one of the DHBs in New Zealand, the administrative staffs in the hospitals takes a laptop to each patient’s bed to confirm the personal, ethnic and demographic profile of the patients. The requirement is have the same application on handheld devices and access the clinical information system through a customised application to facilitate the administration staff to do their work going to each patient. Going by the customer’s (end-user’s) intuition, this is very important. But the DHB and the ministry do not want to invest in this at this stage as it directly does not enhance the patient care or facilitate the medical staff to provide better service through this. According to them, it is a nice to have feature. So, considering the customer’s intuition may not always result in prioritising the requirements appropriately.

In e-health projects, the privacy of the patient is a critical factor. This completely revolves round the encryption and decryption of data and the flow of information and accessibility requirements of such data. So, apart from the functional requirements, the non-functional requirements should also be considered in the architecture of such a solution. So, the priority of the other functional requirements takes a back seat when prioritising these requirements.

Apart from this, the estimates given by the development team also plays a major role. They determine the cost of the change in financial and commercial terms and in majority of the cases, the customer decides the priority on the requirements based on this factor too.

**Observation:** Appropriate prioritisation of requirements after considering many dimensions and aspects of healthcare has a direct impact on the quality of the delivery in e-health projects

This proves hypothesis 2.

The SDLCs the respondents where working on were in the order of rating:

- Waterfall
- Agile/iterative
- Agile/incremental
- Rapid prototyping
- Evolutionary/Spiral
- Scrum

Interview responses suggested that depending on the kind of the project and the deliverable requirement of the customer, the SDLC of the e-health project is decided. Waterfall was majority of the times used when

1. There was an implementation of COTS software and the respondents were project managing
2. Doing the requirements following feasibility study of a project
3. Where hardware projects for e-health projects were implemented like setting up the network for a hospital, setting up a clinical base for remote sensing medical devices

Agile/iterative approach was used when the delivery was in iterations of iteration per 2-3 months. Agile incremental where software is delivered as a service (example: child health platform, LTC, Socrates, clean hands) where the delivery is of range from 2 weeks to once in a year. Every incremental deployment included enhanced feature for an existing functionality like developing a ubiquitous computing feature for an already existing application. Rapid prototyping is used for a few projects where the platform prototype is used as a sales pitch for new business opportunities. Example: after finishing an iterative project with a hospital billing system, an enhancement of existing feature with new technology or a new functionality is deployed as a product prototype for the sales team to negotiate the project. The development of such prototypes can be used as a product that can integrate with many such billing required organisations like the DHBs, the ministry of health etc.

Evolutionary/spiral SDLC model is used where the functionality evolved and was used for many projects like the adapters for information exchange brokers where the messages are exchanged between the prescribing and dispensing entities. This kind of applications evolve over a period of time on usage and specific requirement like starting from prescribing, dispensing, to inactivating, cancelling prescriptions to deferring the prescriptions etc. This evolutionary model works best for projects where there is continuous collaboration and co-ordination between the e-health software vendors and e-health software consumers. This can possibly run for years together. Another example is the e-health insurance company and its e-health software vendors where the requirements ever evolve and ideally take a spiral model. Scrum model is used where the updates are needed in regular intervals.

4.2.4 Managing changes in requirements

About 82.69% of the respondents have experienced scenarios where requirements were collated, developed and delivered to the client and the client changed the requirements leading to rework. Another 7.69% did answer it as ‘may be. The interview responses imply that they were not ready to admit the scenario because they did not want to complain about it. This consolidates that about 90.38% of respondents agree that changes were made after the delivery of the product.

Accommodating change of the requirements in the projects is the essence of the agile software development models. The change will always have an impact on the solution and will end up in rework of many factors in the software development (Al-Durra, 2009; Nikitina & Kajko-Mattson, 2011). Analysing the impact is one of the key needs of managing the change in requirements of any software project (including e-health projects).
About 46.15% agreed that impact analysis is done as a practise on e-health projects to understand and estimate the impact of change and the rework. Another 21.15% seconded it (by rating ‘may be’). There were a few who ranked it as ‘may not be’ and ‘no’. They consolidated to about 15.39%. Another 17.31% were ‘not sure’. The interview responses suggest that the impact analysis is the arena handled by the business analysts, solutions architects and the project managers. That is why about 32.70% selected either May not be, No and Not sure. The project managers’ view was that like prioritisation, the impact analysis is also handled by the analysis and design team and flows into the other teams during estimates and so would possibly appear to the rest of the team like it is a new requirement. Talking to a few respondents revealed that for the experience they have had in the industry, this is how change is handled in the e-health software industry. They suggest that more looking at who should assess the impact of the change, the factors that should be considered to analyse the impact is more important.

The survey respondents reckon that the 5 factors that should be considered during analysing the impact of rework on the e-health project because of the change in requirements are:

- Development effort
- Testing effort
- Project time line
- Impact on other projects’ deliverables
- Resources availability/allocation

This is merely from the software engineers’ point of view. However, the other factors that should be considered are the impact of not delivering on the goodwill of the vendor, the future business options with the customers and many such other factors. This was not assessed in the survey. This is one of the drawbacks of survey. The interviews suggest that having change owners on both ends (customers and the vendors) of the project who cannot just understand the functional impact and business impacts (with reference to overheads, timelines, business processes, system maintenance etc) but the technical impact of the change is also required for analysing the impact of rework. This helps to give a better understanding of the impact to all the stakeholders involved.

The overall impact of the change on all the SDLC phases has to be assessed. Currently the estimates are done by consulting the developers, testers and the solutions designers. There is no formal process on which this impact analysis is driven.

As seen before in chapter two, RE is a collaborative effort of the customers and the development team. For the software vendors, their work revolves only around the development effort. So they give 100% time for understanding the requirements. However, the time given by the customer towards the requirements effort is questionable. In the survey about only 7.69% agree that the
customer spends enough time on the requirements related activities in the e-health projects. About 42.31% rated it as ‘may be’. So about 50% of the respondents were on the positive side of it and the rest 50% of the respondents rated it between ‘May not be’, ‘No’ and ‘Not Sure’. But when analysed between the yes and no responses, only 7.69% selected yes and about 32.69% said no. Further investigation showed that there were solutions architects, project managers, test managers and software developers who answered ‘No’. As these are the roles that are affected directly with this activity, I can observe that

**Observation:** The customer did not spend enough time on the requirements related activities in the e-health projects that the respondents have worked on.

### 4.2.5 Requirements notation techniques:

There are many business analysis and requirements notation techniques currently being used in the market. Cadle, Paul and Turner (2010) have deliberated on 72 essential business analysis techniques for success. Implementing all of them in a project is not practically possible. The survey results suggest that within the e-health projects that the respondents have been working, the top 10 requirements representation techniques are:

- Flow charts (as they serve as inputs for all activities in SDLC and for UAT to the customer)
- Use cases and data flow diagrams (for the testing team and the end users)
- Use case diagrams (for the architects and the testing team)
- Sequence diagrams (for the solutions architects, test analysts, UAT staff)
- Entity relationship diagrams (for tester and DBAs)
- Informal drawing (for the Business Analysts, Enterprise Architects, Solutions Architects, Project Managers and Test Analysts)
- Natural language documentation (for all roles in the survey)
- Informal text (for all roles in the survey)
- BPML/UML (these are more a technical notation and so has a possibility of restricted usage) and user stories
- Decision tables (for the Analysts, Test Analysts, Business Analysts and for UAT)

There were more suggestions like the white board sessions and the architectural diagrams. It was also seen that about 53.85% opted for user stories. Given that agile projects is where change is accommodated and user stories are the way requirements are represented in agile projects as per the literature review in chapter 2, this was further investigated in the interviews. It was observed in the interviews that there is no defined way for the organisation being surveyed to record the user stories. Each requirement is documented with all the assumptions and preconditions and the general functionality adding a few diagrams depending on what is required. The user stories are not developed at all. It has been observed that user requirements is documented as a natural language document after talking to client and having a series of email exchanges with the client/stakeholders about the
requirement. Once the requirements is communicated and confirmed with the client, the solutions architect works on defining the solution and created tasks for the software developers to work on. Visual Studio TFS 2012 is used for creating these entries.

Observation: User Stories are not developed in the RE process in the organisation (e-health software vendor) where the study is being done for agile (iterative) process

4.2.6 Requirements Management:

About 42.31% of the respondents of the survey think that just enough RE effort is done on the e-health projects that they have worked on. However, about 46.15% believe that little/too little work is done which is the reason so many instances of re-work is arising as seen previously. The anticipation of these people is that if good required amount of effort is done on RE and all requirements are identified at the initial stages, the rework will not be as much as seen in the projects they have worked on.

The study has shown that on an overall scale of ‘very good’, ‘good’, ‘ok’, ‘bad’, ‘very bad’ and ‘not sure’, the respondents think that correctness of requirements, clarity of requirements, prioritisation of requirements, functional specifications, non-functional specifications, requirements document structure, early discovery of requirements, time spent on RE and on the whole the different aspects of RE (gathering, elicitation, classification, documentation, verification and validation and implementation) are rated ok (consolidated highest rating). This triangulates the below observation.

Observation: Requirements are not being managed efficiently within the development team in a manner that is compatible to the development tool that is being used.

Further, the respondents have rated equally for ‘ok’ and ‘bad’ against the completeness of requirements.

In my observation, within the different phases of the RE process, the elicitation, analysis, documentation, verification and validation of the requirements activities seem to be working as required. This is because, the stakeholders are that from the customer side and the system analysis and design team in the vendor company. However, when it comes to managing the requirements, it is not just the managing to ensure that they have been communicated to the client and got them approved, but should also be communicated effectively with the others in the team throughout the SDLC so that the requirements are not missed, slipped and lost during SDLC and are seen as issues in UAT. This is currently the issue with the requirements management in the organisation under study.

In Visual studio 2012 TFS, a customised requirements work item is currently being used. Once the requirements are confirmed with the client, the requirements work item is created for that requirement. The tasks (another work item) are created to those requirements but are not linked to the
requirements. These tasks are assigned to the developers and when developed are set to a status of ‘ready for testing’. The testing team uses ‘Microsoft test manager’ which comes with the Visual Studio 2012 to create the test plan and does test suites and test cases for testing the release from the requirements document. When there are any bugs created but are not linked to test cases or the tasks. So at every stage, the traceability of the requirements is not being monitored because the requirements management process is not in place. This is resulting in slipping of the requirements in the SDLC and not being able to be picked up by the development team. On further investigation and understanding of the TFS, it has been observed that this option to traceability is possible in TFS but is not being currently used.

About 48.08% of the respondents think that the requirements management tool is not being used for the e-health projects that they have worked on. Currently, it is the requirements are drafted in word documents and are managed thorough spread sheets. The 26.92% who says ‘yes’ to this in question 23 where referring to these Microsoft office tools like the Word, Excel and Visio. These are surely most efficient if used as required for managing the requirements.

The interviews with the respondents have revealed that in the past on some e-health projects, there have been no business analysts on the project to own and manage the requirements. It was being managed collectively between the project managers, solutions architects and the test analysts. In such projects, whenever there were questions about the requirements for the development and the testing team they had to go to the solutions architect or the project manager and in turn the customer was contacted. In projects where there is no single point of contact for the project from the customers’ end, and no business analysts on the project, the ownership and the responsibility associated with it for the requirements process management has thus been compromised and this has had some adverse effect on the project. Lack of resources and some political issues and some project management issues have been quoted in the interviews by a few respondents.

**Observation:** Not engaging Business Analysts/ Requirements Engineers on the e-health projects leads to chaos as there is no ownership and responsibility exhibited on the complete RE process

In one of the projects, where an information exchange system was to be built, a 3rd party consultant - Business Analyst was appointed by the customer. They had done the business requirements document. This document suggested that the information exchange broker should do a certain tasks during the exchange of clinical messages at a certain component level within the broker. This was more a technical architecture that was defined in the business requirements document. As this came from the customer, the vendor designed the broker as defined. Once it was implemented, the customer come up with the notation that the feature should have better been implemented not at the component level but at the end points
from where the message is originating. The customer was relying on the 3rd party to do the requirements. The vendors were just following them and the 3rd party consultants were never in the scheme of things during implementation. So there was no responsibility vested on any of the 3 stakeholders when this particular broker was built. This was a very interesting point to note. All this showed that the organisation under study was not proactive enough to ask questions and take leadership to control the requirements.

### 4.3 My observation from data collected in the interviews

Like any other software product, for e-health packages too “Requirements” are the fundamental starting point for delivery of the end product (software). Time and money have to be invested to understanding what is being designed and developed and why it is being developed. This is the basic waterfall paradigm. Requirements define the scope on which the time and budget of the project are assessed. So requirements are imperative. Apart from what has been observed in the previous section of this chapter, change in requirements and traceability of the requirements are key aspects that need to be taken care of in e-health projects.

Agile model anticipates changes. However, agility in any project does not negate the fact that requirements (engineering) are still needed. One of the interview responses suggest that probable domination of by the sales force, no business analyst on the project or no proper business analysis on the use cases (user stories) or something similar could be the reasons for the pandemonium in agile projects. “Agile” does not mean ‘no’ requirements. On the contrary agility in the projects means continuous requirements. From the SDLC perspective and the project management perspective, the cost of requirements is distributed in agile methodology over the iterations as against waterfall. Agile projects run longer because the customers get used to iterative change in the requirements and continue to introduce change. This could lead to an elongated scope. This better side of incorporation of change in requirements is for the customer to end up with better targeted outcome. Change in requirements can be optimised by having appropriate responsibility on both the client side and the development team. Having appropriate change owners on both sides (i.e., the customers and the vendors) who can understand not just the functional impact, business impact (with reference to overheads, timelines, business processes, systems maintenance etc) but also the technical impact can better help managing the requirements and its changes. On the both ends, the requirements and the change owners should be able to deliberate how the change impacts the business and how it translates into systems. If a blind faith is vested and the requirements are not understood by any of the owners from the both ends, the resistance in implementing them in the deliverable can creep in. The pandemonium can end up being a struggle to both the parties. This calls for the need of the requirements and its change and the impact of the change to be driven by client side awareness of all
the above mentioned aspects. Further, this should be overlapping into the vendor side as to what is being engineered and meticulous managed within the iteration.

The starting point of RE precedes project management process. However, a change in requirement is a change in the scope of the project and so should be managed by through the project management discipline. There is always a risk of having gap between what is being delivered to what the client really wants. This gap is because of lack of technical comprehension and blind faith by the customer. This may lead to cohesion at the solution is presented to the user. The change management can be a part of project management but the driver of the change is always the requirements engineer/analyst.

In one of the e-health projects that has been examined as part of the study, the project started with an Agile approach where the deliverables are to be made in iterations. The seniors in the project management team agreed on the requirements that are to be delivered and are handed them to the team through the workshops for the tech lead, the developers and the test lead. The developers developed the requirements and the testers did the test design. The progress of the team was tracked through the morning catch ups. There was some miscommunication about one of the requirements and was flown into development and testing phase without the customer’s knowledge. When this requirement featured in the solution and was presented to the client, it came as an unpleasant surprise to them. As a correction process, the project management process was revised and the Kanban board was introduced. The daily sessions were cut down and the using of it did not have much of an impact on the project later. The project manager has been changed and the project team neither does the everyday catch up nor the Kanban board but uses another technique now. Currently, all the requirements have been identified on the board for the iteration and the junior project manager has talked to the team members to get the daily updates while they are working.

With the Kanban approach, the idea is to get the items out of the door. These kinds of changes do have a ripple effect on the development team and they do feel overwhelmed with such changes. This management approach needs to be more robust and rigorous while incorporating such changes in the projects. Irrespective of the approach and the dimension of the project, it is the delivery of the requirements that is at stake in situations like this.

In e-health projects, changes need not be just functional. They can be changes in the policy/strategy, privacy of the patient information, legal communication channels or the legislation. So, requirements engineers are not to just analyse the subjective aspect of the requirement/change but should also be having a multi-dimensional outlook when doing the requirements of e-health projects. So impact analysis is the key to manage the changes in requirements for e-health projects. **This proves hypothesis 3.**
RT is best achieved in linear project models. In the agile e-health projects that have been observed, RT has been cumbersome. This is because there is no process in place for traceability and there is no specific time allocation done for the same in the project activities. The crux and the process of the traceability gets very ambiguous when people do not understand what it is they are achieving and why they are achieving it. The existing TFS is having the traceability functionality that is not being explored and utilised on these projects. RT in e-health projects can be best achieved through using of this existing tracing tool.

A few respondents argue that the industry is now taking awareness on this by emphasising to understand

- what the client wants
- drill down to translate that into what they need
- break that down into system level or technical level
- generate one liner requirements
- generate acceptance and exit criteria
- generating tasks and working on it and
- generating test cases against the requirements and testing against them
- deliver and maintain them in the future through support contracts

This management of requirements ensures that the projects are to the far end and delivers as required. The light weighted documentation helps in granulizing the requirements for each iteration in agile. The key is nailing down the ‘what’ for the client. Agile aptitude and attitude should be carried out in every aspect of the project. The light weighted-ness should not compromise in the client understanding the technical and business implications of the project. If a client is not engaged with the project and technical clarity is not established then the project is not driven by sense of reality which leads to disconnection between the client and the vendors. So, part of the presales effort of the project (requirements gathering) is to ensure that the client understand the functional and technical implication. The client expectation is to be set by defining the requirement right and effectively communicating it. If the requirements engineer does not perceive this aspect in agile projects where change of requirements is seen for every iteration, then the vendor’s perception will be not as per the required requirement but will follow only the perceived requirement. Thus leading to poor requirements gathering, eliciting, analysing, documenting and poor management of requirements in agile projects leads to disasters in the project. So the right perception and definition should be communicated within the vendor’s team by the requirements engineer and this perception should be quoted to the client and educate them. Such disconnect if not addressed earlier, could lead to disasters. On the contrary, if done earlier can result in better quality of the end product and customer satisfaction/delight. **This proves hypothesis 1.**
In all the e-health projects that I have included in my study, are very profound on integration with other existing e-health systems. In all these projects, these integration requirements are very inadequately addressed. In these cases, the end to end working of the systems is testable only in UAT. So unless a mock up or prototype is presented to the client in the UAT environment and the client is involved, the issues and the relevant requirements are not known. On investigation, it was seen that one side of the issue is not doing these requirements but the main reason is that the requirements engineers have not been able to spot these requirements and define them at granular level. In one of the e-health projects, were the solutions architect did realise this and get the team working on this but as there was no documentation properly done, it still have issues as this piece was worked upon the data collected on email exchanges. Another issue is that the client not giving time and sharing information about these requirements also causes these kinds of issues.

4.4 The synthesized methodology

As seen in the literature review section of this thesis, e-Health projects cannot be approached with an ordinary software development viewpoint. A mind shift from software development to software engineering is required for e-health vendors. Gathering requirements is never an issue in the organisation.

Anticipating the future use, ensuring that the new e-health product or software services is commercially viable to vendor and ensuring that the customer is happy with the current solution should be the main vision of the RE process of e-health vendors. This will ensure that the change in requirements can be embedded and addressed at any stage of the project.

Further from here, the small to medium scale e-health vendor that has been studied and observed will be addressed as the e-health vendor for the practical convenience of expressing the solution.

There is a three dimensional approach in defining this solution. The concept of requirements road map from one research and the different factors of requirements planning from another discussion are being combined and synthesized with the concept of managing the requirements using Microsoft Visual Studio Team Foundation Server 2012 in e-health projects has been presented in this chapter.

4.4.1 Requirements Road Map

Strategically defining requirements road map with the following three dimensions embedded can be very beneficial for the organisation:

1. The road map to understand the customer’s problem and there by defining the requirements of the client
2. The generic e-health industry requirements (like feasibility for interoperability of systems, following industry standards, government policy directives etc)
3. Internal principles and requirements like not compromising on the technology, quality and future scalability

Adapting requirements road map for the organisation will ensure the solutions provided will not be just for functional solutions but will abide by all requirements for it to be a commercial model (Gottesdiener, 2007).
Requirements Engineering Process Improvement in Health IT Projects

Figure 40: Proposed Requirements Road Map
For instance, the e-prescribing solution is built as a platform and supported by the organisation and can generate further business opportunities by trading it as a service based platform or deliver a platform where the software services can be reused by the client and the e-health vendor can continuously earn revenue on the same.

The purpose of such engineering model is to define a few requirements and implement them as a functional slab that can co-exist and can further be disintegrated to be used independently. One such example could be to develop an InfoPath form where the user has different components within the form and can select the ones that are required for a particular functionality like the customisable user interfaces. This kind of engineering model necessitates the requirements to be properly defined.

### 4.4.2 Requirements Governance Team

The above approach calls for a requirements governance body that can control and monitor these factors for every e-health project. Ideally, members of this body should be from the strategic governance team of the e-health vendor, project manager and stakeholders from the customers.

This ensures that change will be incorporated from the software engineering perspective and not just from the project management perspective to deliver the future scalability of the product. For instance, in one of the projects, where all e-prescribing transactions are being tracked in the database, the client has seen that the tracking database need not store any details of the prescription but should contain only the registration ID of each prescription/dispensed record. This, from a software development point of view, could be addressed by just catering to the need. However, an engineering perspective would be to create a toggle switch that can be switched on or off depending on the future requirement of the client. This kind of direction can be ensured to be implemented when the requirements governance body within the e-health vendor has a control over such change decisions. This initiative has already been started by the e-health vendor.

Apart from these, architectural guidance could possibly be from external authorities like NEHTA for interoperability projects in e-health. But in projects where multiple vendors are participating like PCEHR systems, there should be strong governance from NEHTA at an enterprise architectural level and programme level so that the interoperable requirements are engineered effectively. This would reduce the scope for requirements gap. This also reduces ambiguity at requirements implementation level.

Further, the responses in the interviews suggested that not all project managers and others who negotiate the requirements read the requirements documents but start negotiating the requirements with the clients on the basis of the knowledge from discussions and emails. The ideal way to handle this would be through having a thorough understanding of the project’s requirements and its prioritisation by the governance team.
4.4.3 Requirements Engineers in e-health projects

In agile models, the change in requirements is always welcome but should be estimated, incorporated and managed across the project till the requirement is delivered to the client. So, the requirements engineers’ works does not stop at just documenting the change but to have an insight into the different areas where it is affected and ensure that the impact is communicated to the appropriate project stakeholders. This is how requirements can be engineered and not just documented. This also calls for having the right people in the right place to ask the right question is required. Understanding the business problem of the client should be the priority. The requirements engineer should be listening but should question and challenge the stakeholders if the problem is not defined properly. If the customer is driving in the solution mode and starting to define how the solution should work then, they should appropriately be stopped by the requirements engineer to identify the problem and leave the solution to be defined by the solution providers. So the requirements engineer should exhibit leadership qualities and take charge in such situations.

However, in some cases, the senior requirements engineers could add their experience to the requirements and skew the requirements. A correct balance of listening to the customers and foreseeing requirements is a most called for professional approach of the requirements engineer towards engineering the requirements in e-health projects.

This Rigor with reference to defining the requirements framework is expected from the requirements engineers of the e-health vendor. A three dimensional approach to have the capabilities set for the requirements management and engineering is required. The three dimensions are the people, procedure and the techniques used by the requirements engineers for the e-health projects. The requirements engineer’s responsibility is not just vested in defining the problem, understanding requirements and documenting them to communicate to the client but should stretch across the SDLC with collaboration with the architects, designers, analysts, developers and the quality analysts in the software engineering team. This ensures the requirement’s functional awareness across the software engineering team.

The RE effort should be high in the beginning of the iteration of the project. But it reduces during the development and testing phases of the iteration to a flat low line. This is because there is always requirements clarifications need throughout the iteration and the project.
Further, there are always changes to requirements during or between iterations. So, the RE activity should be throughout the project and its iterations. It becomes a very small overhead for the rest of the project. But not allocating to the RE overhead can be disastrous to the project.

One of Young’s (2004) publications suggest that projects that had the RE cost for up to 2% to 3% over the total cost of the project had overrun the total cost of the project by 80% to 200% where as projects which invested in 8% to 16% of the total cost of the project on the RE process had overrun the total cost of the project by 0 to 60%. Another study has shown that 75% of the defects found during test for exit / acceptance criteria are because of poor requirements definition (Experimentus, 2010). So, having a requirements engineer on board throughout the project is most required.

In some of the e-health projects that have been used in the study, the e-health vendor has embarked on the project as last minute solution providers or problem solvers. This pushes the governance team to start off the project with compressed development and testing time. As RE is a non-tangible process, it gets compressed even further. Furthermore, many stakeholders in the health industry do not have enough technical knowledge to appreciate the need for requirements documents and as these documents are the only tangible outcomes of the RE process, they do not feel the need for it. The project managers argue that this is the main reason that the structured RE process is compromised on some projects. So, if the client agrees to this bureaucratically, then they usually agree to do the requirements.

RE process and planning of the project are two different processes that co-exist and depending on the size of the project, they are implemented with different intensities. For every customer need, there are usually high level requirements and detailed requirements. Based on these, planning is carried out multiple times in e-health projects. In the feasibility study phase planning is based on the scope of the project. This leads to the high level requirements. These requirements help in further planning the rest of the project in iterations. The high level requirements sometimes becomes very detailed requirements and some requirements engineers find it challenging to not do the high level
requirements in detail. On big e-health projects, the requirements engineers are expected specifically not to go into the details of the requirements when defining the high level requirements.

On such big projects, there are usually two business cases presented. The business case version one will be about the concept of the project. The requirements engineers should work on the scope and the architecture of the possible solution and co-ordinate with the project managers. However, in some instances where the architectural need is complex like the e-health projects, initially the requirements engineers (the business analysts and the architects) should have a collaborative work approach between themselves. The project manager’s intervention starts when the virtual architecture of the solution has stabilised. At this stage, the outcome of this feasibility study is the high level requirements, enterprise design and high level project plan. So, in such cases, the requirements engineers should take extra care to not do the detailed requirements at this stage.

As Sivakumar (2013) suggests, a requirements engineer has to focus on documenting the requirements as epics or user stories focusing on understanding the business problem. This implies that the requirements engineer needs to be a subject matter expert to give continuous support to the development and the testing team to clarify any issues out of these documentations. The requirements engineer is to focus on always be on the business problem and provide inputs to the design and development team to explore different options for the proposed solution. Further, the requirements engineer should have the current issue and the future scalability always in the arena and work with the clients. They should also work with the development team to ensure the problem centric focus is not shifted and is within the scope of the project.

In the concept phase of e-health project, the scope, the impacted areas are identified, and the likely solution is visualised analysing how the solution should look like. This is the requirements engineers’ responsibility to exhume the requirements. The first course is the functional requirements followed by non-functional requirements. Typically the UI design is at the end. This is trailed by the technical design. Moreover, in e-health projects, the requirements are never specific to only one solution. There is always a need to interact with another system or should feed into another system. So, there is always interoperable functionality that should be captured in the e-health projects. It is the responsibility of the requirements engineers to take that information and map it to the solution being provided. So, all these dimensions should also be considered by the requirements engineers when defining the requirements.

In some cases, the change is not to enhance functionality. For instance, with one of the projects that have been studied, there were different messages being sent into the information exchange broker. They were two messages which are a ‘Dispense’ message and an ‘Amend’ message. The ‘Dispense’ message is providing a functionality of dispensing an item in the prescription and amending a dispensed item details. So ‘Amend’ is a subset of the dispense message functionality. A further
investigation on this showed that the ownership of requirements was vested with different people and was prioritised accordingly. This is the reason behind the exchange handling both the messages. This unnecessarily increases the functional load on the message broker. So, the change here is to stop the exclusive ‘Amend’ message from hitting the information broker. Here the change is to downsize the functionality. This is because; the responsibility of owning the requirements was distributed across different people. This issue is a result of inappropriate stakeholders prioritising the requirements inappropriately.

4.4.4 RE process in e-health projects

Even if it doesn’t appear upfront, the requirements when done appropriately can reduce the number of hours spent clarifying ambiguity during development on each requirement on the back end by the development team.

Further, the governance perspective from the project management should be requirements focused. Incorporating the features mentioned by Larson & Larson (2010b) considering the following to be incorporated about the requirements in the project management is a proposed method to be followed.

- Establishing alliance and partnership with the stakeholders and clearly articulate the payback of each requirement to all stakeholders
- Ensuring that relevant functional stakeholders and commercial stakeholders are involved in the prioritisation of requirements. This ensures that the commercial aspect of the requirements is visible to the stakeholders.
- Involving the requirements governance body when negotiating prioritisation of requirements
- Ensuring that the governance team actually reads the requirements document and understands them before going into further negotiations with the client.
- Ensuring that the development of the requirements should be only after tangible amount of requirements is confirmed.
- Ensuring that the quality gates (entry and exit criteria) are well defined in the requirements framework communicated and is executed accordingly. This will further ensure the requirements are managed within the scope of the project
- Ensuring requirements traceability by using the existing team foundation server throughout the project management (this is further deliberated below)
- Educating the software engineering team about the deliverables and the requirements management process
- Incorporating change in the project management process should be done through continuous effective communication to the team.
- Ensuring that the customer agrees (and not compromises) to documenting the requirements
- Ensuring that requirements is adequately documented and is not heavy weight.
• Ensuring that all project deliverables, processes and entry and exit criteria are clearly mentioned in the documentation.

In some cases the requirements are done by the business analyst from the customer’s office. This could be someone employed at the client’s office or some consultants appointed by the customer specifically to do their requirements. In such cases, it should be the business requirements. With reference to Zachman’s framework (Open source enterprise architecture framework) (Zachman, 2008), the ‘what’ namely the problem should be defined. However, these business analysts usually are overwhelmed by the complexity and create slightly technical requirements documents. These documents tend to answer the ‘how’ question of the requirements. The customers expect the vendors to straight away work based on these requirements and do not allow for the vendors to decide on the solution architecture.

The ‘how’ aspect should not be addressed in the requirements that come from the customer. If it does, then it actually would be deliberating about the technical aspect without much knowledge of the possible solution. The technical document is usually the outcome of the analysis and design phase of SDLC. This is the starting point of all disasters in the project. With documents like these on the basis of which projects get implemented, the document does not carry the responsibility that is vested in the architects who analyse and design the solution. In situations like this, the requirements engineers should be able to push back to the client about the issue and should be able to be more authoritative to take charge of the requirements and the engineering process of the requirements.

The functional requirements documents line up to deliberate what gets implemented and the technical documents indicate ‘how’ it will get implemented. In projects like the ones mentioned above, it is ideal that the architects from all different stakeholders are to work together to decide about the ‘how’. So, in e-health projects, where majority of the times the scenarios are like these, even if the high-level requirements and the business requirements are done by the customers or the customer appointed third party business analysts, the functional, non-functional, the technical solution design document should be done by the vendors. These kind of situations in the e-health projects call for the vendors to have technical business analysts in the requirements engineers team who have the strong business domain knowledge, technical knowledge and excellent inter personnel skills to minimise the intervention of external business analysts on the project. The approach has to be to do the ‘how’ documents by the e-health vendor.

In cases where this is not possible, the e-health vendor is supposed to gather those requirements documents and do their own documents and play them back to the customers and get their sign off on those documents to ensure it has been translated right. This is synthesising waterfall methods in agile models. This could be feasible where the iterations of the projects are ranging from 3 to 6 months.
However, this approach may not be possible for agile projects where the iterations are very small like monthly or bi-monthly iterations. In such cases, doing the user stories and deriving the functional, development and testing task would be ideal.

Ideally, when vendors’ do not own the requirements documents they should own the technical requirements. In cases where the requirements are done by the third party business analysts, they are never done as changed requirements. The versions of the documents change with added functionality. Vendors see this as a change and thus initiate the change process. However, the requirements document never gets stable. So in these scenarios, instead of going through the formal change management process, a service level agreement between the clients and the vendors can help to incorporate such changes into agile development and thus can be done in iterations as a T & M model project where the time and material (resources utility) are charged to the client.

It would be ideal to have all the different types of requirements documents written. However, deciding on what kind of requirements documents are to be written is led by the scope of the project and its impacted areas. At times, the business analysts start with the user requirements and then ends up doing the functional requirements which flows into the technical requirement document. In complex e-health projects, the purpose of the documentation of the requirements initially is to give an understanding of what is being done. The user requirements define the use cases and this could help in further doing the user stories during development. In case of projects where the iterations are longer and project complexity demands the creation of different requirements document, the document should address all the business rules that are to be implemented.

Within agile projects like interoperability projects in e-health industry – as mentioned above, where the technical complexities and intricacies are very high, the requirements if spoken in business language can be simple business requirements. For instance, within the same information exchange broker which facilitates the transferring of e-prescription messages, if sending different prescription messages from different vendors is required, then, different prescription messages (like prescribe, dispense, amend, cancel and inactivate prescriptions) from one clinical information system should have different mapping needs within their XML schemas. This should then be mapping to the same elements in the CDA documents from another clinical information system. In such cases, the mapping document serves as the requirements document and the technical requirements. Currently, in the above mentioned projects, these documents are missing. These mappings are being held only in the CDA mapping tool which for a normal eye would just look like a spider web. For projects where these are missing, it is always a good idea to retro create these documents as moving forward from here, it will get even complicated when visited by another interoperability issue. So these kinds of technical requirements should be a part of the project deliverables. The requirements engineers should
be able to identify these requirements and appropriately educate the stakeholders about eliciting and documentation of such requirements.

Within this study, this scenario has been noticed twice were these CDA mapping documents are missing. These are on two projects that had been worked upon and implemented simultaneously. So, there was no option of learning from one project to implement it in the other. In this kind of situations there is an area of blurb where the customers think it is within the scope of the project and the vendors think it to be a change and this will end up being a reason of conflict.

RE is different within the different SDLC methodologies. It is the paradigm shift in agile projects. Requirements are not done as a large piece. AGILE is as structured as waterfall approach. However, in the name of agile approach; software engineering teams do not think it necessary to revisit the estimates for each of the iteration. To elaborate more on this, if the estimates for a specific requirement ‘A’ are given say two months ago where the solution has ‘X’ features, after two months, there will already be a few other ‘Y’ number of features added. So the number of requirements now are ‘X+Y’. So if the customer would like the software team to build ‘A’ on those estimates, the initial estimates would not have considered the differential ‘Y’ requirements’ impact on the solution for ‘A’. This will surely impact on the project cost, the solution design and possibly on the architecture of the solution. All these factors will impact the deliverable. If the client is not made aware of these, then this could lead to disagreements and coercion in the project. To avoid this, the agile requirements engineers should revisit the requirement (that has to be changed) and re-estimate internally with the software development team and then confirm the do-ability of the deliverable to the client. Thus requirements should be constantly accessed and monitored throughout the life cycle of the project.

Recording these learning’s as a part of the project helps not just the project planning but will also improve the RE effort on similar projects in the future. These individual learning’s should be captured and streamlined into the project activities as a quality measure.

Within e-health projects, there is always a varied set of dependencies that determine the priority of the requirement. It could be customer requirement, the functional requirement, non-functional requirements, and a change in the existing solution, a security requirement, a privacy requirement or totally a new service. So it is the responsibility of the requirements engineers to understand the requirement and its dependencies to get to the factors that can influence the requirement.

Further, within the interoperability project that was studied, one requirement is that the other vendor who does similar e-prescription messaging is to work on the interoperability requirements along with the e-health vendor. However, they ended up coming into the scenario only at the UAT stage. The end to end solution feasibility required both the vendors to equally scribe the requirements and work together to get to this solution. One of the metaphors that were used by one of the respondents was
that the solution was designed and built as a ‘shark’. But the customer realises that it was not ‘shark’ but ‘dolphin’ that they wanted and unfortunately, now the vendors are now left to convert the shark into a dolphin. Architecturally this kind of shift will be a big change but can lead to coercion in the project and the vendor can lose the goodwill and the customer is left with disappointment. All this could possibly be because the RE process is not optimised and is not customer need centric.

Very lean governance on both the vendors and no deemed governing authority between these two vendors created a lot of chaos with the solution. Further, the client who engaged these two vendors kept changing the requirements which are not documented but communicated on the emails. This calls for a total new dimension of RE of interoperable e-health projects and communication channels in such projects.

The requirements engineers should also ensure that the following are incorporated in the RE process

- Ensuring that the high level requirements are not detailed. It should only define the scope and the enterprise design of the solution.
- Ensuring to ask the right questions while eliciting the requirements
- Incorporating ‘user stories’ in iterative projects within the SDLC process for internal use by the development and testing team.
- Ensuring that the user’s perception and interpretation of the process that is being worked upon are clearly documented as this is the basis on which the development is done in iterative projects where the iteration time lines are low
- Consolidating and managing the requirements within the e-health vendor in a repository (TFS)
- Ensuring that apart from communicating the requirements to the clients, they should be communicated to the internal development team through workshops and continuous support.
- Ensuring that the architectural diagrams are regularly versioned and changed as part of the RE process. This can help the new members of the team to up skill faster
- Revisiting the requirement (that needs to be changed) and re-estimating internally with the software development team and then confirm the do-ability of the deliverable to the client.
- Identifying the key dependencies and influences of the requirements with reference to the security, performance and usability factors.
- Considering all different aspects and stakeholders and gauging the priorities set and assist the project manager and the governance team to prioritise the requirements appropriately.
- As new e-health systems always interact with the existing systems, ensuring that the integration requirements of such systems are always thoroughly elicited and documented so that they can further be communicated and agreed upon.
• Ensuring that the requirements engineers possess sound technical and domain knowledge to understand such capabilities of the requirements.

In e-health projects where the solutions are designed to address the middle tier requirements and not just the front end requirements, the focus need not be on using a requirements management tool but should be more concentrated on the requirements tracking practices and traceability. The requirements tools can be put to good use only where the project’s requirements are massive in volume. For small and medium size projects where the volumes of requirements that are to be delivered are not massive, it is the matter of using appropriate process model that can help in managing the requirements better.

In e-health projects, flow charts, data flow diagrams, entity relationships diagrams, state transition diagrams/tables, use case diagrams, sequence diagrams, etc., deliberate on the complex functionality of the different components and the interconnectivity between them. The architectural design of the project is at times best understood in the diagrams. Having such diagrams will facilitate the new staff on the project to come up to speed with less time spent on knowledge transfer. So doing them as a part of requirements documentation is essential. Further, because there are no proper requirements documentation on a few existing projects, currently for each change, the software engineers are to reverse engineer from the code to understand the issue and the change. In complex projects, this could be a major overhead which can be avoided by good requirements notation and documentation.

The e-health vendor in this study has had experience in e-health projects for about two decades now and since then would have had some operational metrics and lessons learnt it is repositories. Currently these are not being used for analysing the impact of the change on any project. The impact can be on different phases of the SDLC, on the solution and on the usability for the user. Currently there is no process in place to analyse this. The e-health vendor should evolve a process where the impact analysis can be assessed in real terms against some projects metrics.

All above reasons call for a paradigm shift in the e-health vendor’s approach towards requirements. In agile projects where change has to be incorporated in the product backlogs, doing the required requirements activities before development starts for the iteration is inevitable. This can be made easy with the collaborative work of the requirements engineers with the clients, stakeholders, development and the testing team. Further incorporating a test driven development approach will ensure that the requirements are peer reviewed, verified and validated at each phase of the SDLC. This ensures that the requirements gaps are identified if any, at a very early stage and can be worked upon. The change control and management can be lot easier through proper impact analysis.
Once these are all worked upon accordingly, the internal requirements management can be meticulously executed using the existing licensed Microsoft Visual Studio TFS 2012. Below is my forethought of the optimised RE process for the e-health vendor that has been studied:

1. Identify the business need and define the business problem
2. Gather requirements information from all relevant sources
3. Elicit the information and classify requirements
4. Document the requirements (ideal to have user stories with relevant use case diagrams, state transitions and decision tress and decision table explaining all the required business rules for the solution that has been designed)
5. Send the document to the client and the internal requirements governance team
6. Confirm if the document deliberates on what the client needs and if it matches to the road map defined by the governance team. If it does not meet these requirements, then get the feedback of the change from them and go through step 3 to step 5.
7. If confirmed, use the agile template model for the project and add these requirements as ‘parent’ user story to the product planning.
8. Confirm if impact analysis is required on these requirements. If required, do the impact analysis and add the details to the requirement
9. Prioritise the requirements with the appropriate client stakeholders, governance team, project managers and requirements engineers
10. Confirm if the requirement is prioritised and approved after the impact analysis process.
11. If not approved, revisit the client with the product backlog at an agreed later date and go through the process from step 5
12. Set the user story for the iteration.
13. Confirm if the user is set for the current iteration
14. If yes, create a test plan for the iteration in Microsoft Test Manager
15. From Step13, create linked child use stories for all granular requirements against the parent user story
16. Create linked tasks to all child user stories and assign to the developer. Create test cases for each child user story for the iteration
17. Developers develop this task and attach the code to the source code to TFS against the work item which is deployed into internal test environment.
18. Drag all these test cases into the test plan and test
19. Confirm if any bugs are found during testing
20. If bugs are found during testing then create a linked bug from within the Microsoft Test Manager and assign to the developer. The Developers develop this task and attach the code to the source code to TFS against the work item and follow the process from step 17
21. If no bugs found, the tester to pass the tests and set the status to ‘Ready to UAT’ and assign to the deployment manager
22. The deployment manager to package all user stories and build into UAT
23. If a bug found in UAT, then follow from step 20 again
24. Else deploy the requirement into production.

Below is the flow diagram of the process above.
Figure 42: Proposed RE Process in small and medium scale e-health vendor companies
Integrating TFS in this process can ease the hassle of tracing the requirements. The user stories, tasks, test cases, bugs are all the inbuilt work items in TFS for the agile project template. The user stories can be created within the project. All the user stories for the project can be seen in the ‘Product Planning’

![Figure 43: Product Planning in Visual Studio TFS 2012](image1)

and when assigned to the iteration, can be seen in the ‘Product Backlog’

![Figure 44: Product backlog in Visual Studio TFS 2012](image2)
The standard user story will have the feasibility to assign it to a team member, set the statuses from ‘active’ to ‘ready for UAT’ the statuses follow the activity and the phase in the SDLC of the iteration like active, work in progress (Dev.), Dev complete, ready for testing, ready for UAT, close. The area of the component within the solution can also be assigned to.

![User story in Visual Studio TFS 2012](image)

Figure 45: User story in Visual Studio TFS 2012

Once the user story is saved, within the implementation tab, new child or parent user stories can be created.
and new tasks can also be created.

The new test cases can be created from the ‘test cases’ tab.
All these when created in this order are linked to the user story and can be found in the ‘All Links’ tab.

Microsoft Test Manager is an integral component of Visual Studio 2012 in which the test plan can be created and all user stories can be seen and tested. Once the user story is added to the test plan, all test cases linked to the user story appear in the test plan.
Any bugs found in testing can be created from here and gets linked to the test cases.

This can further be moved between iterations if required.

So following this process not just ensures requirements management but also requirements traceability (forward and backward) from the user story to its bugs and from the bugs to the user story can be achieved. A pictorial representation of the same is seen below.
This management process, if incorporated will help the project managers and business analysts to get to all requirements under product planning and can get the product backlog to work with on each of the iteration. Further, creation of child user stories at granular level of the requirement can ensure management of requirements without missing the requirements during the development phase. Breaking down the user story to the granular level will further help the system analysts and the solutions designers to visualise the solution better and create the tasks accordingly. The developers can unambiguously develop the solution and so can testers test. This ensures that no requirements are missed out during the development and testing phase.

The above proposed solution of the RE process will lead to effective external and efficient internal management of requirements in the e-health vendor company where this study has been carried out. This model has been deliberated with a few software engineers and the feedback has suggested that this should be further presented to the management and leadership team to implement this model in the next project iteration.

**4.5 Summary**

This chapter encapsulates analysis and interpretation of the data collected in survey and interviews. The 3 hypothesis drawn earlier in chapter two have been proved to be appropriate in this chapter. Further, the following have been observed.

- **Observation 1:** requirements documents serve as an input for planning in e-health projects
• **Observation 2**: Software developers do not refer to e-health requirements documents but rely on the deliberations with the tech leads, architects, analysts and other peers for understanding the requirements

• **Observation 3**: It is required to have high level requirements confirmed before starting development in e-health projects

• **Observation 4**: Structure RE practices helps in enhancing planning, delivery management and the quality of the e-health projects.

• **Observation 5**: Appropriate prioritisation of requirements after considering many dimensions and aspects of healthcare has a direct impact on the quality of the delivery in e-health projects.

• **Observation 6**: The customer did not spend enough time on the requirements related activities in the e-health projects that the respondents have worked on.

• **Observation 7**: User Stories are not developed in the RE process in the organisation (e-health software vendor) where the study was done for agile (iterative) process.

• **Observation 8**: Requirements are not being managed efficiently within the development team in a manner that is compatible to the development tool that is being used.

• **Observation 9**: Not engaging Business Analysts/Requirements Engineers on the e-health projects leads to chaos as there is no ownership and responsibility exhibited on the complete RE process.

Further, a synthesized approach has been suggested that could help optimise the requirements engineering process in e-health vendor companies. The suggested methodology incorporates a requirements road map, requirements governance team and meticulously management the requirements in the Microsoft Visual Studio Team Foundation Server.
Chapter 5

5. Conclusion

5.1 The research
In all the projects that have been used for this study, at some stage of the project, there have been some issues with the requirements and the way requirements change has been handled. This research project is an outcome of a curiosity to understand where things could possibly go wrong and where they can best be improved.

The study has helped to

1. identify and define the underlying problems within the RE processes
2. draw hypothesis from extensive literature review
3. do a survey and interviews with 50 software engineers and to analyse the results of the same
4. analyse and define some observations as mentioned in the previous chapter
5. present a solution that has been deliberated in the previous chapter

5.3 Future Research Direction
Not putting enough RE effort but just doing the iterative deliveries is not agile approach. If e-health vendors work on agile e-health projects with this style, then it is practically covering up the inefficient way of doing things under the name of agility. Inefficient way of doing the requirements has been the way of working on disastrous projects (Hull, Ken & Jeremy, 2011). This inefficiency is mostly being covered up in the name of agile way of doing projects. In many software projects, all the software
engineers are billable by the hour. So, the software companies are always looking at billing the client for the project. Except for the software developers, the rest of the team’s work is not tangible to the client. So, the approach taken by many software companies is to start the development work with as much less requirements so that the client billing can start. So, this leads to not having the requirements completely analysed for the iteration and starting the development and thus the deviation of the actual cost to the estimates. This is the practical reality of the businesses for many software vendors in general. Unfortunately this is being seen in e-health software vendors too. This is an issue that opens gates for further investigation.

Interview responses have suggested that RE processes can be coped well in e-health projects but for some drivers like the project management issues and political issues. It is intriguing to investigate more on this. This is another direction that can be pursued in future research.

Investigation into the interoperability of e-health projects has revealed that the RE process dynamics, the communication channels and the working approach have many unravelled dimensions to them. This surely paves way to further research to understand these different dimensions of RE process for e-health interoperability projects.

5.4 Limitations of this research

There are a few limitations in this research. However, these have been realised only while analysing the data.

- The 4th and the 15th survey questions in the survey were close ended questions and the options for the answers were limited. These should have had the ‘Other’ option where the respondent would have had an option to give more answers. This issue was realised only after the survey was answered by a few and the amending the question at that stage would have skewed the results. However, this was mitigated through capturing this information in the interview responses.

- The research was based on agile projects and incorporation of change in agile e-health projects. However, the survey questionnaire was not designed exclusively for agile projects. This is surely a limitation of this study.

- The impact analysis has been researched but the synthesized methodology does not cater exclusively to it. The solution incorporates impact analysis as an area but specific solution has not been designed. This is because the data analysis drove into the direction of the presented synthesized methodology.

- Further, the study and the solution are more driven by how the business requirements and functional requirements should be monitored and managed in e-health projects. However, the non-functional requirements and its management have not been addressed in this study.
References


Schwaber, K (20011). *The enterprise and scrum*. Canada : Microsoft Press


Appendix 1

Participant consent form

**Participant Consent Form**

**Research Project Title:** Requirements Engineering Process Improvement in Health IT Projects

I have had the research project explained to me and I have read and understand the information sheet given to me.

I understand that I don't have to be part of this if I don't want to and I may withdraw at any time prior to the completion of the research project.

I understand that everything I say is confidential and none of the information I give will identify me and that the only persons who will know what I have said will be the researchers and their supervisor. I also understand that all the information that I give will be stored securely on a computer at Unitec for a period of 5 years.

I understand that my discussion with the researcher will be taped and transcribed.

I understand that I can see the finished research document.

I have had time to consider everything and I give my consent to be a part of this project.

Participant Signature: .................................. Date: ..................................

Project Researcher: ................................. Date: .................................

**UREC REGISTRATION NUMBER:** ( )

This study has been approved by the UNITEC Research Ethics Committee from (November 12, 2012) to (November 11, 2013). If you have any complaints or reservations about the ethical conduct of this research, you may contact the Committee through the UREC Secretary (ph: 09 815-4321 ext 6162). Any issues you raise will be treated in confidence and investigated fully, and you will be informed of the outcome.
Information sheet

Information for participants

Research Project Title: Requirements Engineering effort to incorporate dynamically changing requirements in Healthcare IT Projects

Synopsis of project: A case study will be conducted to understand and improve the requirements engineering process on an e-health project that has run in two phases. This could possibly help in proposing an optimum requirements engineering model for e-health projects.

What we are doing

I will be conducting interviews with the projects owners and do a survey with the software engineers involved in the project to understand the usability of the requirements document during the life cycle of the project.

What it will mean for you

The software engineers may have to give half an hour of their personal time to answer few survey questions.

The project owners will also have to give an hour of their personal time to answer some questions on the project. This will help me do the case study.

If you agree to participate, you will be asked to sign a consent form. This does not stop you from changing your mind if you wish to withdraw from the project. However, because of our schedule, any withdrawals must be done within 2 weeks after I have interviewed you.

Your name and information that may identify you will be kept completely confidential. All information collected from you will be stored on a password protected file and only I and my supervisors will have access to this information.

Please contact us if you need more information about the project. At any time if you have any concerns about the research project you can contact our supervisor:

My supervisor is GUAN YUE HONG, phone 815 4321 ext. 6021 or email ghong@unitec.ac.nz
Privacy statement

Research Title: Enhancing Requirements Engineering effort to incorporate dynamically changing requirements in Healthcare IT Projects

Researcher/s Name: CHITTI ANUPAMA

Address: 1B, Kuaka Place, New Lynn, Auckland

Phone number: 09-8265944, 029 770 1541, 09-5260427

Email: chitta04@wairaka.com

I CHITTI ANUPAMA Agree to treat in absolute confidence all information that I become aware of in the course of transcribing the interviews or other material connected with the above research topic. I agree to respect the privacy of the individuals mentioned in the interviews that I am transcribing. I will not pass on in any form information regarding those interviews to any person or institution. On completion of transcription I will not retain or copy any information involving the above project.

I am aware that I can be held legally liable for any breach of this confidentiality agreement, and for any harm incurred by individuals if we disclose identifiable information contained in the audiotapes and/or files to which we will have access.

Signature: ................................................................. Date: ......10/09/2012.............

UREC REGISTRATION NUMBER:

This study has been approved by the UNITEC Research Ethics Committee from (November 12, 2012) to (November 11, 2013). If you have any complaints or reservations about the ethical conduct of this research, you may contact the Committee through the UREC Secretary (ph: 09 815-4321 ext 6162). Any issues you raise will be treated in confidence and investigated fully, and you will be informed of the outcome.