Studying Documentation Requirements for Quality Assurance in Healthcare Software Development Environments following Scrum Practices

A Thesis Submitted for Examination for the Degree of MPhil in Engineering at Massey University New Zealand

Shanuka G. Wickramasinghe

2016

ABSTRACT

Over the past decade software development has expanded into almost every sector of business and technology. Currently, Agile software development methods are much preferred over traditional software development methods which rely on heavy documentation. Agile methodologies such as Scrum (the focus of the study) rely on minimal documentation. However, software development organizations who seek accreditation against an internationally recognized quality management system (QMS) standard such as ISO 9001:2008 need to maintain a certain level of documentation to meet the requirements stipulated in the QMS standard. This study was undertaken to answer the following overall research question, in relation to healthcare software development: *what would be the minimum level of documentation that would be acceptable for a Health-IT organization pursuing Scrum, if they are to maintain an internationally recognized QMS standard such as ISO 9001:2008?* This overarching research question was first investigated through in-depth literature synthesis and subsequently discussed with a panel of experts. An iterative research design utilizing Delphi-like problem solving method was used to gather insights from Scrum practitioners. The study identified 23 documents to have varying levels of usefulness and importance to three categories of Scrum users, specifically Scrum Master, Product Owner, and Development Team. The study further identified the level of conciseness required in each document (to suit each category of Scrum users) and the stage in which each document should be prepared to add maximum value in using documentation. The study identified seven negative experiences Scrum practitioners come across: documents being difficult to understand by nontechnical customers; purpose of documents not being explicit; no follow-up with client's feedback; excessive re-work on documents; deficiencies in document validation; lack of risk analysis reports and disruptions in software development. The study also identified seven problems practitioners face in creating important documents: lack of skilled document writers; last minute/hasty document preparation; misunderstanding the purpose/intent of Agile; lack of a common documentation standard; perceiving document creation as a burden; poor tooling for documentation and lack of right staff. It is expected that the study would benefit both the academia and the practitioner in gaining greater insights on the issue of documentation in Scrum.

ACKNOWLEDGEMENT

Firstly, I would like to thank my husband for all his continuous support, dedication and encouragement given throughout the period of my study. Without him backing me up at the times when I was under immense stress and pressure, I would not have been able to complete my thesis. Although my extended family (parents and close relatives) are not in New Zealand, without their blessings and love, completing this study could have been extremely difficult. I hope I have made them proud.

A thank you is also due to my supervisors, Dr. Nihal Jayamaha and Dr. Anuradha Mathrani. Thank you for all the help and guidance you gave me to organize my work in a logical fashion. I really learnt a lot by working with you.

Also I wish to acknowledge all participants in data collection rounds, for their support in making this research study possible.

Finally, I would like to thank to my previous employer Orion Health for allocating me study leave and helped me with the thesis.

Without all of your help and guidance, this thesis would not have come to fruition.

Thank you again, everyone.

Shanuka Gimhani Wickramasinghe
BSc (Hons)
Business Analyst
Bank of New Zealand

TABLE OF CONTENTS

Contents

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
Introduction

1.1 Software Development and its Current State

Over the past few decade software development has gained an increased importance in the sphere of information systems because software development affects innovation and change in business and technology (Abrahamsson, 2002; Cohen, Lindvall, & Costa, 2003; Ghezzi, Jazayeri, & Mandrioli, 2002). Given the rapid rate at which human needs and wants increase, the expectation of having faster computing and effective systems to build and communicate with software is also increasing at a similar pace (Cohen et al., 2003). Consequently, sophisticated software systems that cater to complex information processing requirements are being used around the world, helping millions of people to accomplish their tasks efficiently and accurately on a day-to-day basis (Eden & Aviram, 1993; Jansen, Spink, & Saracevic, 2000). The software development environment being covered in this study is within a healthcare context, which has its own level of information processing requirements.

Agility is a key requirement today, at least in the software development industry, to satisfy the customers; being Agile enables the provider to respond to their needs quickly (Cohen et al., 2003). Consequently 'Agile methods', which were introduced in the early 2000s have now become the norm, rather than the exception, in the software development industry (Schwaber & Beedle, 2002). Being Agile also means being flexible, meaning being able to adopt to the changing requirements more efficiently, evolving solutions on an incremental basis to maximize customer value. This is the main reason for Agile methods being chosen in favor of the traditional methods by software professionals around the world (Ming, Verner, Liming, & Babar, 2004).

1.2 Agile Software Development

In traditional software development, work begins with the elicitation of a well-documented set of "complete requirements", which is followed up by a matching architectural and high level design diagram (Atahualpa, 2014). In the traditional approach, when the requirements and the designs are completed at the project initiation, these are considered to be concrete, meaning that changes cannot be made even when a requirement change has been identified (Awad, 2005). With the rapid changes of the industry and the technology, having concrete requirements made software professionals frustrated, and to compound matters, often the customers also became progressively unable to conclusively state their needs—yet, the customers expected more value from the software (Awad, 2005; Stober & Hansmann, 2010; Stoica, Mircea, & Ghilic-Micu, 2013). In terms of quality assurance, traditional methods were considered to be documentation driven, "heavy weight" software development methods (Beck, Beedle, Van Bennekum, et al., 2001). Agile methodologies, which are inherently flexible and "light weight" in nature, were introduced by a well-documented list of consultants to overcome problems in the traditional, documentation driven, heavy weight approach. In software engineering, the term 'heavy weight' is used to mean the traditional, process-driven and highly regimented approach to software development. The antithesis of this is the 'light weight' approach. Lightweight approaches are inherently people driven (Beck et al., 2001).

Agile is a generic term being used to represent specific types of approaches such as 'Extreme Programming', 'Scrum', 'Adaptive Software Development', 'Crystal', 'Feature-Driven Development', and 'Pragmatic Programming' (Cohen et al., 2003). Out of these representations, Scrum seems to be the most popular Agile methodology because of its adaptability to a variety of situations (Schwaber & Beedle, 2002). Scrum is also the software development context covered in this study. A salient feature of any Agile methodology is low reliance on detailed documentation, which is also one reason Agile methodologies are being treated as light weight.

1.3 Quality Assurance and Quality Management System Standards

Any organization that produces a good or service has a quality management system (QMS) (Summers, 2010). A QMS is the organizational subsystem containing the resources (e.g. people), the structure, policies and objectives to design, develop, produce and maintain what the customer wants (Schlickman, 2003; Summers, 2010). It is one thing to have a QMS in an organization and a completely different thing to assure the customer that the QMS under consideration meets certain minimum standards to provide the requisite quality. This is where quality assurance and QMS standards come into their own. When developing software systems, quality is a primary consideration that every development team has to take into consideration, regardless of the specific process or methodology being used (James, 1989). Most organizations that develop software are quite explicit about the quality assurance standard/s to which they comply with. ISO 9001:2008 is arguably the most recognized international standard on a QMS (Hooper, Cianfrani, Tsiakals, & West, 2001).[1] This standard specifies certain *auditable* requirements the QMS has to comply with. ISO 9001:2008 itself stipulates certain mandatory documentation requirements while the auditors who audit the QMS of the organizations will also often require additional documentation being available as evidence of compliance with each clause and sub-clause of the standard (Bach, 2003). To be more specific, to comply with an internationally recognized QMS standard such as ISO 9001:2008, an organization needs to generate certain high-level documentation (e.g. quality policy and quality objectives and the quality manual) as well as operating level documents (e.g. quality procedures, quality records) on quality assurance, which can often drive the organization towards a document-driven heavy weight approach, which does conflict with the Agile approach.

QMS standards such as ISO 9001:2008 have become commonplace because organizations need such standards, if not to improve their QMS, at least to demonstrate to the customers that the QMS of the organization can be relied upon for delivery of high quality products and services that the customers want. Also, often, the customers themselves insist on the producer (in this study, the healthcare software development organization) to demonstrate compliance with an international QMS standard such as ISO 9001:2008. Being ISO 9001:2008 (or similar) compliant usually means being document heavy (heavy weight) while being Agile means document light (light weight). Contemporary software development organizations are confronting with the challenging proposition of being Agile at the same time being ISO 9001:2008 (or similar) compliant. The question then is how could an Agile/Scrum software developer prepare their documentation to be 9001:2008 compliant? It is this pragmatic question that drives this research, within a healthcare software development context in New Zealand.

[1] At the time of finalizing the methodology this thesis, ISO 9001:2015, the QMS standard that replaces the current standard (ISO 9001:2008) was in draft/discussion stage. This research does not look at ISO 9001:2015, which contains more elements on risk management.

1.4Healthcare Software Development and the New Zealand Context

Since healthcare software (Health-IT) is developed to work closely with clinicians, practitioners and patients, it is unsurprising that Health-IT has been a regulated field, compared to other domains (Anderson & Agarwal, 2011; Jepsen & Jepsen, 2008). Health-IT contains personal health information which are treated as highly sensitive in medical practice (Anderson & Agarwal, 2011; Frost & Massagli, 2008). Therefore, maintaining a patient's privacy and security is considered as a high-priority task in healthcare software development (Dehling & Sunyaev, 2014). Unlike the other software, healthcare software can affect peoples' lives and hence, a healthcare software development process should embrace quality from the beginning to the end (Jepsen & Jepsen, 2008). Since failure of Health-IT software can cause potential harm to the society, the development teams are entrusted with the responsibility of finding the hot spots of security, privacy, and other confidential aspects before the software is being released (Armour & Thizy, 2010).

Often, healthcare software is based on very complex and lengthy standards and specifications (Armour & Thizy, 2010; Jepsen & Jepsen, 2008). It is very unlikely that all the members in the development team read and understand these complex standards and specifications. Therefore, maintaining a subset of documents containing the important (or critical) quality conformance requirements, from a universe of all desirable documents, would likely to have a salutary effect, by way of providing proper guidance to the development team (Armour & Thizy, 2010; Cellucci, Trimmer, Farnsworth, & Waldron, 2011). Moreover, maintaining records and keeping track of the work done—in terms of development, testing and other validation—will keep the team on the right track by being able to develop the right product (Anderson & Agarwal, 2011; Cellucci et al., 2011). Consequently, there will be less chance of missing important elements of the software that is needed to be developed to sell a product that is valued by the customers. Yet another important fact is that, regardless of how well the development team executes software development, if the customer doesn't understand how to use the product and/or how to troubleshoot problems, the product is likely to fail (Yunan, 2013). Therefore, it is important to have documents (e.g. the user manual) which are simple and easy to understand by the users such as doctors, clinical staff, patients and the administrative staff, in a Health-IT context (Yunan, 2013).

1.5 Aims and Objectives of the Study

Contemporary businesses face unique challenges posed by the business environment: competitors, customers, regulatory bodies and so on. Organizations formulate and implement strategies to overcome these challenges. Although some challenges are opportunities to capitalize, often the challenges pose serious threats to the business. In software development, taking recourse to Scrum (or an alternative Agile methodology) is a well-known effective business strategy to mitigate the threats. Scrum and other Agile methodologies do not favor documentation, which is however, a necessary aspect in quality assurance. In particular, an internationally recognized QMS standard such as ISO 9001:2008 will require a certain level of documentation for accreditation against the standard. ISO 9001 accreditation or not, as the researcher has highlighted in the previous section, certain level of documentation is necessary for a software product to function efficiently and effectively.

The aim and objectives of this study are governed by the following *overarching research question*:

"What would be the minimum level of documentation that would be acceptable for a Health-IT organization pursuing Scrum, if they are to maintain an internationally recognized QMS standard such as ISO 9001:2008"?

1.5.1 Research Aim

The aim of this study is to identify the type of documents an Agile (Scrum) healthcare software development organization needs to have to seemingly meet the requirements of an internationally recognized QMS standard such as ISO 9001:2008.

It is important to note that whether or not the QMS of an organization meets the requirements of ISO 9001:2008 is a matter to be judged by the auditor (typically known as the registrar) of the certifying body. Evidencing documented information is only one requirement for ISO 9001:2008 accreditation. A researcher can only make a judgment (based on data collected from the relevant experts) as to whether the QMS of an organization contains adequate level of documentation, for ISO 9001:2008 accreditation.

1.5.2 Specific Research Objectives

The specific objectives of the study are *to* determine the following, in the case of a healthcare software development organization that uses Scrum:

I. The critically important documents for quality assurance; more specifically, documents that are required to (seemly) meet the documentation requirements of ISO 9001:2008.
II. The content that should be included in each document and the stage of the project at which each document should be prepared.
III. The issues surrounding documentation.

The three research questions identified from the literature are posed at the end of chapter 2 (Literature Review). Each research question corresponds with the above three research objectives on a one-to-one basis.

1.6 The Benefits of the Study

This research is beneficial to both the practitioners and the academia.

For software professionals who work in Scrum or trying to adopt Scrum can use the findings of this study to build an appropriate guideline for its staff, in regards to documentation. Currently software professionals in Scrum environments are largely unaware of the requirements of the ISO 9001:2008 QMS standard for certification/accreditation. Needless to say, this study is not about educating staff on the ISO 9001:2008 standard. Rather, the study looks at quality assurance from a documentation perspective. Therefore, this study educates the Scrum practitioner on documentation aspects of quality assurance.

Academia engaged in research on quality systems, QMS standards, and Scrum can benefit from this study because there is very limited published work on quality assurance in healthcare software development. Furthermore, the findings from this research can be either extended to the latest version of the ISO 9001 standard (ISO 9001:2015) or the research design can be replicated in a different setting (e.g. in an organization that is ready to switch to ISO 9001:2015) to generate new information (see footnote #1).

1.7 The Limitations of the Study

No study is perfect. All studies have limitations. This research contains the following limitations:

(a) The findings are based on a single case study. Although multiple participants were utilized to collect data, the findings (be it quantitative or qualitative) of a case study should be generalized with caution. The findings may not be generalizable across the universe of Health-IT organizations pursuing Scrum.

(b) The study is limited to the perspectives of the few respondents who participated in this study. Thus the findings may be biased towards Agile projects undertaken by these respondents. In addition, in regards to matters related to documentation (e.g. identification of useful and important documents), although panels of respondents were formed at various stages of the project to obtain convergent information, the researcher found at the outset that reaching a consensus on the terms of reference entrusted upon the panels was impossible. This was mainly due to the time pressure and the culture of the organization (the researcher was a relatively junior staff member of the case study organization). These factors forced the researcher to engineer an approximate solution based on respondent voting (details in section 4.2 in Chapter Four).

(c) The study does not take into account the ISO 9001:2015 QMS standard, which replaces the existing QMS standard (ISO 9001:2008) in 2016. The new standard puts more emphasis on risk management and less emphasis on so-called mandatory documentation (Hampton, 2014). The risk emphasis in the new standard may alter the types of documentation that are required for ISO 9001 accreditation (the researcher recommends that her study be replicated once the new standard comes into force).

1.8 The Structure of the Thesis

Rest of the thesis is organized as follows.

Chapter 2 begins with a brief introduction on the history of software development, including challenges faced by traditional software development environments, which actually led to a movement towards Agile methods. Chapter 2 then looks at the literature on the practices and significance on Agile methods, giving more emphasis on Scrum practices. The chapter then covers the literature on healthcare software development, including challenges and restrictions imposed upon by the ISO 9001: 2008 QMS standard, in relation to documentation requirements. Finally, the research questions are posed based on the knowledge gaps identified in the literature review.

Chapter 3 describes the methodology used in the research, in answering the research questions to achieve the research objectives. Therefore, this chapter begins with an overview of different social research paradigms available to answer research questions. This is followed by review of five prominent research designs available to conduct social research in order to choose the most appropriate research design. The researcher also explicitly states to which research paradigm her research belongs. Thereafter, chapter 3 focuses on data collection rounds and basic rules around it, covering the detailed description of sample selection and data collection. Chapter 4 provides results (organized in the chronological order based on the data collection rounds) along with an accompanying discussion covering substantive answers to the research questions.

Finally, Chapter 5 provides the summary and conclusions of the study, indicating what was achieved (and how), in regards to each research objective. The chapter concludes with directions for further research.

CHAPTER 2
Literature Review

2.1 Introduction

With the development of technology, computers and computer based applications have become indispensable in all industries. Software is a series of instructions which tells the hardware what actions to be performed. Software applications can specifically be developed to perform unique tasks or operations with less human intervention which helps business to perform their day-to day tasks effectively. Hence software has been a very important tool in operational activates and decision making for over five decades now, although it penetrated into day to day activities of a lay person, not so long ago.

This chapter firstly discusses about traditional software development and the issues it faced and gradual adoption of Scrum as the main development method. Then the chapter focuses on heath software development and current issues facing on the development industry. As a solution guideline then the chapter addresses the world renowned ISO 9001 standards and its applicability on a Scrum project. Further the chapter discusses existing literature and studies on documentation on Scrum projects and issues. Finally, the chapter presents the identified research questions.

2.2 Agile Software Development and Scrum Adoption

Software development originated as an un-planed and messy activity containing numerous codes (Rico, 2005). In the past, software was written with minimal planning and design considerations, with a short-term focus (Wirth, 2008). For the most part, this was appropriate at the time because back then, software solutions were produced for small scale systems (Taylor, Medvidovic, & Dashofy, 2009). However, with the passage of time and the development of the technology, systems started to become larger, requiring complex and heavy[2] software solutions. This leads to complexities around designing and producing solutions as well as fixing bugs and adding new functions (Taylor et al., 2009). In-order to develop the software through proper planning, methodologies were introduced (David Avison & Fitzgerald, 2003). Avision

[2] Heavy software solutions are the one which requires excessive amount of effort, in terms of planning, executing and verifying. These actions consume vast amounts of organizational resources.

and Fitzgerald (2003) define methodology as “recommended collection of philosophies, phases, procedures, tools, documentation, management, and training for developers of systems”. Software development methodologies help users in achieving efficient and high quality results. Depending on the era and demand, these methodologies have evolved from its basics.

Traditional methodologies, which were the norm few decades ago, are called ‘Plan-driven’ methods, because they relied heavily on complete set of requirements specified at the beginning (Stober & Hansmann, 2010). Therefore, software development could not be started up until requirements were completed and architectural and high level designs were ready (Stober & Hansmann, 2010). In competitive and dynamic environments, which are the norm today, plan driven approach was not always found to be practical because software requirements were not always rigid; they kept evolving as the user gathered new information from their operating environment. Thus slowly but steadily, the so-called rigid, plan driven, heavy-weight processes became un-popular both among the user and the producer (Stober & Hansmann, 2010). Consequently, most software development professionals started looking for more flexible methodologies that could accommodate changing requirements (Basil & Turner, 1975). Iterative methodologies were introduced as a solution; these iterative methodologies can address the changing needs of the customers (Basil & Turner, 1975). Based on the iterative methodologies, more effective and rapid development methodology was introduced in early 2000; this methodology was named “Agile”.

2.2.1 Agile Software development

Agile methodology was introduced by group of expert software development professionals, at a meeting on 11-13 February 2001, at The Lodge at Snowbird ski resort in the Wasatch Mountains of Utah. The meeting was held to discuss the future trends in the software development industry. The experts attempted to characterize the salient features of contemporary software development to identify an underlying theme or aspect in 21st century software development that could be reliably distinguished from traditional software development. They found that software development in the 21st century is characterized by having to be flexible, having to evolve with the customer (as the customer learns more about their requirements their software needs also change) and having to be less rigid (light weight). The experts named these commonalities of contemporary software development under the umbrella term "Agile Methodologies" (Beck, Beedle, Van Bennekum, et al., 2001). Consequently, the "Agile Manifesto" emerged as the guideline for developing Agile methodologies. This manifesto is being used by thousands of software development organizations around the world as the *doctrine in software development* (Schwaber & Beedle, 2002). Unlike the traditional software development methodologies, Agile methodologies claim to be more responsive to change; Agile methodologies emphasize effective communication between the stakeholders and the 'development team'; most importantly agile approaches develop software incrementally, in an efficient and effective manner (Beck, Beedle, Van Bennekum, et al., 2001). Figure 2.1 depicts how Agile methodologies could be distinguished from traditional software development.

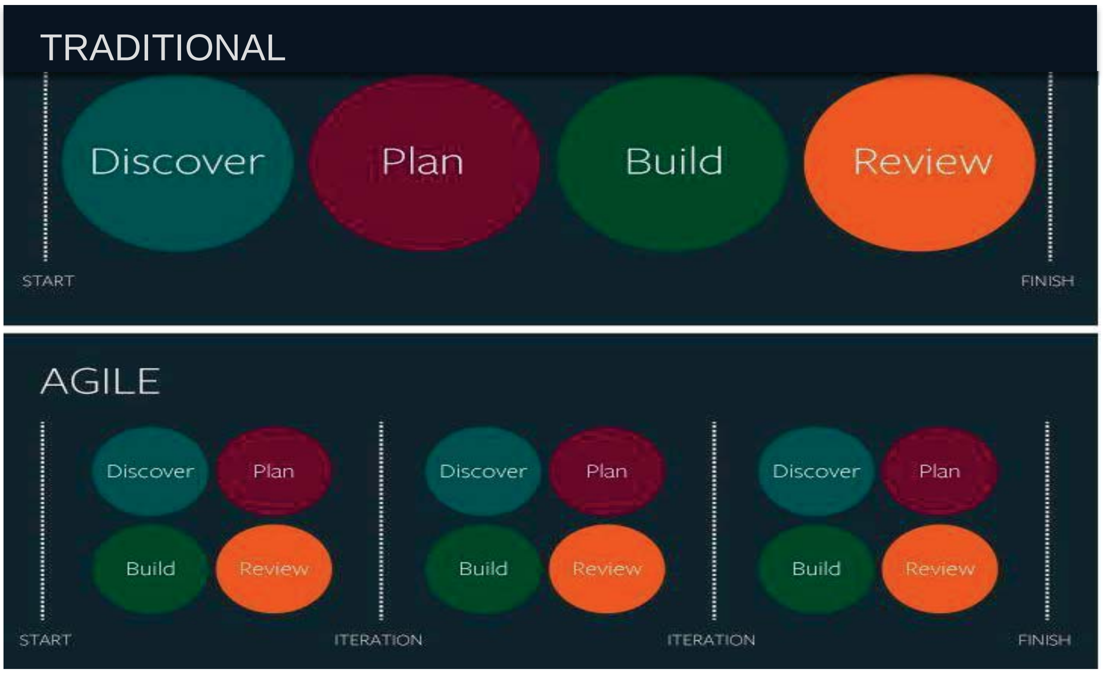


Figure 2.1: Agile vs traditional software development (Source: Archer, 2005)

2.1.1.1 Characteristics of Agile methodology

According to the Agile manifesto, Agile methodology values – "individuals and interactions over processes and tools"; "working software over comprehensive documentation"; "customer collaboration over contract negotiation" and "responding to change over following a plan" (Beck, Beedle, Van Bennekum, et al., 2001).

With the values it highlights the importance of using light and sufficient methods to minimize time-consuming and very costly software processes such as detailed planning sessions, which end up with substantial documentation (Qumer & Henderson-Sellers, 2006). More importantly, Agile Methodology tries to manage volatile and changing customer requirements effectively, by making the customer part of the development team (Conboy, Wang, & Fitzgerald, 2009). In-order to gain these benefits, Agile comprises of unique characteristics. According to the Agile Manifesto, there are 12 principals upon which Agile is constructed on. The following seven characteristics have been identified as the salient features of Agile.

Modularity

Agile is identified as a modular process (Abrahamsson, 2002; Miller, 2001). When a process has modularity, it allows the process to be broken-down into smaller pieces called activities (Miller, 2001). Having many activities to work on means Agile allows the development team to pick and work on activities, which helps the team to achieve an effective and a fast output (Ming et al., 2004).

Iterative

Being iterative, an Agile process has the ability of acknowledging and correcting if something is done incorrectly, or deviated from its purpose (Miller, 2001). In-order to support this, Agile has short-development cycles of repetitive nature, covering towards a deliverable output (Cohen et al., 2003). Each of these cycles has some amount of activities that lean towards achieving high quality working software. These cycles (often referred to as “iterations”) are in weeks (2-3 weeks per iteration is typical), and a number of iterations will be required to achieve the project deliverables (Ming et al., 2004).

Incremental

Having iterative cycles (often named as “Sprints”) will always help Agile projects to build the deliverables incrementally (Miller, 2001). Agile does not target at building the required solution at once; it carefully partitions the increments and work paralleled to achieve the deliverables (Miller, 2001). Independently built partitions will then be integrated to a singular solution, through testing and high quality (Awad, 2005).

Adaptive

Adaptiveness is one of the key characteristic of an Agile process, which made Agile popular among software professionals (Miller, 2001). Due to the changing nature of the requirements, it is possible to have changes exposed in the cycle that require additional activities that are not planed (Cohen et al., 2003). Agile has the capability and flexibility of accommodating such changes and if the goal cannot be achieved during iteration new activities are planned to reach to the goal (Miller, 2001).

Time-bound

Since Agile relies on iterations, work can be planned properly and accurately (Miller, 2001). Therefore, scheduling the activities for each cycle is more accurate and achievable. Functionalities can be reduced or activities can be re-scheduled if the resources are over-committed in the allocated time (Miller, 2001).

People oriented

It is obvious from the values of the Agile Manifesto, that Agile favors people over technology and processes (Beck, Beedle, Van Bennekum, et al., 2001). Structure of the Agile respects thoughts of the people and people are given freedom to achieve their goals without being overly pressurized (e.g. the developers get a chance to work on an item that they have chosen and decided; this increases the productivity and quality (Ming et al., 2004). Consequently, Agile produces committed and upskilled employees for the organization e (Miller, 2001).

Collaborative

Communication is identified a vital part in Agile (Miller, 2001). When each component is developed separately, collaboration of the teams is highly important towards integrating the right product (Miller, 2001). Being a lightweight process, Agile empowers high degree of collaboration (Cohen et al., 2003). Figure 2.2 depicts how Agile works in encompassing the aforesaid characteristics.

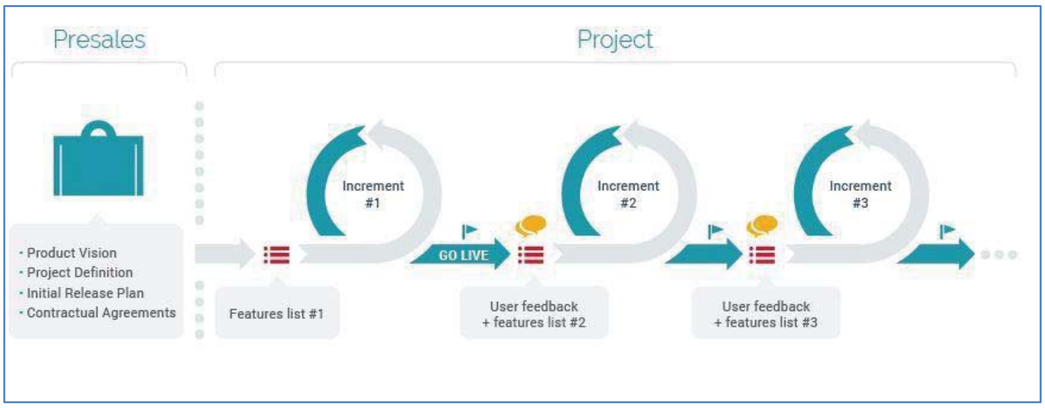


Figure 2.2: How Agile works (Source: Esker, 2008)

As evidenced in Figure 2.2, requirements and product vision can be obtained from customers who are often handled by pre-sales. Based on the requirements, increments can be planned. At the end of an increment, user feedback is obtained and with the user feedback, the next iteration is planned and executed. A number of iterations will be carried out up until the product is delivered.

Agile methodology is the umbrella for many other software development methodologies being practiced around the world. Next section describes various Agile Methodologies widely used by software professionals.

2.2.2 Various Agile Methodologies

Based on the main characteristics of Agile, a number of methodologies have been introduced under the hood of Agile. Adaptive Software Development (ASD), Agile Modeling, Crystal Methods, Dynamic System Development, Extreme Programming (XP), Lean Development, and Scrum are some of them (Awad, 2005). As these methods follow Agile characteristics, they usually result in high quality software and customer satisfaction (Awad, 2005).

Following table (Table 2.1) shows the differences between certain Agile methods that are being widely used around the world.

Table 2.1: Agile Methods (Sources: Awad, 2005; Beck, Beedle, Van Bennekum, et al., 2001)

Extreme Programming (XP)	- Widely used as an object-oriented development approach (Kleppe, Warmer, Bast, & Explained, 2003). - Short development cycles and evolutionary design - Continuous feedback and incremental planning - Has four phases Planning, Designing, Coding and Test (Beck, Beedle, Van Bennekum, et al., 2001) - Members of the team spend few minutes each day on programming, project management, designing, feedback and team building activities (Williams, 2003).
Adaptive Software development (ASD)	- This is a popular technique for developing complex software solutions. - Collaboration of the team and self-organizing team are the key ingredients of ASD. - Popular in component-based development (Kleppe et al., 2003). - Has three phases, which are speculation (planning), collaboration (requirements gathering), learning (implementation/test/gather feedback/tech reviews); and release happens after making adjustments to the increments (Beck, Beedle, Van Bennekum, et al., 2001).
Dynamic System Development	- This is well suited for projects with tight-time lines. - Active user involvement is considered as a key aspect (Beck, Beedle, Van Bennekum, et al., 2001). - Only adequate work is done in each iteration to support the movement of next iteration (Beck, Beedle, Van Bennekum, et al., 2001).
Scrum	- Scrum is the most widely spread variation of Agile (Schwaber & Beedle, 2002). - Scrum works well for the projects that have tight schedules, often changing customer requirements and critical business goals (Beck, Beedle, Van Bennekum, et al., 2001).

	- Work items are carried out in iterations called ‘sprints’ and the work is demonstrated to the customer’s sprint end ‘demos’ (Beck, Beedle, Van Bennekum, et al., 2001).
Crystal	- This method distinguishes the features and use workshops to review the work habits of the team (Beck, Beedle, Van Bennekum, et al., 2001). - Face-face communication is highly emphasized.
Feature Driven Development	- This is identified as a practical methodology for object-oriented programing (Kleppe et al., 2003). - Empathizes on feature at most, therefore design and development is done for a feature. (Beck, Beedle, Van Bennekum, et al., 2001)

Since this research study mainly focuses on Scrum, the next section describes how Scrum works in a software development environment.

2.2.3 Adopting Scrum as a Software Development Methodology

Scrum is an iterative and incremental process which is used to develop products or manage work (Rising & Janoff, 2000). Scrum focuses on, how a team should function in order to achieve the potential set of functions in the end of the sprint, by being flexible in changing environments. Scrum is well known as a framework within which people can address complex adaptive problems, yet delivering products with highest possible value, productively and creatively (Schwaber & Sutherland, 2013).

One of the key reasons for Scrum being popular is that it is not tied to any specific software development method or practice. Nonetheless, Scrum requires certain amount of management practices and tools being used to avoid issues that rise with complexity and unpredictability (Cristal, Wildt, & Prikladnicki, 2008).

According to the so-called “Scrum guide” (Schwaber & Sutherland, 2013), a Scrum team consists with 3 roles: the product owner, Scrum master and the development team. Each of these roles is described in turn.

Product owner is a person, who is responsible of managing the product backlog (product backlog is described in next section) (Schwaber & Sutherland, 2013). The duty of the product owner is to clearly state the vision of the product and help the development team to sort out the priorities of the work items (Schwaber & Sutherland, 2013). By using a clear and visible product backlog, product owners make sure that the vision and the items of the backlog are clearly understood by the development team. Since the product owner is responsible for the product, once a product related decision is made, no-one is allowed to derail the development team from that decision. Therefore, the development team is not allowed to act on others' decisions (Schwaber & Sutherland, 2013).

As the name implies, the Scrum master is the person who is always considered to be the most knowledgeable on Scrum practices, in a Scrum team (Schwaber & Sutherland, 2013). Scrum master can always help the team to remove impediments on the processes that they follow or act as a "servant-leader" to maximize the value created by the Scrum team (Schwaber & Sutherland, 2013). Scrum masters coach the teams to become cross-functional and self-organizing. At the same time, the Scrum masters help the members of the team to progress in their career through developing the soft skills of the development team (Schwaber & Sutherland, 2013). Also, the Scrum master helps the product owner to manage the backlog, facilitate meeting with the team where necessary, and manage between the team and the product owner (Schwaber & Sutherland, 2013)

The development teams in Scrum are said to be self-organizing and cross-functional teams (Schwaber & Sutherland, 2013). Being self-organizing, the teams gain confidence in finishing their work, giving their best effort, rather than being pushed by a manager (Ktata & Lévesque, 2010). With cross-functional teams, teams gain enough knowledge and skills to complete their work without being dependent by anyone else (Sutherland, Viktorov, Blount, & Puntikov, 2007). This does not mean that the development team consists with all the development roles in software development, to the extent that they have necessary skills to accomplish their tasks. In Scrum, size of the development team is always kept small, typically between 1-6 members. A very small team is not productive and a very large team is hard to handle, thus requiring the right compromise (Schwaber & Sutherland, 2013). The Scrum master and product owner are by definition part of the development team, although depending on the organizational structure and the culture, this need not always be the case (Rising & Janoff, 2000).

Scrum has no rigid rules about what the size should be and who should be included in the team.

Since Agile, and therefore Scrum, emphasizes more on “working software”, rather than documentation, Scrum has only defined three main artifacts— product backlog, sprint backlog and increments—because key documents or reports can be maintained and made visible to the team and outside parties whoever is interested (Schwaber & Beedle, 2002).

The product backlog is an “ordered list of everything that might be needed in the product” and it is the “single source of requirements for any changes to be made to the product” (Schwaber & Sutherland, 2013). Product owner is the main person who is responsible with the backlog. The product owner studies the market trends, demands and customer requirements and add items to the product backlog (Sutherland, 2004). The items in a product backlog can be requirements, features, functions, enhancements or fixes which will result in making product changes in future releases (Rising & Janoff, 2000). Most Product owners, make sure of adding basic description of the task, priority and estimation to the item, as it will provide an overview of the task to the development team (Rising & Janoff, 2000). Product backlog never shrinks or dies, as-long-as the product exists; instead, product backlog will enhance with new requirements or fixes. Having a product backlog for each product will define the future of the product and based on the priorities, development teams pick up items from the product backlog to create their sprint backlog (Schwaber & Sutherland, 2013).

The sprint backlog is created out of the items selected from product backlog for the sprint (Schwaber & Sutherland, 2013). Most importantly, sprint backlog has a plan on how to achieve the sprint goal (Schwaber & Beedle, 2002). Sprint backlog contains adequate detail on the items in the sprint for the development team to start achieving it. The sprint backlog is more likely to be modified during the sprint by the development team (Pino, Pedreira, García, Luaces, & Piattini, 2010). During the sprint, the development team follows the sprint plan and understands more about the work required to be completed, in-order to achieve the sprint goal. Achieving sprint goal contributes towards the increments of the final deliverable (Schwaber & Sutherland, 2013).

The increment is the completed portion of work in each sprint, which should be in “usable” condition (Schwaber & Sutherland, 2013). End of the sprint, the increment should meet the “definition of done” in the development team, which means it is in usable condition, based on the team’s definition (Schwaber & Beedle, 2002). However, regardless of the decision of the product owner, the increment should be in a releasable condition. This means minimal functions should be working, tested and documented (Schwaber & Beedle, 2002).

2.3 Healthcare Software Development

2.3.1 Healthcare Software

With the acceleration of the digital age, healthcare industry is increasingly looking towards adaptation of IT solutions to ease the work and eliminate the use of paper based medical records, prescriptions, faxed referrals, and paper filled cabinets (Pagliar, 2007; Yunan, 2013). Instead of slavishly following the directions of the general practitioners (GPs), patients are more actively participating in making medical decision process as never before by informing themselves about their illnesses and managing their health actively (Levinson, Kao, Kuby, & Thisted, 2005). Technology including internet plays a very important role in this rapid change as it can facilitate broad knowledge and service among wide population. Therefore, more and more health care providers are taking aid of healthcare software to improve patient care (Jepsen & Jepsen, 2008). Patients with computer knowledge consume the medical applications to maintain better communication with their general practitioner (GP)/specialist, learn and share health records. More importantly, patients can take actions to improve the quality of their lives much the same way as they use online-banking or vacation planning (Dehling & Sunyaev, 2014; Forkner-Dunn, 2003; HealthIT, 2013).

Technology-based healthcare applications have become popular worldwide, leading to various application types such as follows (Forkner-Dunn, 2003; HealthIT, 2013).

1. **Electronic health records (EHRs)**
   Electronic systems that store patient’s health information are referred to as EHRs. These allow GPs to easily keep track of their patient's health information to enable them to access the information, when a patient makes a medical compliant; EHTs even allow

remote access (HealthIT, 2013). An HER also makes easier for the doctor to share information with medical specialists and other relevant parties (subject to patient consent) to arrange efficient and effective treatment (McConnell, 2003; Pagliar, 2007).

2. **Personal health records (PHRs)**

   Patients can use a PHR to keep track of information from their GP visits (Dehling & Sunyaev, 2014). Moreover, PHR can also reflect their life outside the doctor's office and their health priorities, such as tracking the food intake, exercise, and blood pressure. Sometimes the PHR can link the records with the doctor's HER which maintains better synchronization of the information (Frost & Massagli, 2008; HealthIT, 2013).

3. **E-prescribing**

   Electronic prescribing (E-prescribing) provides an advanced direction to the paper-based prescriptions by replacing them electronically (HealthIT, 2013). This allows doctor to communicate directly with the pharmacy, which means a patient can go to the pharmacy to pick up medicine without having to bring along the paper prescription (Gagnon & Scott, 2005).

4. **Health Information Exchange (HIE)**

   HIE allows doctors, nurses, pharmacists, other health care providers and patients to appropriately access and securely share a patient's vital medical information electronically, by improving the speed, quality, safety and cost of patient care (HealthIT, 2013; Mandi & Lee, 2002).

When developing healthcare solutions, it is understood that information privacy and security will be preserved at all times (Dehling & Sunyaev, 2014). Information privacy and security are the most important requirements a customer expects in service delivery and catering to this requirement falls into hands of the health software vendor. In fact, to successfully satisfy the customer needs, one of the major responsibilities of the health software development teams is to preserve information privacy and security (Frost & Massagli, 2008). The natural tendency of a human being to find out and talk about others around them and for this reason it is not surprising that often, the information privacy and security requirements are being overlooked at the beginning of a project (Armour & Thizy, 2010).

Table 2.2 depicts key requirements of healthcare software. These requirements have been identified by Dehling and Sunyaev (2014) and Karsh (2004), based on case studies.

Table 2.2: Key Requirements of Healthcare Software (Sources: Dehling & Sunyaev, 2014; Karsh, 2004)

Requirement	**Description**
Confidentiality	No unauthorized person must be able to access users' (patients') information
Anonymity	The real identity of users (patients) must not be revealed without patient consent
Authorization	Access must be limited to necessary information and data segregation must be ensured
Limited Access	Unnecessary access rights must be revoked
Non-Disclosure	It cannot be possible to force users (patients) to reveal information they do not to reveal
Transmission and Storage Security	Eavesdropping has to be prevented during transmission and storage
Un-linkability	It must not be possible to reveal relationships between items through observation
Availability	Up-to-date information must be available whenever needed
Integrity	It must be ensured that information content is as intended and not unintentionally changed
Backup	Redundancy must be employed to ensure that data can be restored.
Long Storage times	It must be possible to store information as long as it is required (even a lifetime or longer)
Scalability	Patient health system has to be adaptable to changing performance needs
Resilience to Failures	Failure of single nodes must not impede the

Requirement	**Description**
	performance of the whole service
Recoverability	It must be possible to restore lost information to a specific point in time
Accountability/Non- Repudiation	Accesses to and uses of information must be attributed to the respective party and it must not be possible to any such actions afterwards
Audit Trails	Relevant activity (e.g. document accesses) must be logged
Authentication	It must be determined who is using the software and verified that they are who they claim to be
Perimeter Definition	The boundaries of trusted access to the information system must be known and controlled
Network Security	Unauthorized access must be avoided and access rights must be managed
Physical Hardware Security	Impairment of hardware (theft, natural disasters) has to be presented.
System Vulnerability Analysis	System vulnerabilities must be detected
Intrusion Detection	Unintended actions/IS activity must be detected
Usability	Important information has to be easily accessible
Access Control	Users (patients) have to be able to control who can access what information
Patient Access	Users (patients) have to be able to retrieve information stored on them
Informed Consent	Users (patients) have to agree to uses of their information and patient consent must be managed.
Education, Alerts, and Reminders	User ethics, obligations, and proficiency must be reinforced

Requirement	**Description**
Credential Substitutability	Authorization details must be substitutable (loss, technological obsolescence)

If the software is unable to meet the requirement(s) mentioned above (Table 2.2), it can increase the risk of failing customer expectations in the production environment and may even harm the users (Dehling & Sunyaev, 2014). Although it is high unlikely to achieve all the requirements mentioned above, health care vendors should at least try to achieve as many of them to protect information security and privacy (Dehling & Sunyaev, 2014).

2.3.2 Issues that Arise in Health-IT Development

Similar to most IT industries, health-software vendors are increasingly moving towards iterative software development (Jepsen & Jepsen, 2008). This is due to the flexibility of the development process, which can address the rapidly changing requirements in the healthcare domain. However, the requirements to be captured in health software development are very sensitive, and the process following to develop the health software systems must be sufficiently sophisticated. Therefore, the process of software development is considered to be very complex compared to other software development processes (Karsh, 2004). Moreover, the health standards defined by world class health regulatory organizations, such as the US food and drug administration (FDA) and integrating the healthcare enterprise in Europe (IHE) for the products are always lengthy and complex, which means software developers need to dedicate considerable time and effort to understand and interpret the regulations correctly (Rasmussen, Hughes, Jenks, & Skach, 2009).

In an environment of Scrum, the requirements are captured more regularly and the stakeholders are kept engaged in the development process (Schwaber & Beedle, 2002). However, as mentioned above, due to the nature of the complexity of the healthcare domain, issues still arise in the development process (Schwaber & Beedle, 2002).

Not tailoring to the requirements based on different situations are identified to be one of the major issues in health software development (Armour & Thizy, 2010). Requirement of a ‘GP can be different from a medical specialist or medical consultant. Considering a usual clinic run by a GP, the software and records should support the nature of the consultation that happens between GP and the patient. Also, the software should be able to handle daily patient queues. However, the precise scheduling is not very required (Armour & Thizy, 2010; Gagnon & Scott, 2005). Unlike a GP, a medical consultant in a hospital requires software that allows not only capturing patient data, but also evaluating outcomes for patients with particular illness in order to foresee and detect trends in relation to treatment (Armour & Thizy, 2010; Gagnon & Scott, 2005).

Another identified major issue is that *non-functional requirements tend to be overlooked* in the requirements gathering stage (Armour & Thizy, 2010). This can either be due to not having a proper requirement checklist or fewer stages of requirements assessment. Patient and information privacy is a key piece of mandatory non-functional requirement (Pagliar, 2007). To promote the awareness among healthcare software vendors, governments have created laws (Health Insurance Portability and Accountability Act of 1996-HIPAA (Personal Health Information Protection Act) which impose penalties over organizations which has privacy vulnerabilities (Armour & Thizy, 2010). Privacy laws are very important as most people wish to have their medical records privately. Most hospital staff have access to patient records and where there are no strict privacy rules being imposed upon in software development and usage; as such, patient information can be accessed and shared inappropriately (Karsh, 2004). According to a case-study conducted by (Armour & Thizy, 2010), there have been numerous privacy breaches lately, thus emphasizing the importance of having privacy rules in health software systems. The following three examples of disciplinary actions taken by health authorities provide a general flavor of privacy breaches that have taken place in the United States.

- Hospital workers being fired for looking at popstar Britney Spears’s Medical Records
- Sixteen hospital workers in Houston being fired for HIPAA violation
- Twenty hospital workers being fired for viewing former American football player (who was gun down by an assassin) Richard Collier’s medical records

The next issue is more in-line with the practices of the primary case study company (Scrum Health), which is, *actual writing of the code beginning before identifying the actual work-flow of the business (or hospital)* (Momtahan, Lloyd, & Simpson, 2007).

Usually, the traditional method of software development begins with requirement gathering, which incorporates copious documented notes and the diagrams (Ming et al., 2004). With the flexibility of Scrum, it is documented in the literature that development teams often miss-interprets the guidelines and build their own processes, causing issues (Ming et al., 2004; Momtahan et al., 2007). Armour and Thizy (2010) assert that current business process (or hospital process) must be understood and model properly before designing a new solution as most hospitals still follow the typical processes for admit, diagnose, treatments and discharge. They argue that if the software solution is impacting on changing those processes, this can result in frustrated users moving back to old paper-based ways (Armour & Thizy, 2010).

Yet another major issue in current software applications in health is that the *application does not properly cater to the time constraints within which healthcare professionals work* (Anderson & Agarwal, 2011). It is documented in the literature that often, the development team does not consider the efficiency and the performance of the product, when the software is being built, either due to the lack of knowledge they have about the real-life health profession or due to negligence (Anderson & Agarwal, 2011; Armour & Thizy, 2010). With the growing number of patients in the waiting list, health practitioners cannot afford to be waiting till the software is being developed, if the development team takes excessive time to respond. It is documented in the literature that efficiency and performance of software are of paramount importance in software development (Anderson & Agarwal, 2011; Armour & Thizy, 2010).

Jepsen and Jepsen (2008) observe that one of the most frequently received customer complaints about health software delivery and use is on the user manuals. Jespen and Jespen assert that development teams tend to produce very complex and technical user guides for practitioners, on the assumption that users are technically competent on technology. Armour and Thizy (2010) content that apart from doctors and nurses, the hospital administrators and executives, middle managers (each department/clinic has a manager), physician-run office worker, or independent researchers can become users of the health applications at any time. Jepsen and Jepsen (2008) highlight that the documents and guidelines being provided should address the workflows being performed by all the user groups; they also assert that these should be as simple as possible to become easy to understand. They also highlight that sometimes the systems are only catering the basic flows being performed by doctors and nurses and that other scenarios are being left out. It is being argued that when designing a solution, all the direct and indirect (such as reporting and analytics) user scenarios should be considered and noted (Armour & Thizy, 2010; Heeager, 2012).

Due to the changing nature of the IT industry, it is often seen that employees are moving around different industries, which leads the projects being completed un-successfully (Jepsen & Jepsen, 2008). It has been noted that in most companies the so-called "stars" in the teams seem to have more knowledge on a product and "new-starts" find it hard to extract details of the products due to less reference places (Jepsen & Jepsen, 2008; Momtahan et al., 2007). Jepsen and Jepsen (2008) observe that once a "star" leaves a company, the projects get jeopardized due lack of knowledge. This often leads to re-requirement gathering and design sessions, which is time consuming adding more headache to the stakeholders such as healthcare professionals (Jepsen & Jepsen, 2008). It is also identified that developers with less healthcare experience put more emphasis on verification rather than validation (Armour & Thizy, 2010). Software can always do what developers wanted it to do (this is known as *verification*), however it is somewhat harder to make sure if you are building the right solution for the job (this is known as *validation*) (Armour & Thizy, 2010). It is documented that due to lack of knowledge, teams tend to perform testing around validation (e.g. 'crashing a software') but put less effort on testing actual scenarios to determine if anything is missed or implemented incorrectly rather than the actual scenario. This leads to a state where more software solutions are being rejected in the "user acceptance testing" phase due to incorrect functionality (Armour & Thizy, 2010; Rasmussen et al., 2009).

Cho (2008) asserts that with the nature of the Scrum practices, more development teams find it harder to manage the evolving requirements in a traceable way. It is documented that due to lack of knowledge of practices and processes, development teams often presume that Scrum means having no-documentation, and as a consequence, key documentation being left-out (Cho, 2008; Ratanotayanon, Kotak, & Sim, 2006). It is also documented that in a complex domain such as health software, this is a major issue as most of the time, development teams find it hard to trace a new function or a change with its original requirement (Akif & Majeed, 2012; Armour & Thizy, 2010).

The aforementioned issues identified in the literature highlights lack of knowledge and poor practices being adopted by development teams. This creates poor outcomes in health software development, including requirements, design and test artifacts. Not having proper requirements gathering mechanism leads to insufficient requirements, not maintaining the collected requirements, and not producing artifacts properly; less knowledge about actual business flows and scenarios advocates that development teams do not have proper guidance and control over the processes that they are following. Though Scrum is flexible and often welcomes changes, having the aforementioned issues suggest less knowledge and myth on Scrum. Heeager (2012) states that the most common myth about Scrum is thinking that Scrum means ‘no-documentation’, which affects negatively on documentation preparation and maintenance, subsequently leads to missed requirements and lack of guidance.

Heeager (2012) asserts that in Scrum, often the development teams do not receive guidance and control over how the things should happen in product delivery life cycle because Scrum has very little or no mechanism for formalize process and procedures on such guidance. Lee and Xia (2010) observes that in Scrum, autonomy and cross functional teams are given the highest priority and formal processes and the guidelines are given the least priority. Likewise, Schwaber and Beedle (2002) observe that coding and actual product is given highest priority, keeping documentation to a minimum as much as possible. However, these arguments do not mean that Scrum does not and should not include any documentation. As Elke and Roland (2008) point out, development teams do need to be emphasized that while documentation is not the imperative in Scrum, some level of documentation needs to be maintained of a successful implementation of Scrum.

Ambler (2012) contend that tasks for documentation can be estimated, prioritized, and treats as a work item that stacks along with all other work items that the development team must address. Ratanotayanon et al. (2006) as well as Sutherland (2004) observed that not having a documented process for development teams to follow has found to be a frequent cause for product failure and customer dissatisfaction. Momtahan et al. (2007) well as Rottier and Rodrigues (2008) have argued that maintaining a good set-of documented requirements can help to share the product knowledge equally among all team members, making the output/outcome less relent on people—the argument being, people can come and go but the system should remain.

Armour and Thizy (2010) highlight that as the user guides become more technical and complex, customers get frustrated, especially in healthcare applications because the users are skilled in healthcare but not in technology. Armour and Thizy go on to argue that when the guidelines become hard to refer to, the information system can be perceived by users as a liability than an asset. According to some authors (e.g. (Ambler, 2012; Pagliar, 2007) documents such as user manuals produced for outsiders should address the need of the project stakeholders; there should also be a mechanism of communication with an external group; the documents should support the users; they should guide the whole team to follow a slandered way-of doing; they (documentation) should also serve as a start-up guide for new developers.

Therefore, it is documented in the IT literature that in-order to mitigate the issues the development teams can produce and use of user manuals, documented processes and documented product features (e.g. requirements, risk analysis reports, traceability reports, bug tickets, or any other measurable and readable document related to product) (Rüping, 2003; Tony, 2003). Rüping (2003) also mentions that instead of lengthy, very complex, bored to read documents, developers should adopt short and concise, unsophisticated documents to impress the users. Mansour (2013) mentions that hhaving standardized ideas and results bundled together for referencing purpose will always add higher visibility to the work and development teams can get extra benefits out of it. Elke and Roland (2008) mentions that production of essential/useful documentation (as opposed to no documentation) by the development teams will help minimize the risk of systems being developed to become non-useful legacy products. The next section looks at ISO 9001:2008 quality management system standard, especially in relation to documentation requirements of accreditation.

2.4 ISO 9001:2008 Quality Management System Standard

As argued in the previous section, to mitigate the issues arising in software development environments, it is important to have a well-defined methodology for the employees to follow. It is acknowledged that a good approach can lead and operate organizations successfully by directing and controlling in a systematic and transparent manner (De Vries, 2001; Juran & Gryna, 1988). Juran and Gryna (1988) assert that management systems that are designed to continually improve performance while addressing the needs of all interested parties can deliver successful results for the organization. It has been acknowledged that in selecting an appropriate management system to an organization, satisfying and enhancing customer needs is the key (Juran & Gryna, 1988; Schlickman, 2003). As such, management systems based on the concept “Total Quality Management” (TQM) such as ISO 9001:2008 has gained very wide acceptance in the industry (Chang & Lee, 2005; Oakland, 2013; Sampaio, Saraiva, & Guimarães Rodrigues, 2009). TQM philosophy is based on the premise that customer satisfaction can be achieved by maintaining quality of the products and/ or services delivered by an organization, by integrates efforts of everyone throughout the organization. Harvey (1998) stated alternatively, managing an organization for customer satisfaction—which results in long-term sustainability and growth of the organization—emphases TQM, acknowledging the importance of other management systems (e.g. the human resource management subsystem, the financial subsystem system) in place in the organization (Australian and New Zealand Standard, 2006; Juran & Gryna, 1988).

As per ISO 8402 vocabulary, quality management is defined as “all activities of the overall management function that determine the quality policy, objectives and responsibilities, and implement them by means such as quality planning, quality control, quality assurance and quality improvement, within the quality system” (ISO, 1994). As per ISO 8402 vocabulary, quality system is defined as "the organizational structure, responsibilities, procedures and resources needed to implement quality management" (ISO, 1994). By considering the two definitions, QMS can be interpreted as interaction and combination of people, processes and documentation to satisfy customer needs by having improved practices and processes which result in reducing waste, cost and inefficiencies (Mohammed, 2006).

Vast number of companies around the world are trying to follow ISO 9001:2008 quality management system (QMS) standard (this standard specifies the requirements for accreditation) for continuous improvement and to provide the customer the assurance that they (the organizations) have systems in place to design, develop, deliver and maintain quality products and services for the customer (Oakland, 2013; Sampaio et al., 2009). In this study ISO 9001:2008 QMS standard is chosen as the primary quality management standard. Hence the recommendations to address the issues arise in Scrum projects developing healthcare software solutions are based on ISO 9001:2008, especially the documentation requirements stipulated in the standard for ISO 9001:2008 accreditation.

ISO 9000 is an international quality management system (QMS) standard that consists of series of standards. ISO 9000 covers the "Fundamentals and Vocabulary" used in the series of standards, including the certification standard ISO 9001. The certification standard ISO 9001 stipulates the "QMS Requirements" for accreditation against the said standard. Finally, ISO 9004 provides "Guidelines for Performance Improvement" which essentially serves as a supplement for the certification standard ISO 9001, in the sense, organizations that wish to pursue TQM to obtain ISO 9001 accreditation and beyond, are given guidelines as to how their organization could be improved to the requisite level (Australian and New Zealand Standard, 2006; Cianfrani, Tsiakals, & West, 2009).

ISO 9001 is chosen as a quality management system model over one million certified organizations in more than 100 countries (Oakland, 2013). There is empirical evidence to support the notion that the operating effectiveness and efficiency of organizations improve after implementation of ISO 9001 (i.e. after gaining the accreditation/certification) (Cianfrani et al., 2009; Oakland, 2013). In addition, ISO 9001 has that potential to please the customers because the standard serves as an *assurance* that the organization from which the product/service was purchased, contains the necessary ingredients for planning, designing, manufacturing, delivering and servicing the product/service the customers purchased (Hooper et al., 2001). Having ISO 9001 being implemented in an organization provides a better understanding to managers on what to do and how to do it; it provides controls to assure that the operational activities meet the requirements for quality; it provides a basis for quality control and continuous improvement in that when the processes are out of control, resulting in nonconformities, the standard requires managers to take corrective and preventive action; finally, the standard provides an increased opportunity to communicate problems in a non-threatening manner providing an environment where people are not blamed for issues that can only be resolved by the managers (Cianfrani et al., 2009; Mohammed, 2006). The researcher notes that much of the quality control and quality improvement concepts on which ISO 9001 hinges upon is based on the principles advanced by quality guru Edwards W. Deming. For example, Deming distinguished between special cause variation and common cause variation associated with products and services (Summers, 2010). Special cause variation occurs when a process is subjected to a cause/s that is not inherent in the system (e.g. use of defective raw material), which according to Deming, should be handled by frontline workers taking action to correct the process (e.g. stop the plant); common cause variation on the other hand is the variation inherent in the system; this variation (if excessive), according to Deming, requires management intervention due to the capital intensive nature of the improvement required (Summers, 2010).

It is important to note that the ISO 9001:2008 QMS standard, which previously existed as ISO 9001:2000 (between the year 2000 through to 2008) and ISO 9001:1994 (there were also ISO 9002:1994 and ISO 9003:1994 which covered less scope than ISO 9001:1994) was originally proposed (prior to ISO 9001:2000) for the manufacturing industry. Therefore, applying this standard to software development still requires much understanding and effort (Coleman & O’Connor, 2008; McMichael & Lombardi, 2007; Stalhane & Hanssen, 2008). In the next section, the QMS requirements are interpreted for software development almost in accordance with ISO 9000:2008, most specifically its documentation requirements stipulated in the standard.

The requirements in the standard has categorized in to 20 main quality attributes which are seen as major parts of a QMS with well-known concepts (Sampaio et al., 2009). Also in the context of the software development some software requirements are covered by multiple requirements in the ISO 9001:2008 standard (Sampaio et al., 2009). For an example, documentation requirements for testing can be covered by either 4.2.3 "Control of documents", 4.2.4 "Control of records”, 7.3.6 "Design and development validation", or 7.1 "Inspection and testing activities" (Australian and New Zealand Standard, 2006).

2.4.1 ISO 9001:2008 Documentation Requirements

In a strict sense, a quality management system of an organization (e.g. a quality system that is accredited against ISO 9001) has no one on one relationship with documentation. A quality management system is a management system that contains the necessary ingredients for designing, developing, producing, delivering and maintaining a product or service that meets customer expectations (Leung, Liao, & Qu, 2007). These aspects are covered in the four final clauses of ISO 9001: Management Responsibility (Clause 5), Resource Management (Clause 6), Product Realization (Clause 7), and Measurement, Analysis, and Improvement (Clause 8) (AS/NZS, 2000). From the point of view of the organization, documentation facilitates effective communication and knowledge sharing. From an auditing perspective, documentation provides tangible evidence to the auditor on the presence or absence of systems in place on the quality imperatives of planning (P), doing (D), checking (C) and acting (A), which are subsumed in aforementioned four clauses (Sinha & Hernandez, 2010; Tricker, 2014). Based on the objective of an organization, ISO 9001:2008 categorizes the documentation requirements (sub clause 4.1 of Clause 4) into three main parts:

1. ***Communication of Information*** – Use as a tool for information transmission and communication (Australian and New Zealand Standard, 2006; ISO, 2009).
2. ***Evidence of conformity*** - Provision of evidence that what was planned, has actually been done (Australian and New Zealand Standard, 2006; ISO, 2009).
3. ***Knowledge sharing*** – Use to distribute and preserve the organization's experiences. A typical example would be a technical specification, which can be used as a base for design and development of a new product (Australian and New Zealand Standard, 2006; ISO, 2009).

It has been argued that the type and the content of the documentation depends on the nature and the culture of the organization's products and processes (Australian and New Zealand Standard, 2006; Wright, 2003). It has also been argued that the degree of formality of communication systems and the level of communication skills within the organization are major factors in determining the required documentation in an organization (Stalhane & Hanssen, 2008, 2009). The ISO 9000 series of standards stipulate that documents can be in any form, such as paper, magnetic, photograph, electronic or optical computer disc and master sample (Australian and New Zealand Standard, 2006).

Prior versions of the standard define documentation as a "way of enabling communication between parties" (ISO, 1994). There is no evidence in the literature to the contrary of this definition. It is documented in the literature that meeting documentation requirements contributes towards confirming that customer requirements and quality improvements are achieved; repeatability and traceability of the work being done; provides appropriate training and work instructions and evaluate the efficiency of process follows (Australian and New Zealand Standard, 2006).

Rüping (2003) provides examples of useful documents for Scrum (Table 2.3), in the light of documentation requirements stipulated in ISO 9001 standard.[3]

Table 2.3: Useful documentation in software development; (Adopted from: Australian and New Zealand Standard, 2006; Rüping, 2003)

Documentation Requirement as Defined in ISO 9001:2008	**Description**	**Sample Document (Used in Scrum)**
Quality manual	Provides information related to organizational QMS	Organizational Quality procedures.
Quality Plan	How QMS apply to a specific project /product	Internal and external standards
Specifications	Requirement document	Requirements specification
Guidelines	Recommendations or suggestions	Test Plans
Records	Results of the tasks)	Source code files, Test Results
Documented procedures, work instructions and design document		Design documents, Work procedures

The sub clause 4.2 of ISO 9001:2008 specifically relates to the documentation requirements of the QMS system, which consists with four parts (Table 2.4).

[3] Rüping (2003) referred to the year 2000 version of ISO 9001 QMS standard (i.e. ISO 9001:2000), which is almost identical to ISO 9001:2008.

Table 2.4: ISO 9001:2008 Sub Clause 4.2 (Source: (ISO, 2009)

Clause	Documentation Requirement Type	Description
4.2.1	General requirements	Specifies what is to be included in QMS documentations such as Quality policy, Objectives and Manual (Myhrberg, 2009).
4.2.2	Quality manual	Quality manual defines the full scope of the requirements, specifying what some factors are in-scope or out-of-scope (Myhrberg, 2009).
4.2.3	Control of documents	Define necessary processes to ensure that documents are in control including document approval, review, updating and re-approval, legibility availability of documents, revision status and identification (Myhrberg, 2009).
4.2.4	Control of records	To ensure that QMS is operating as intended and effective, addition to 4.2.3, necessary processes has to be followed to control records. The controls will apply to quality record and specify among other things such as the storage, retrieval, maintenance time and disposition (Myhrberg, 2009).

2.4.1.1 General Requirements

General requirements stipulated in ISO 9001:2008 (sub clause 4.2.1), are useful to identify the documentation requirements in each stage of the development cycle in Scrum. The researcher observes that because ISO 9000 series of standards define documentation to be “in any form or type of medium" (ISO, 2009), the ISO 9001:2008 QMS standard can be used by the quality team to highlight the purpose and the conformance of each stage in a Scrum project. Rüping (2003) observes that while Agile emphasizes on minimum documentation, from a quality assurance point of view and learning, documentation is relevant to the purpose of the organization (or for the event) because it induces a commitment to comply with the requirements of ISO 9001:2008 and continually improve the effectiveness of the QMS. Similarly, Cianfrani et al. (2009) assert that documentation requirements stipulated in ISO 9001:2008 provides a framework for establishing, reviewing and communicating quality objectives with a development team or the organization. While acknowledging the importance documentation, Osteen and Robbert (1995) argued that documents and data shall continuously be reviewed and approved before publishing to its audience.

2.4.1.2 Quality Manual

ISO 9001:2008 stipulates that format and the structure of the quality manual is a decision for each organization as it depends on the size, culture and complexity of an organization (ISO, 2009). Tricker (2014), Schlickman (2003) as well as Cianfrani et al. (2009) highlight that for small organizations, it is more appropriate to include the description of its entire QMS within a single manual, including quality policies, quality objectives, documentation of interaction of processes (these are high level documents) as well as operational documents such as standard operating procedures, quality plans, work instructions (documented procedures) and forms, templates and similar required by the standard, whereas for large multi organizations it may be necessary to create a hierarchy of documentation, ranging from top level documents (quality policy and quality objectives) through to operational-level documents used in quality assurance (Figure 2.3).

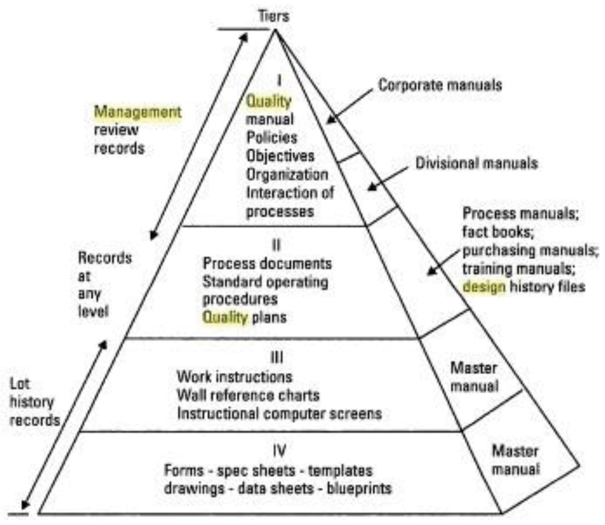


Figure 2.3: ISO 9001 documentation pyramid (Source: Schlickman, 2003, p. 37)

Similarly, it is acknowledged in the standard that based on the context, complexity and size documented procedures such as Control of documents (4.2.3), Control of records (4.2.4), Internal audit (8.2.2), Control of nonconforming product (8.3), Corrective action (8.5.2) and Preventive action (8.5.3) that could be enclosed in a manual, could vary (ISO, 2009). McMichael and Lombardi (2007) argue that in a Scrum environment, based on the context, project people may find it convenient to combine the procedure for several activities into a single documented procedure (for example, corrective action and preventive action) or they may choose to document a given activity by using more than one documented procedure (for example, internal audits). However, in order to demonstrate compliance with ISO 9001:2008, the organizations have to be able to provide objective evidence (not necessarily documented) that its QMS has been effectively implemented (Tricker, 2014).

2.4.1.3 Control of Documents and Records

In ISO 9001:2008, the requirements for the documentation review controls are defined under clause 4.2.3 and 4.2.4 respectively for the document and the records (ISO, 2009). These requirements enforce that a document shall be reviewed and corrected several times before making the document available (ISO, 2009). Hence a document goes through several rounds for corrections, accuracy, understandability and consistency of the content. Stalhane and Hanssen (2008) contend that having accurate documents is vital in Scrum teams as it helps to overcome most of the issues mentioned in section 2.2.2. Similarly, they argue that reviewing documents can be done in multiple rounds, hence this can be successfully incorporated with sprints in Scrum due to its iterative nature. Myhrberg (2009) argues that having details of the reviewer and comments after each iteration of review is beneficial for the next reviewer. Myhrberg (2009) also argues that in-order to maintain the consistency of the reviews, all reviewers are encouraged to follow the same reviewing format.

Requirements on maintaining the document version in use is another major requirement in sub clause 4.2.3 (ISO, 2009). Tony (2003) asserts that these requirements are beneficial to the Scrum teams when updating the documents to the latest versions, as they can successfully inform the interested parties about the outdated document versions. Schwaber and Beedle (2002) argue that having a mechanism of version control is useful for Scrum teams as information is always updated due to accommodation of requirement change. Brennan (2010) argues that maintaining a controlled set of documents in a Scrum team will lead the team to success by providing basic measurements for performance, shared knowledge among all team members, aiding staff training and acting as an initial start-up guide, serving as a baseline for assessment and improvement, adding a positive impact on quality, cost, productivity and customer satisfaction.

Ming et al. (2004) contend that maintaining a set of documents in a Scrum team enhances the communication as it provides a transparency in the activities and the decisions being made. Ming et al. (2004) as well as Ambler (2006) also highlighted that documentation helps to maximize stakeholder return on investment because a company can have a better understanding on cost and benefit aspect of projects. They also highlight that documentation should cover maintained records on what has been done and how to do these, information on process that are likely to change (Ambler, 2012; Ming et al., 2004). However, it will be useful for Scrum teams in health software development environments to know how the documentation requirements stipulated in ISO 9001:2008 could be used as a guideline to solve the issues arising due to lack of documentation normally associated with Scrum.

Mohammed (2006) observes that there are several requirements of ISO 9001:2008 through which an organization could add value to its QMS and demonstrate conformity by the preparation of other documents, even though the standard does not specifically require them. Some such documents referred to in the literature are: "Process flow charts and descriptions", "Organization charts", "Specifications", "Development instructions", "Documents containing internal communications", "Release schedules", and "Test plans and activities" (Mohammed, 2006; Osteen & Robbert, 1995). It has also been highlighted in the literature that such documents have to be controlled in accordance with the requirements of clause 4.2.3 and/or 4.2.4, as applicable (Cohen et al., 2003; ISO, 2009).

2.5 Documentation in Scrum projects

In traditional software development comprehensive documentation had a key-place in the development life cycle and was claimed that having a good set of documentation is a success factor. Documentation is defined as a success measure in requirements gathering, solution designing, software testing and quality, and software maintenance (Filipe , Ademar, Hugo , & Nuno, 2009; Tony, 2003). Therefore, it was believed that having less documentation leads towards lack of success in requirements gathering, designing, and maintenance and ultimately results in low software quality (Abrahamsson, 2002; Filipe et al., 2009; Rico, 2010).

Elke and Roland (2008) observe that with the introduction of Agile methods (such as Scrum), software professionals have resorted to the practice of having less documentation in their development work, misinterpreting the Agile manifesto statement “"working software over comprehensive documentation”. Abrahamsson (2002) observes that with this misinterpretation, software teams have created a new rule over documentation: “Agile methods (such as Scrum) have banned the documentation”.

Careful observation of Agile manifesto show that this manifesto does not ban documentation. The following statement sheds more light on this: “while there is value in the items on the right, we value the items on the left more” (Beck, Beedle, Bennekum, et al., 2001). This statement emphasizes that item on the right (“comprehensive documentation”) has its place in Agile software development (Rico, 2010). The creators of Agile manifesto, place an emphasis on high-quality interpersonal trust and communication, working software, customer interaction, and personal, technical, and organizational flexibility thus valuing the documentation, (Rico, 2010). Agile methodologies such as Scrum acknowledge documentation in a different way than traditional software development methodologies do. Followers of traditional software development values formally published software documents with hundreds of pages before beginning the software development life-cycle, where-as Agile methodologies (such as Scrum) values the production of working software first, and then the concise, user-driven multi-media documents which achieves interpersonal communication similar to traditional software methods (Elke & Roland, 2008; Rico, 2010). Figure 2.3 visualizes the documentation effort in traditional software development vs Agile development. It clearly symbolizes that Agile is still capable of achieving all the necessary documentation artifacts in the same time with lesser effort and lesser number of documents.

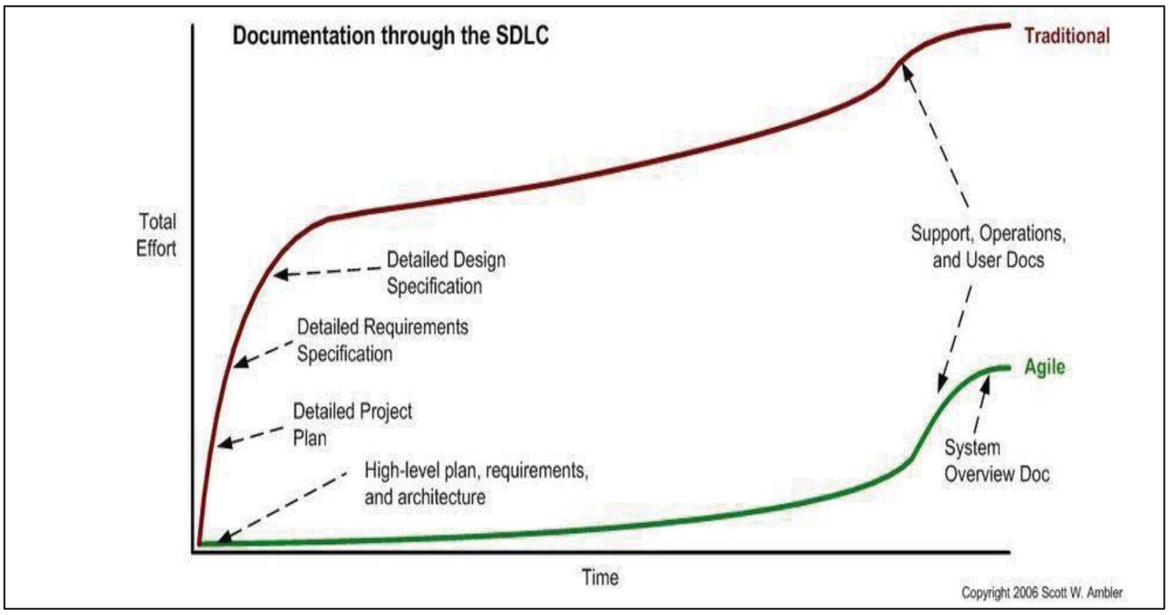


Figure 2.4: Documentation through the SDLC (Source: Ambler, 2006)

Since the nature of Scrum, welcoming requirements changes at any time of the development life cycle, information to be documented is changing very rapidly (Schwaber & Beedle, 2002). This leads to many Scrum teams having to wait until the very end of the development cycle to start documentation, reasoning that they want information to be stabilized before documenting to avoid re-work, cost and the risk of producing invalid information (Tony, 2003). By leaving the documentation task to the end, Scrum teams can save cost as they are not spending time in documenting information that changes; the risk is reduced because there is significantly less chance that the existing documentation will be out of date (Ambler, 2006). Ambler asserts that working on a document with less stabilized information produces the risk of having to rework the documentation once the information has changed. In other words, most development teams do not want to invest much time documenting projected ideas such as the requirements or design early in a project (Ambler, 2012). Instead, they wait until later in the lifecycle (by which time the information would be stabilized) to know what information is actually useful with the project. However, this results in increasing the effort of documentation as documentation tasks may be few iterations behind the software development effort (Ambler, 2012; Tony, 2003). Leaving the documentation tasks to the later stages can also cause issues such as: (a) teams forgetting some of the reasons behind the decisions that they made for a problem that they identified, (b) teams may not have the right people anymore to write the documentation because they've moved on to other projects (Ambler, 2012; Rüping, 2003; Tony, 2003).To avoid these issues, development teams are advised to start the documentation parallel to the development work, following right procedures to create concise documents and maintain consistency with the products (Ambler, 2012; Rico, 2010).

Ambler (2012) observes that documentation can start in a small scale and then iterate depending on the situation. With the start of the development of the project, documentation can be initiated by writing a little with information available, then review it gathering feedback and improve the documentation based on the feedback in an iterative fashion throughout the project (Ambler, 2012; Filipe et al., 2009). Consequently, documentation evolves with the product and no overheads—documentation development can be considered as part of the organizational learning process (Ambler, 2012; Filipe et al., 2009). Most importantly, documentation begins with less effort, in an incremental fashion. It is identified that the best documents are written iteratively, rather than being produce at once, because iteration enables to incorporate user needs (from the documents) via regular feedback (Mansour, 2013).

Mansour (2013) assert that documents that initiated at the early stages of a project can be used as a way of communication or a record of a communication to improve the understanding between project's stakeholders. Since recording initial knowledge of the project, decisions being made, and other important information from the beginning of the project are essential, documentation can help the Scrum teams in achieving the aforesaid objectives very easily (Ambler, 2012; Mansour, 2013). Ambler (2012) as well as Mansour (2013) assert that the goal of documents writing should be ensuring that development teams understand how the system works, so they can evolve their understanding on the system over time, rather than producing copious documents that they may or may not use during the project.

Mansour (2013) as well as Rüping (2003) argue that documentation can act as a medium of transferring knowledge in-terms of the requirements, technical design and testing. They also assert that documentation can act as a medium of recording information such as conversations with project stakeholders and can keep them actively involved with development. They highlight that although resolving stakeholder issues always work well face-face, rather than sharing documents, documents nonetheless help to keep track of what has been communicated, which in turn can be used to improve project outcomes.

Filipe et al. (2009) as well as Joaquim (2008) highlight that in producing documents iteratively, updating them is very important. Further, they contend that if the Scrum team maintains requirement documents, development notes, verification and validation reports up-to-date throughout development, then the chances are that these become valuable documents that be refereed to when needed (Filipe et al., 2009; Joaquim, 2008). Filipe et al. (2009) highlight that lesser the documents being up-to-date, lesser the value they would have, and therefore, it is important to update and review the documents throughout the development life-cycle.

Ambler (2012) shows that keeping documents simple (but not too simple), deciding the most important documents to have (i.e. include only the documents that cause least confusion), writing information in most appropriate places, and reviewing and making the documents public, are some of the key necessities in Scrum. Ambler argues that if the content of a document is not well defined, then regardless of how good the process is, documents would become less useful to the intended audience. Therefore, Ambler asserts that the structure and content of the document is important in Scrum projects because Scrum projects are very “time-bound”.

Ambler (2012) also highlights that a clear statement of the purpose of a document is a major requirement in documentation. Olivier and Pierre (2013) show the need to be restrictive with the amount of documents created during the development and the importance of identifying the purpose and the audience of the documents. Similarly, Filipe et al. (2009) as well as Tony (2003) assert that if a document fulfils a clear, important immediate goal of the overall project, then content of the document can be aimed to meet that goal. Rashina, James, and Stuart (2010) assert that each system has its own unique documentation needs and that “one size fits all” approach to documentation is not effective. They therefore argue that it might be worth understanding that the same "repeatable process" and the same set of documentation templates cannot be followed in every project.

Ambler (2012) asserts that identifying the audience of a document helps to realize the expectations of that document. Similarly, Filipe et al. (2009) and well as Rashina et al.

(2010) show that getting know the different roles in a Scrum team, stakeholders and other interested parties and engaging them into document preparation and review process produces the maximum outcome from the documents. Filipe et al. (2009) also show that identifying user actions, how they perform it, and how they want to work with the documentation are important as it helps to produce useful documents. Filipe et al. (2009) also go on to argue that keeping the audience informed at all times on the documentation (and the status) helps to produce the document collaboratively, which is in line with the Agile culture.

Sergio, Nicolas, and Thia (2005) show another important factor in documentation. That is, when determining the document content, irrespective of who writes the document, the document writer must make sure that the document has been written to a level that suits the audience who uses the documents. Similarly, Ambler (2012) and Filipe et al. (2009) assert that if the targeted audience is non-technical, then the document should be written in a manner that makes it easier for the readers to understand the content. They argue that knowing the right audience increases the value of the document as the document becomes a useful source of referring.

2.6 Chapter Summary and Research questions

The literature reviews highlighted that health software development is a field that contains more risk compared to other software development fields. Health software is developed to work closely with clinicians, practitioners and patients. Most of the time, data contained in the software are highly sensitive. Therefore, maintaining patient’s privacy and security is the most important aspect in health software. Unlike most other software applications, health software can affect people’s lives directly. Hence health software development requires achieving higher quality. Failure of a health-software can cause potential harm to the society. Therefore, development teams are responsible of finding the loose threads of security, privacy and other confidential items before they release the software. The literature review showed that documentation plays an important role in helping development teams in mitigating risk to achieve the quality objectives.

The literature reviews also highlighted that since health software is based on very complex and lengthy standards and specifications for the most part, it is very unlikely that all the members in the development team do read and understand the contents in the documents. Therefore, having a subset of important requirements documented for the work that is being done is important and so is issuing proper guidance to the development team. The literature reviews also showed that maintaining records and keeping track of the work done—in terms of software development, testing and other validation activities keep the team on the right track by developing the right product. This is because these measures minimize the chances of missing important parts of the software, which in turn leads towards producing a quality product. Another important aspect highlighted in the literature review was that regardless of how well the development team performs the development, if the customer doesn't understand how to use the product and how to troubleshoot, the quality performance of the product is significantly affected (in most cases this equates to product failure). Documentation was shown to be useful because having simple user manuals for the user (e.g. doctors, clinical staff, and patients for the most part) can mitigate the aforementioned risk of product failure.

In order to address the issue of limited documentation in Scrum, the internationally recognized ISO 9001:2008 QMS standard—this standard specifies the requirements of the QMS, including documentation requirements for quality assurance—was reviewed as this standard was chosen as the baseline to identify what documents Scrum teams should use. [4] With the said standard comes the question, how concise and precise the documents ought to be, in regards to Scrum. Yet another associated question is the timing of producing the documents and the ownership. More specifically, determining the right point in time to write each document and who should be the best person to take the ownership of the documents is vital. Finally, understanding the issues that arise with writing documents along with the development activities is important if documentation is to become effective in reducing quality risk in health software.

Although several researches have studied the compatibility between ISO 9001:2008 and Agile and/or implementing ISO 9001:2008 within Agile e.g. (Ambler, 2006, 2012;

[4] The ISO 9001:2015 standard (the replacement of ISO 9001:2008), which came into effect in late 2015, was in the draft stage at the time of researcher's data collection. It is important to note that this new standard emphasizes even more on risk management, while preserving basically the same elements that ISO 9001:2208 covered.

McMichael & Lombardi, 2007; Melis, Ambu, Pinna, & Mannaro, 2004; Nawrocki, Jasinski, Walter, & Wojciechowski, 2002; Stalhane & Hanssen, 2008; Wright, 2003) past research do not specifically cover health software development. Neither do they address identifying specific issues in relation to specific clauses of ISO 9001:2008 on documentation requirements. The research questions of the studies are given below.

RQ1: *Given that ISO 9001:2008 QMS standards is identified as the baseline, and given that Scrum happens to be the working method of a health software development company, what specific operational-level documentations are useful and important for such a company to comply with the requirements of the said QMS standard?*

RQ2: *How concise and when should the documents be prepared to maintain good communication among different parties in the software development lifecycle?*

RQ3: *What are the issues that cause documentation processes to be less effective, in a typical Scrum environment?*

It is important to note that this research looks only at operational-level documents (see Figure 2.3), as these are the documents that seemingly conflict with Scrum. High-level documents such as quality policy and quality objectives are strategic documents that do not conflict with operational activities such as Scrum.

CHAPTER 3
METHODOLOGY

3.1 Introduction

As outlined in the previous chapter (section 2.3.2) on issues that arise in healthcare software development, many Scrum-based healthcare software projects do not use an appropriate documentation system to align the project's quality goals with the software product being supplied to the clients.

The aim of this study is to identify the type of documents an Agile (Scrum) healthcare software development organization needs to have to seemingly meet the requirements of an internationally recognized QMS standard such as ISO 9001:2008.

I. A software development company that pursues Agile Methodologies sees accreditation against a quality management systems (QMS) standard as being necessary to gain competitive advantage in the market and that accreditation against the international QMS standard ISO 9001:2008 *is identified as the baseline by the company, and*

II. Scrum is the primary software development method used in the *software development company*

Based on the above setting, continuing from the previous chapter, the methodology adopted in answering the three research questions (see below) are given in this section.

RQ1: What specific operational-level documentations are considered useful and important for a healthcare software development company with Scrum practices to comply with the stated quality management system standard?

RQ2: How can the documents be prepared in a timely and concise manner to maintain good communication among different parties in the development lifecycle?

RQ3: What are the issues that cause documentation processes to be less effective, in a typical Scrum environment?

Chapter 3 is organized as follows. In the next section (section 3.2), different social research paradigms are discussed. This is necessary because the methodology used by a researcher is shaped by the particular paradigm to which they subscribe to (Lincoln, Lynham, & Guba, 2011). In section 3.3 the researcher justifies the specific research design, the case study methodology, in the light of her chosen research paradigm. In the next section (section 3.4) the researcher focuses on data collection methods used and provides a detailed description of the sample cases selected for this research. In the next section, the main focus is on outlining the iterative design used for data collection and the structure of the questions used. Along with the questionnaire design the said section describes how each data collection round was carried out and what techniques and tool were used. Finally, in section 3.5, a chapter summary is presented.

3.2 Research Paradigms Used in Social Research

The researcher's study relates to organizations and people, hence to social research. Therefore, it is important to investigate different paradigms available to conduct inquires of social nature. Two concerns that relate to a social research paradigm are: what constitute appropriate knowledge (epistemology) and how one views the social world (ontology) (Babbie, 2015; Bryman, 2012; Punch, 2013). There are two dominant social research paradigms: positivism and interpretivism. A third paradigm pragmatism is also gaining traction in social research. Since the ontology, epistemology, and therefore the methodology used by a particular researcher depend on what paradigm they use, the three paradigms are discussed next.

3.2.1 Positivism

The positivistic ontology holds that the researcher and their subjects are independent and external to what is being observed (that is, value free), in the sense, people are not part of fashioning knowledge on social phenomena (Babbie, 2015; Bryman, 2012; May, 2011). The positivistic epistemology holds that knowledge claims can only be made by following the scientific method of observing a phenomenon, hypothesizing a cause-effect explanation/s, collecting data (to test the hypothesis/s), testing the hypothesis/s, and confirming, refuting or refining the hypothesis/s for the next round of testing. Therefore, positivistic research methods generally follow *quantitative methodologies* (e.g. experiments, quantitative data collection through survey instruments), in which statistical principals play a major role.

Post-positivism is a variant (a less stringent version) of positivism which does not rely on strict value free judgment in the positivistic ontology. Post-positivists assume that observations are fallible (there is an element of error and uncertainty in any observation) and therefore knowledge claims require multiple sources of evidence (e.g. methods, data streams). However, post-positivists also advance knowledge by theorizing and testing hypotheses using quantitative data.

3.2.2 Interpretivism

The interpretive ontology holds that knowledge is socially constructed—that is, knowledge is fashioned by value judgements of the people (Bryman, 2012; Ritchie, Lewis, Nicholls, & Ormston, 2013; Sarantakos, 2012). The interpretive epistemology holds that knowledge is subjective (as opposed to being objective as in positivism) and therefore, the role of the researcher is to "grasp the subjective meaning of social action" by adopting a research strategy that distinguishes objects (as in natural science) from people (Bryman, 2012, p.30). This research strategy requires tapping into the "common sense thinking" of the people, who are an integral part of "social reality", because the action of people have meaning to them and their social institution (Bryman, 2012, p.30). Consequently, researchers who follow an interpretivist paradigm rely predominantly on non-quantitative research methods (qualitative research methods) to understand social phenomena.

3.2.3 Pragmatism

A third and an emerging paradigm in social research is the pragmatic paradigm (Denzin, 2010; Morgan, 2014). As the term implies, pragmatism is primarily concerned with practical considerations rather than philosophical considerations: epistemology and ontology. Thus pragmatism primarily deals at methodological level rather than the epistemological level. As such, pragmatism is invariably associated with what is known as mixed methods. Mixed methods are associated with the use of a mix of quantitative and qualitative methods. Which method (quantitative or qualitative) becomes dominant depends on the nature of one's research objectives (Creswell, 2014; Morgan, 2014).

3.3 Justifying the Case Study Design and the Corresponding Research Paradigm

Case study design is defined as “an empirical inquiry that investigates a contemporary phenomenon within its real-life context, especially when the boundaries between phenomenon and context are not clearly evident” (Yin, 2013, p. 16). Therefore, a case study research can be used, when phenomenon is broad and complex, when full depth investigation is needed and most importantly, when the phenomenon cannot be studied outside the context (Dubé & Paré, 2003; Feagin, Orum, & Sjoberg, 1991; Voss, Tsikriktsis, & Frohlich, 2002; Yin, 2013). Another attractive feature of a case study design is that it allows flexibility in terms of the methods and the researcher allowing a researcher to conduct an exploratory type of a research (Cavaye, 1996; Eisenhardt, 1989; Merriam, 1998).

Case studies are often used to describe phenomena, to develop theory, and to test theory (Yin, 2013). In case study approach, both qualitative and quantitative approaches can be used (Dubé & Paré, 2003). Qualitative data collection methods which concerns words and meanings are such as interviews, documentation, questionnaires and observations can be used (Dubé & Paré, 2003; Wohlin, Höst, & Henningsson, 2003). Quantitative data collection methods which concern numbers and measurement are such as questionnaires and time series (Dubé & Paré, 2003; Wohlin et al., 2003).

Based on the authorities in case study research (e.g. Dubé & Paré, 2003; Myers & Avison, 1997; Yin, 2013) and IT-specific case study researchers (e.g. Dubé & Paré, 2003; Myers & Avison, 1997) it seems that case study research is well suited for IT industry. Dube and Paré (2003) identify five main reasons for case studies being highly suitable for IT research:

1. The discipline in IT organizations is information systems, where interest shifts to organizational rather than technical issues (Benbasat, Goldstein, & Mead, 1987; Darke, Shanks, & Broadbent, 1998).

2. Having access to real situations allows an IT researcher to experience and report real life IT experience with the rapid changes occurring in the IT world as well as in the organization (Dubé & Paré, 2003).

3. Since being holistic is the main characteristic in case study research, case studies can be best utilized to understand the complex and wide ranging interactions among organizations, technologies, and people (Dubé & Paré, 2003). Having the capability of accommodating both qualitative and quantitative research methods, case study approach can bring richness and flexibility to the research, in-terms of analyzing complex environments (Dubé & Paré, 2003).

4. The ability of performing in-depth analysis in case-study approach opens up new ideas, opportunities and new phases of reasoning for the issues facing the IT industry and its workforce (Dubé & Paré, 2003).

5. Case study approach suits exploration and hypothesis generation. Also, case studies can be used to provide explanations and testing hypothesis (Benbasat & Weber, 1996; Dubé & Paré, 2003).

Orlikowski and Baroudi (2008) arrest that the research methodology can be selected depending on the nature of the reserch questions. If the research question is a "what" type question, then it is more likely that the question can be addressed by surveys and data analysis, which converges towards a quantitative type study (Lee & Xia, 2010). However, even if a "what" type question is served as an entry point to other questions, a “what” question can be addressed via an exploratory study such as a case study (Kaplan & Duchon, 1988). When the research question takes the form "why", "how" and "when", the question can more elegantly answered by a qualitative method and the case-study approach becomes a good candidate for the study (Lee & Xia, 2010; Yin, 2013).

While there is no silver bullet to justify that case-study best suits a particular piece of research, Yin (2013) asserts that if the research questions are meant to explain some present circumstance (e.g. in the form of “what”, “why” and “how”), the “case-study research” becomes more relevant relative to rival methods.

In the researcher's study, the research questions are more focused on defining the present situation in a Scrum project. Therefore, the case study approach fits the bill (studying a real-life scenario). Moreover, because there is lack of control over certain variables such as the 'organizational procedures and standards', 'market regulations', as Yin (2013) advocated, the case study design becomes a very suitable approach for the candidate's study.

Since this study focuses on identifying documentation requirements in a Scrum project (being viewed as necessary by a panel of experts) to accredit against the ISO 9001:2008 QMS standard, there are two major domains being touched by the study. One is the Scrum project concept, which is in the field of (healthcare) *software developmen*t. The other is ISO 9001:2008 certification (more precisely, the documentation requirements as stipulated in ISO 9001:2008 QMS standard), which is in the field of *quality systems*. Since the research questions cannot be answered without actually engaging with key personnel, the case study method was chosen as the appropriate research design for the researcher's study; the researcher observes that the case study method has been used to answer research questions of similar nature in the field of information technology (IT) (e.g. see Darke et al., 1998; Dubé & Paré, 2003). IT is a discipline where rapid changes take place within a short period of time and the case study method enables the researcher to participate in the real-life context to experience the dynamic changes taking place in a complex work environment (James, 1989; Yin, 2013).

Finally, the case study was conducted within a *pragmatic research paradigm* using a mix of quantitative and qualitative data. A fully quantitative research involving large sample survey data (a cross-sectional design within a positivistic/post-positivistic paradigm) is not suitable because there is no cause-effect or correlational hypothesis to be tested. Also, large sample data collection (quantitative, qualitative or a mix of both) is not feasible for this research due to limited time and other resources (e.g. funding).

3.4 Data Collection and Research Design Considerations

With the given tight time frame available, a relatively small number of participants (*n* = 20), being Scrum development team members representing different specialties were recruited for the study; the number of participants involved varied from one stage of data collection to another in the multi-stage data collection process (details in section 3.5). The 20 participants recruited represent 4 organizations specializing in Agile (Scrum based) software development within the same geographical location (Auckland). Also, in New Zealand IT industry, it is hard to find development teams who are highly matured in Scrum processes and at the same time enthusiastic on achieving ISO 9001:2008 certification. As a byproduct of data collection, the researcher learnt that most IT companies are scared to put new processes in place because of the additional effort (e.g. training and creating knowledge to effect appropriate documentation systems that fit an Agile environment) needed to support those processes. Although the participants recruited for the study were found to be very enthusiastic about the terms of reference assigned to them by the researcher, not all of them were highly experienced in Scrum-based processes. This is a limitation of the study.

3.4.1 Unit of Analysis

Unit of analysis is an important aspect in case study research (Dubé & Paré, 2003). The unit of analysis refers to the subject (who or what) being studied. In this research, the unit of analysis is the "Scrum project team who develops healthcare software". Defining the unit of analysis helps a researcher to define the boundaries of their case study (Yin, 1999). The boundaries guide the researcher in choosing the right sample for data collection. The boundaries for this study are as follows:

- Development team who are using Scrum practices
- Development teams consisting of 1-6 number of members (see section 2.2.3 for composition of Scrum teams)
- Development teams specialized in healthcare software solutions.

3.4.2 Rigor - Reliability and Validity

Traditional quality criteria (in case study method or otherwise) of rigor are the validity and reliability of the data and the conclusions achieved by the research (Bryman, 2012; Yin, 2013). Establishing tests to check reliability and validity of quantitative data are well documented but in the case of qualitative data, examining reliability and validity of data can be more challenging (Whittemore, Chase, & Mandle, 2011). Weiner (2007) defines reliability as "the degree to which a measurement technique can be depended upon to secure consistent results upon repeated applications"; Weiner defines validity as the "degree to which any measurement approach or instrument succeeds in describing or quantifying what it is designed to measure". Yin (2013) gives four common tests on validity: construct validity, internal validity, external validity, and reliability. The final item reliability is a necessary but not a sufficient condition for validity (Bryman, 2012; Weiner, 2007).

3.4.2.1 Construct Validity

Construct validity identifies if the research study has used correct operational measures to measure what should actually have been measured in the study (Yin, 2013). In general, social research involves measurement of concepts (constructs) that require multiple measures to capture the richness of the concept (Hoyle, Harris, & Judd, 2002). Within a case study framework, construct validity can mean "multiple source of evidences" that provide a "chain of evidence" and/or when the key informants have reviewed a draft case study write-up for factual accuracy of information provided by the informants (Yin, 2013).

Constructs in this study are the specific concepts that the researcher attempts to capture using multiple measures. In social research constructs are the basic elements that a researcher uses to explain a social phenomenon (Bryman, 2012; Hoyle et al., 2002). Examples from the IT discipline could be "Agile maturity", "intensity of IT use" and similar concepts that are not directly observable. Thus a social researcher has to identify appropriate operational measures that represent the relevant constructs (concepts) of their study. For this study an example of operational measures—for the construct "conciseness of documents" related to RQ2—include relevant records which can be presented in the documents in a Scrum project to gain ISO 9001:2008 certification. Yin (2013) asserts that constructs and their operational measures should be defined explicitly prior to the commencement of data collection, along with a relevant explanation on what is expected from the audience. This protocol was followed by the researcher.

Yin (2013) asserted that the primary strategy a case study researcher should adopt to addresses construct validity is to use "multiple sources of evidence", which is also known as triangulation in social research (Gibbert, Ruigrok, & Wicki, 2008). Multiple sources of evidence could be data collected at different times, locations, and from different people; use of more than one investigator; and the use of multiple methods (Lindgreen, Hingley, Stavros, & Westberg, 2009; Mathison, 1988).

In this study data were collected using focus groups, mainly through a semi-structured survey. This included multiple rounds of interviews targeting best suited participants for each round. Each round was carried out leaving a 4-6-week gap between each round. Thus data collection was accomplished using different people at different times using different data collection methods—surveys, interviews, focus group briefings, brainstorming—filling the *multiple sources of evidence* (on the same phenomenon) requirement for construct validity. The reader should note that use of different locations and investigators was not possible due to resource constraints and academic reasons respectively.

The second strategy that Yin (2013) prescribed to strengthen construct validity in case-study research, "establishing a chain of evidence", was implemented by formulating logically related (sequential) research questions and marshalling evidence in a sequential fashion (the 3-step process is described later).

The third strategy that Yin (2013) prescribed to strengthen construct validity in case-study research, to review the case study report by key informants (respondents), was implemented as follows. In this study data were gathered via questionnaires using multiple rounds and prior to commencement of the next round (and also at the end of the final round) the researcher briefed the findings of the previous round to each participant via the questionnaire. For example, the introduction and background mentioned in the questionnaire used in the second round of data collection was the key findings of the previous round of data collection.

3.4.2.2 Internal Validity

The second type of validity test is internal validity. Internal validity is a concept applied only for casual and explanatory studies but not for descriptive or exploratory studies

(Yin, 2013). Internal validity refers to the ability of the study's research design to exclude any rival explanations (confounding effects) of the phenomenon being studied (Brewer, Reis, & Judd, 2000; Bryman, 2012). Since this study is an exploratory study, the necessity to implement specific strategies to enhance internal validity did not arise.

3.4.2.3 External Validly

The third type of validity test is external validity. External validity refers to the domain within which the findings of a research can be generalized (Yin, 2013). In social research, there are two types of generalizations: statistical generalization and analytical generalization (Yin, 2013). Statistical generalization refers to making a statistical inference on the population based on the sample. The cases (units of analysis) in a case study are not sampling units drawn from a population, and hence statistical generalization does not apply in case study research; what is applicable in case study research is analytical generalizability, where a researcher provides a theoretical basis for the case study findings (Yin, 2013). This theoretical basis is in turn used to explain events similar to what was observed in the case study. In this research, the researcher has argued that the results produced can be reproduced in a context similar to that of the case study, namely a Scrum-based software development environment in which ISO 9001:2008 accreditation has become necessary.

3.4.2.4 Reliability

An important facet of validity (more technically construct validity) is reliability. In a strict sense, reliability is a positivistic notion related to the measures used to operationalize a social concept (construct). In positivism reliability refers to the extent to which a measure used to operationalize construct is "free from random error" (Voss et al., 2002). Thus reliability refers to extent to which repeated measurements (by the same person or a different person) produce similar results (Voss et al., 2002; Yin, 2013). In a case study context, reliability can be established by a clear case study protocol and maintaining records of the evidences such as a case study database (Yin, 2013).

Protocol used in this research is the documented procedures, which can be repeated throughout the research; also these can be used by another researcher. In this research data collection is done by using sets of questionnaires, which enable all the participants to provide their feedback using a unified data collection framework. Semi-structured focus group interviews were carried out iteratively in this research by following same pattern and guidelines throughout the research.

Also, records of all the responses received were maintained in a database file, among other things, for verification purposes. The database contained all the relevant journals, articles, questionnaires, recording of the results, and notes taken via the interviews. Moreover, the researcher has provided a clear description of the data sources and how the findings follow from the data, which is an important aspect of reliability and validity of the findings (Benbasat et al., 1987).

3.4.3 Selecting Appropriate Data Collection Techniques

Given that case study method has been selected as the appropriate research design for the study and pragmatic paradigm has been chosen as the research paradigm, the next question is to select a suitable data collection method that fits a case study research design, within a pragmatic paradigm. In keeping with the pragmatic paradigm, a mixed method approach involving a mix of quantitative and qualitative data was selected as a suitable approach to answer the research questions. Semi structured interview questionnaires were chosen as the most suitable data collection technique to collect the necessary mix of quantitative and qualitative data.

In research there are two sources from which data can be collected: primary sources and secondary sources (Kumar, 2005). Primary sources are data collection sources the researcher created at the time of their field research (in the case of this research, the semi-structured questionnaire) while the secondary sources are the sources that are external to the researcher’s field research which the researcher can recourse to (e.g. administrative records of an organization) either as a standalone data source or as a data source that supplements the primary source (Kumar, 2005; Venkatraman & Ramanujam, 1986). The researcher used both primary and secondary data sources in their study (Table 3.1), which strengthens the case study findings (increased construct validity) on account of utilizing multiple data sources (Yin, 2013).

Specifically, the following data sources and data collection techniques (Table 3.1) were identified to answer the research questions.

Table 3.1: Data Collection Techniques and Data Sources Used to Answer Each Research Question

Research Question	**Data Collection Technique**	**Justification**
RQ1: What specific operational-level documentations are considered useful and important for a healthcare software development company practicing Scrum to comply with the ISO 9001:2008 QMS standard?	Semi-Structured Questionnaire **[P]**	Answering RQ1 requires input from experts on both Scrum and ISO 9001:2008 QMS standard. Data need to be collected from experts iteratively (multiple stages) until the question is answered through consensus.
RQ2: How can the documents be prepared in a timely and concise manner to maintain good communication among different parties in the development lifecycle?	Semi-Structured Questionnaire **[P]**	Answering RQ2 requires input from experts on Scrum. Data need to be collected from experts iteratively (multiple stages) until the question is answered through consensus.
RQ3: What are the issues that cause documentation processes to be less effective, in a typical Scrum environment?	Semi-Structured Questionnaire **[P]** Reports **[S]** Documents **[S]**	Answering RQ3 requires input from experts on Scrum. Data need to be collected from experts iteratively (multiple stages) until the question is answered through consensus. Inclusion of information collected from the analysis of data reports provides richer evidence to answer RQ3.
Note: **[P]** – Primary Source; [**S]** – Secondary Source		

From Table 3.1 it becomes evident to the reader that in order to answer the research questions—where one research question follows from the other—a sequential data collection approach should follow. The nexus between RQ1 and RQ2 is shown below.

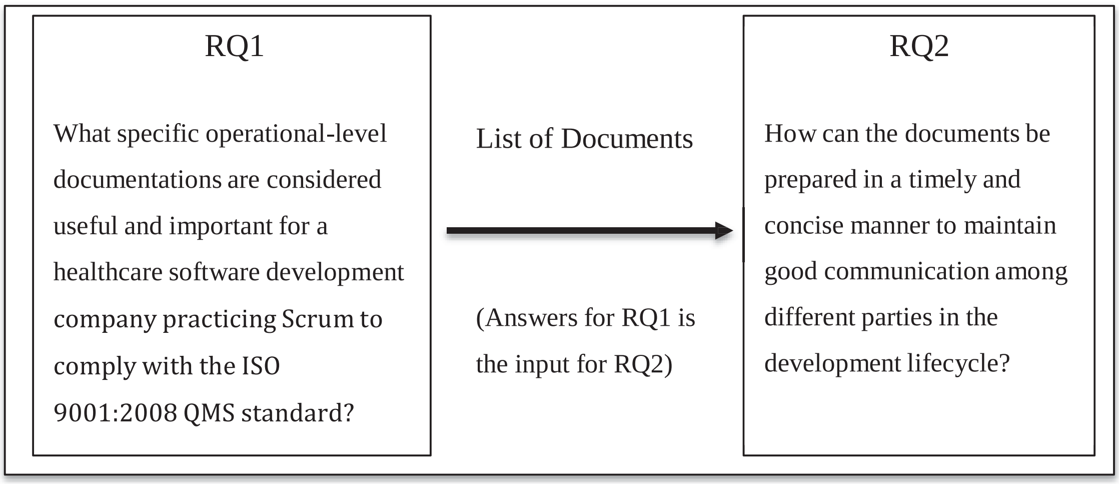

RQ1
What specific operational-level documentations are considered useful and important for a healthcare software development company practicing Scrum to comply with the ISO 9001:2008 QMS standard?
List of Documents
(Answers for RQ1 is the input for RQ2)
RQ2
How can the documents be prepared in a timely and concise manner to maintain good communication among different parties in the development lifecycle?

Figure 3.1: The interrelationship between RQ1 and RQ2

Answering RQ3 requires adequate knowledge and experience on the part of Scrum development team; this is because then only the difficulties/issues that arise when documents are being produced (see RQ3) can be easily identified. Use secondary data sources to supplement primary data leave the researcher much space and freedom to conduct more investigation in connection with RQ3. Although answers to RQ3 can be found without recourse to findings on RQ1 and RQ2 (i.e. treating RQ3 as an independent research question), the findings on RQ1 and RQ2 can provide additional information (and hence a more comprehensive analysis) to answer RQ3.

Considering the dependency of RQ1 and RQ2, data collection rounds were prepared so as to have multiple rounds of data collection (each round collecting data to answer one research question only, starting from RQ1). More specifically, each round generates information based on the feedback (aggregate responses/information) received from the previous round to answer a specific research question corresponding that round (e.g. RQ2 in round 2). The researcher took steps to ensure (to the extent possible, given the time other resources available to the researcher) that questionnaires were presented (in each round) to an audience that is best equipped to answer the questionnaires; this is to enhance the validity of the findings. Also, to make the data collection process efficient and effective, the questions in each round were presented to a predetermined panel (details in section 3.5).

Consequently, the following sampling process was adopted in recruiting subjects (case study participates) for different data collection rounds (more details are given in the next section, section 3.5).

3.4.4 Selection of Samples

A population can be defined as a collection of people, items or events which a researcher would like to research on. It is not always possible to study every single element belonging on the population due time and other resource limitations (Kaiser & Dickman, 1962). Therefore, a subset of the population is usually selected in research and this is called a 'sample' (Fraenkel, Wallen, & Hyun, 1993). If the sample is random and large enough to perform the research, then the data collected can be used to make statistical inferences about the population (Kotrlik & Higgins, 2001). In case study research, it is not necessary to select a random sample although a sample representative of the population (in the researcher’s study the population is Scrum teams interested in QMS accreditation via ISO 9001:2008) enhances the external validity of the results (Yin, 2013).

In this research study sample is selected using a “convenience sampling strategy” (Dubé & Paré, 2003). Participants were identified based on the main objective of the research, considering the convenience of recruiting them to the study. Therefore, team members of software development teams who are *approachable* and who have Scrum development practices in place were recruited for the study. Based on pre-defined criteria (outlined below), 20 panel members were recruited for the study. The criteria used to select participants (panel members) are as follows.

Diversity, in terms of Roles Being Played, Within a Scrum Project.

- Members of internal development teams who are engaged in development roles such as Product owners, Scrum Master, Technical writers, Developers, Testers and etc. should be included (saturation of one specific role to be avoided).
- At least few respondents in the panel should be those who are engaged in roles that involve significant customer interaction (e.g. Support engineers, Business Analysts).
- At least few respondents should be engaged in roles that have management level responsibilities (e.g. Project Managers, Team Leaders).

- Sufficient number of respondents who are familiar with quality assurance systems and ISO 9001:2008 QMS standard (e.g. Quality Assurance Engineers) should be included in the panel.

Variability of Experience

While more experienced staff (respondents) are preferred over less experienced staff as panel members, the researcher decided on the following minimum level of experience as inclusion criteria in the panel.

- At least a third (30% approx.) the respondents should have 5 years of work experience on Scrum.
- At least about half of the respondents (50%) should have 3-5 years of work experience on Scrum.
- The remaining panel members can 2 years of work experience or less.

Domain Knowledge

- At least 95% of respondents should come from the healthcare IT industry.

Based on the above criteria, Figure 3.2 depicts how the samples were selected for each round of data collection (questionnaires and interviews).

As per ‘Figure 3.2- Depicting sample selection’ out of the massive software professional population, sample is identified based on the 3 factors listed above. More specific details of the characteristic of the sample are outlined in section 3.4.4.1. Out of that sample for each round of data collection, suitable number of participants is chosen.

Section 3.5 provides details on the specific data collection method and the criteria used to choose participants in each round of data collection.

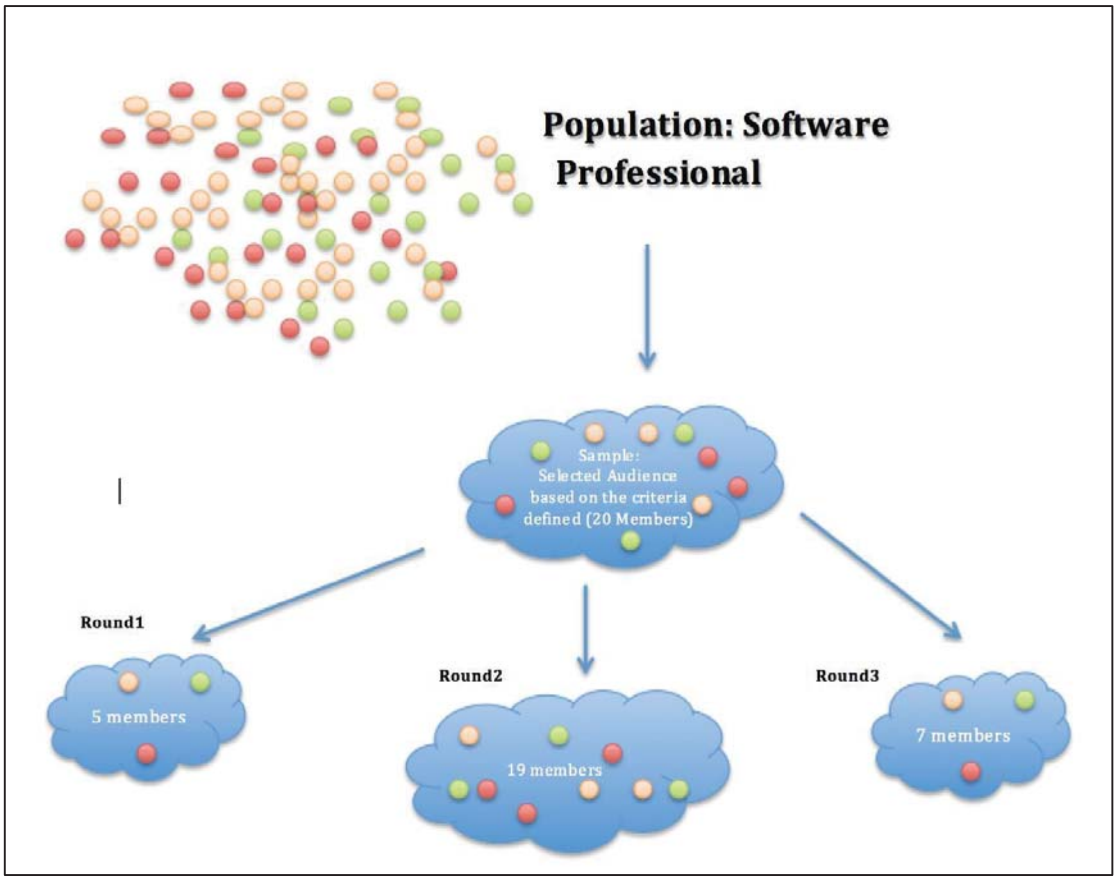

Population: Software Professional
Sample: Selected Audience based on the criteria defined (20 Members)
Round1
5 members
Round2
19 members
Round3
7 members

Figure 3.2: Depicting sample selection

3.4.4.1 Characteristics of the sample selected

Based on the selection criteria of panel members described above, the composition of the panel members selected for the study is shown in 'Table 3.2- The Composition of the 20 Member Panel'.

Table 3.2: The Composition of the 20 Member Panel

Role in the Development Team	Expertise	Experience (Years)	No. of Respondents
Development Team Leaders	Team management, Sales, Scrum practices, Health-IT context knowledge	1-2	2
Project Managers	Project Management, Scrum practices, Health-IT context knowledge	1-2	3
Business Analysts	Business Analyst skills,	2-3	2

Role in the Development Team	**Expertise**	**Experience (Years)**	**No. of Respondents**
	Scrum Practices, knowledge on health-IT context		
Product Owners	Stakeholder management, Negotiation, Decision making, Development standards, Scrum practices, Health-IT context knowledge	1-4	2
Solution Architect / Technical Lead	Development guidelines, Technology, Team Management, Scrum practices	5-6	1
Test Engineer	Quality Assurance, Scrum practices, Health-IT context knowledge	2-3	3
Quality Management System team member	Quality Management systems, ISO 9001:2008, Scrum practices, Health-IT regulations	3-4	3
Software Engineer	Development guidelines, Technology, Scrum practices	2-3	4
User Interface developers	User interactions, Scrum practices	1-3	1
Technical writes	Technical writing, Development standards, Scrum practices	2-5	2
Customer Support Engineer	Query management, Agile skills, Health-IT context knowledge	4-5	2
Implementation consultants	Stakeholder management, Product knowledge, Health-IT context knowledge	3-5	2

3.5 Review of the Literature on Panel-Based Data Collection Methods and Selecting an Appropriate Method to Answer Structured Questions

In selecting an appropriate data collection method, the researcher defined five requirements (eligibility criteria) the method should meet. These are shown in Table

Table 3.3: Eligibility Criteria for the Primary Data Collection Method

ID	**Eligibility Criteria**
EC1	It should address a panel of audience
EC2	It should be able to accommodate multiple rounds of questionnaires
EC3	It should be iterative, in the sense, it should have the ability to construct questions (questionnaire items) in a particular round, based on the feedback received in the previous round
EC4	It should be able to accommodate different type of questions in each round, whilst focusing on the main objective of the study
EC5	It should invite maximum diversity of the answers, so that the responses can be considered rich in content (also, not biased)

Based on the eligibility criteria mentioned above (Table 3.3), a well-known panel based data collection technique known as "Delphi Methods" is reviewed in the following subsection in order to ascertain whether a technique that comes under an umbrella term "Delphi methods" or similar technique can be used for this research study.

3.5.1 Delphi Methods

The Delphi methods refer to a generic decision tool that is widely used for accommodating ideas/inputs opined by experts within a certain topic (or a subject area) through an iterative process that converges on a solution—by considering real-world scenarios—for the decision problem (Hsu & Sandford, 2007). This technique was developed by Dalkey, Brown, and Cochran (1969) at the Rand Corporation USA in the 1950s to forecast future technological capabilities that might be used by the US military in warfare. The technique has been generalized to solve many different decision problems that are hard to solve using conventional methods (Hsu & Sandford, 2007).

The Delphi technique is well suited for consensus-building (with the aid of a facilitator) by using a series of questionnaires to collect data in an iterative fashion, from a panel of selected subjects (Hsu & Sandford, 2007). The iterations are designed by the facilitator based on the feedback received from the respondents (panel members) in the previous round of data collection (Hsu & Sandford, 2007). In a Delphi study that involves questionnaires[5] every participant would respond to the terms of reference (TOR) of a questionnaire and the role of the facilitator is to collect the responses, summarize the results and re-specify the requirement's (TOR) for the next data collection round and collect new data with a view to converge to a solution (in this research, the researcher played the role of the facilitator) (Hsu & Sandford, 2007). Theoretically, the Delphi process can be continuously iterated until consensus is reached (Skulmoski, Hartman, & Krahn, 2007). Practically, a typical Delphi process would iterate for 2 to 3 rounds in reaching a consensus solution (answer) to the decision problem. At the end of the Delphi iterations, every member in the panel will agree upon one or common set of answers (consensus) which are considered to be the "correct" answers (Skulmoski et al., 2007). When the respondent(s) does not agree upon one answer, it is the duty of the facilitator to help the dissenting member to arrive at a common standing (Skulmoski et al., 2007). Hsu and Sandford (2007) describe a typical 3-round Delphi process as follows.

Round 1: The opening round will typically have an open-ended questionnaire, which serves as the basis for soliciting specific information about a certain content area from the Delphi subjects. Based on the responses received the facilitator would develop a more structured questionnaire for data collection in the next round.

Round 2: In the second round, each participant receives a second questionnaire; in this round the participants are asked to review the items in the questionnaire—the items being prepared based on the findings in the first round.

Round 3: In the third round, each Delphi panelist receives a questionnaire that includes the items and ratings summarized in the second round; the participants are asked to elaborate their judgments and further validate their responses. At the end of the round 3 typically all participants would agree upon (through necessary facilitation processes) a common solution to the decision problem, thus terminating the Delphi process.

[5] Questionnaires need not the tool that a facilitator uses always to collect responses.

Based on the eligibility criteria defined in Table 3. 1, the researcher assesses the suitability of Delphi methods for this study as follows (Table 3.4).

Table 3.4: The Suitability of the Delphi Approach for the Researcher's Study

Eligibility ID*	**The Specific Attribute of the Delphi Approach**	**Compatibility with the Research Setting**
EC1	Delphi technique is specifically designed as a group communication technique aiming to achieve specific objectives (Hsu & Sandford, 2007). As Hsu and Stanford (2007, p. 1) assert "the Delphi technique is well suited as a method for consensus-building by using a series of questionnaires delivered using multiple iterations to collect data from a panel of selected subjects". A specific attribute of the Delphi approach is the preservation of anonymity of the respondents, which is not possible in an open type group communication process such as a project meeting (Dalkey, 1972; Dalkey et al., 1969; Hsu & Sandford, 2007)	Preserving the anonymity of the participants' identity limit negative communication that could otherwise occur in a face-to-face panel discussion setting, which would have been actually possible in this research, because most of the respondents selected for the research work in the same case-study healthcare software development organization (and the others are also known to the participants of the case study organization); the Delphi approach thus avoids the interplay between different personalities (typical in an organization), which usually promote unpleasant experience and restrain freedom of expression. Thus Delphi seems to preserve the best of both words—group communication and avoidance of conflicts—which suits the researcher's study.
EC2	The Delphi method is an iterative process which is used to collect results of experts using a series of	Ability to accommodate multiple rounds of questionnaires and adopting an iterative problem

Eligibility ID*	The Specific Attribute of the Delphi Approach	Compatibility with the Research Setting
	questionnaires interspersed with feedback received from its participants (Skulmoski et al., 2007). However, at the end of the Delphi method, everyone in the panel will agree upon one or common set of answers which are considered to be the "correct" answers (Skulmoski et al., 2007)	solving process suit this research. However, bringing all the respondents to a unified answer (in spite of the facilitator's best efforts) was not practicable for this research due to three reasons: time constraint (no more than 3 rounds of data collection and briefing of results was possible), diversity of the participants in terms of experience and their roles in Scrum teams, and the formal authority bestowed upon the facilitator— who is also the researcher (an explanation follows).
EC3	A Delphi method iteratively collects results based on expert opinions, using a series of questionnaires interspersed with feedback received from the participants (Skulmoski et al., 2007). However, at the end of the Delphi method, everyone in the panel will agree upon one or common set of answers which are considered to be the "correct" answers to the problem statements (Skulmoski et al., 2007)	When multiple data collection rounds exist, each question round should consider the feedback received from the previous round. This can be followed by the researcher without hindrance; the researcher also leant that iteration helps to increase the validity of the results as the quality of the questionnaire also improves as new information is received from the previous round.
EC4	Delphi technique is iterative and sequential; it can accommodate	Multiple data collection rounds pervasive in the Delphi approach

Eligibility ID*	**The Specific Attribute of the Delphi Approach**	**Compatibility with the Research Setting**
	questions based on the objective of the decision problem and the data collection round (Hsu & Sandford, 2007).	permits the researcher to build up the knowledge sequentially, starting from a board unstructured problem statements through to more focused ones in the subsequent rounds. In this regard, the Delphi approach is compatible with the researcher's study.
EC5	In Delphi method, every member would agree upon a common (consensus) answer (or a set of answers). When a respondent(s) does not agree with the answer (opinion) given by others, it is the duty of the facilitator to facilitate the dissenting member/s to fall in line with the rest to find a common ground.	As mentioned earlier, due to practical reasons it was not possible to reach consensus. This was foreseen by the researcher and one reason for selecting a relatively large sample (by Delphi standards) is to account for the variability of the responses that cannot be controlled by the researcher through deliberation.
* See Table 3.3		

As evidenced from the contents in Table 3.4, the Delphi approach cannot be applied in its purest form for the researcher's study. The main issue is in-built constraints imposed upon the researcher to moderate (facilitate) the responses (opinion) given by the panel members. Although the researcher is a Business Analyst in the case study organization the researcher is relatively new to New Zealand (the researcher hails from Sri Lanka) and this limits the researcher's formal authority to moderate proceedings (usually facilitators and moderators are people in senior roles). While not being able to moderate or influence the respondents enabled the researcher to obtain the *Massey University human ethics approval* under a *Low Risk Notice* (Appendix–E), the researcher admits that their role in the case study organization was a limiting factor of this study.

The next subsection provides details of the specific problem solving method used in the study; the researcher names this method the “Delphi-like panel-based problem solving method”.

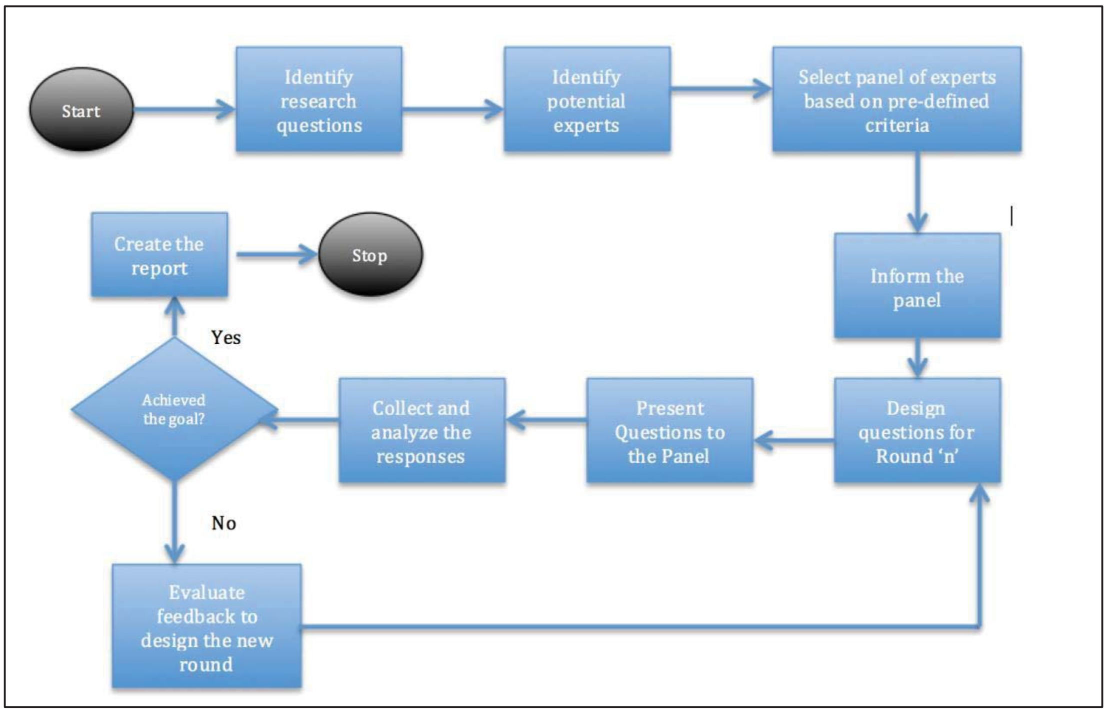
3.5.2 The Specific Method Used in this Study: The Delphi-like Panel Based Problem Solving Method

The primary objective of this research is to identify the minimal documents required in a Scrum project to comply with the documentation requirements for ISO 9001:2008 certification. As mentioned in chapter 2 (section 2.4), the ISO 9001:2008 standard stipulates the so called mandatory documentation required (e.g. the quality manual, quality policy and quality objectives); most of these documents are the so-called high level (strategic level) documents; the standard remains nonprescriptive when it comes to operational-level documentation requirements; operational documentation is necessary for a fair minded registrar (the auditor in a third party organization that audits the organization for compliance with ISO 9001:2008 QMS standard) to assess the quality system (for certification worthiness) of the candidate organization (Poksinska, Jörn Dahlgaard, & Eklund, 2006; Wealleans, 2005). Many auditors (ISO 9001:2008 registrars as well as the internal auditors of the case study organization are no different) are trained to relay on documented evidence. For this reason, it is desirable to have more documents as opposed to less as an insurance against uncertainty that would arise for lack of information (Wealleans, 2005).Thus, this study requires to improve the researcher’s understanding of the overall decision problem—how can one balance the seemingly conflicting requirements of lower/none the better in Agile and larger the better in ISO 9001:2008 QMS standards, as far as documentation is concerned?—starting from very low initial knowledge to gaining sufficient knowledge to solve the problem. This was taken into account in designing the appropriate iterative, panel-based problem solving process.

The researcher assumes that the case study is rigorous enough to make the findings of the study (the solution to the overall decision problem) *analytically generalizable*, in the sense, the findings can be used to explain (among other things) what documentation are likely to be useful and important in a Scrum environment to meet and sustain the quality assurance level envisaged in ISO 9001:2008 QMS standard.

As mentioned in the previous section, Delphi is an iterative process of converging towards a solution (typically 3 iterations although more iterations are sometimes needed) through sequentially designed questionnaires using the expertise of a panel of respondents. Similarly, in this research, 3 rounds of questionnaires were presented to a select audience (the composition of the panel was given earlier) that differed from one round of data collection to another (Figure 3.2), which is a departure from the Delphi approach (see Figure 3.3.). The survey questions in each round were based on two factors: feedback from the previous round and the target audience in each round (it could be argued that the target audience for each round was selected based on the nature of the questions in the questionnaire for that particular round).

Figure 3.3 depicts a Delphi process involving '*n*' number of iterations. This model was taken as a guideline for designing survey questions (for each round) and managing the overall decision making process, notwithstanding the two notable departures from the Delphi approach mentioned earlier (constraints on reaching a consensus and having to change the panel from one iteration to the other). Each element in Figure 3.3 is described in turn, in relation to the researcher's study.


Start
Identify research questions
Identify potential experts
Select panel of experts based on pre-defined criteria
Inform the panel
Design questions for Round 'n'
Present Questions to the Panel
Collect and analyze the responses
Achieved the goal?
Yes
Create the report
Stop
No
Evaluate feedback to design the new round

Note: Initially *n*=1.

Figure 3.3: A typical sequence of the Delphi approach (Source: Skulmoski et al., 2007)

3.5.2.1 Identify Research Questions

As outlined in chapter 2, the three research questions of the study emerged from the researcher's literature review.

3.5.2.2 Identify potential experts

As mentioned elsewhere, the target audience (respondents) of the questions (in the questionnaire) is a diverse group of representatives from software development teams who work in Scrum environments—mostly practitioners specializing in healthcare software development. The sub criteria that dictated the choice of different panel members for different rounds of data collection are given in subsection 3.5.2.4.

3.5.2.3 Select panel based on the pre-defined criteria

The members for the panel will be selected out of the potential experts based on the pre-defined criteria below. It is planned to select 20 members for the panel, where 5 members will be from an outside organization and 15 members from case study organization. The selection criteria were described thoroughly in section 3.4.4.1.

3.5.2.4 Inform the panel

Before presenting the questions to the panel, the potential panel members were briefed one-on-one basis about the purpose of the questions (in each round) and what is expected from the respondent. If a respondent showed confidence in responding to terms of reference of the draft questionnaire the respondent was included in the panel (otherwise the respondent was excluded). This criterion resulted in 3 different panels (some respondents were common) for the three different rounds of data collection/problem solving.

Design Questionnaires for Round 1: Round 1 contained two open-ended questions (see Appendix A for the relevant questionnaire); one question required the participants to list the universe of documents that are used in Scrum project while the other question required participants to list the different roles the Scrum team members play under the three broad categories, Product Owner (PO), Development Team (DT), and the Scrum Master (SM) (details given later). The questions for this round were open-ended questions because the knowledge on the subject domain was too low at the beginning to invite close ended type questions; in addition, the respondents were given full control in responding to each question such as to liaise with the other members in brainstorming. This was possible because in round 1 only 5 members were chosen (from a pool of 20) to participate (Figure 3.2). Table 3.5 shows the composition of the 5-member panel along with their expertise and tenure in the case study organization.

Table 3.5: The Role, Expertise and Experience of the Panelists

Role in the Development team	**Expertise**	**Experience in the Organization (Tenure)**
Development Team Lead	Team management, Sales, Scrum practices, Health-IT context knowledge	2 years
Project Manager	Project Management, Scrum practices, Health-IT context knowledge	2 years
Solution Architect / Technical Lead	Development guidelines, Technology, Team Management, Scrum practices	6 years
Test Engineer	Quality Assurance, Scrum practices, Health-IT context knowledge	3 years
Quality Management System Team Member	Quality Management systems, ISO 9001:2008, Scrum practices, Health-IT regulations	4 years

Present the Questions to the Panel: Since it was decided to keep the identity of each respondent anonymous, data collection rounds other than round 1 were silent rounds where the researcher (facilitator) met the participants in the case-study organization individually to distribute the questionnaire to gather data; this also gave the opportunity for the researcher to explain certain things that were not clear to the respondents in the questionnaire and also to interview the respondents where necessary. In the case of 5 respondents who were outside the case-study organization, the researcher dispatched the questionnaires via e-mail.

Collect and Analyze the Responses: Once the responses are collected data analysis is carried out to sort the responses. All the collected data were maintained in an excel sheet in researcher’s own computer, where data remained securely. For the purpose of decision making raw data were shared with the supervisors, yet used secured file transferring.

Check if the Goal is Being Achieved: After analyzing the data, if it was seen that the goal has been achieved, then data collection rounds were concluded to commence the creation of reports. If not, new questionnaires were created to represent next rounds, keeping in mind that each round should have its connection with the previous rounds.

Evaluate Feedback to Design the Next Round: Creation of the next questionnaire commenced if it was identified that the goal has not been achieved; the new questionnaire improves the questions again and focus more on the goal. It was identified that to get the best results out of these questionnaires, the questions should be focused on one target in each iteration; missing loops were filled in the next iterations based on the responses. Planed iterations for this research were 3 rounds.

Create the Report: At the end of each iteration all data were placed in to one place. Analyzing the patterns, trends, common answers will be helpful to make decisions.

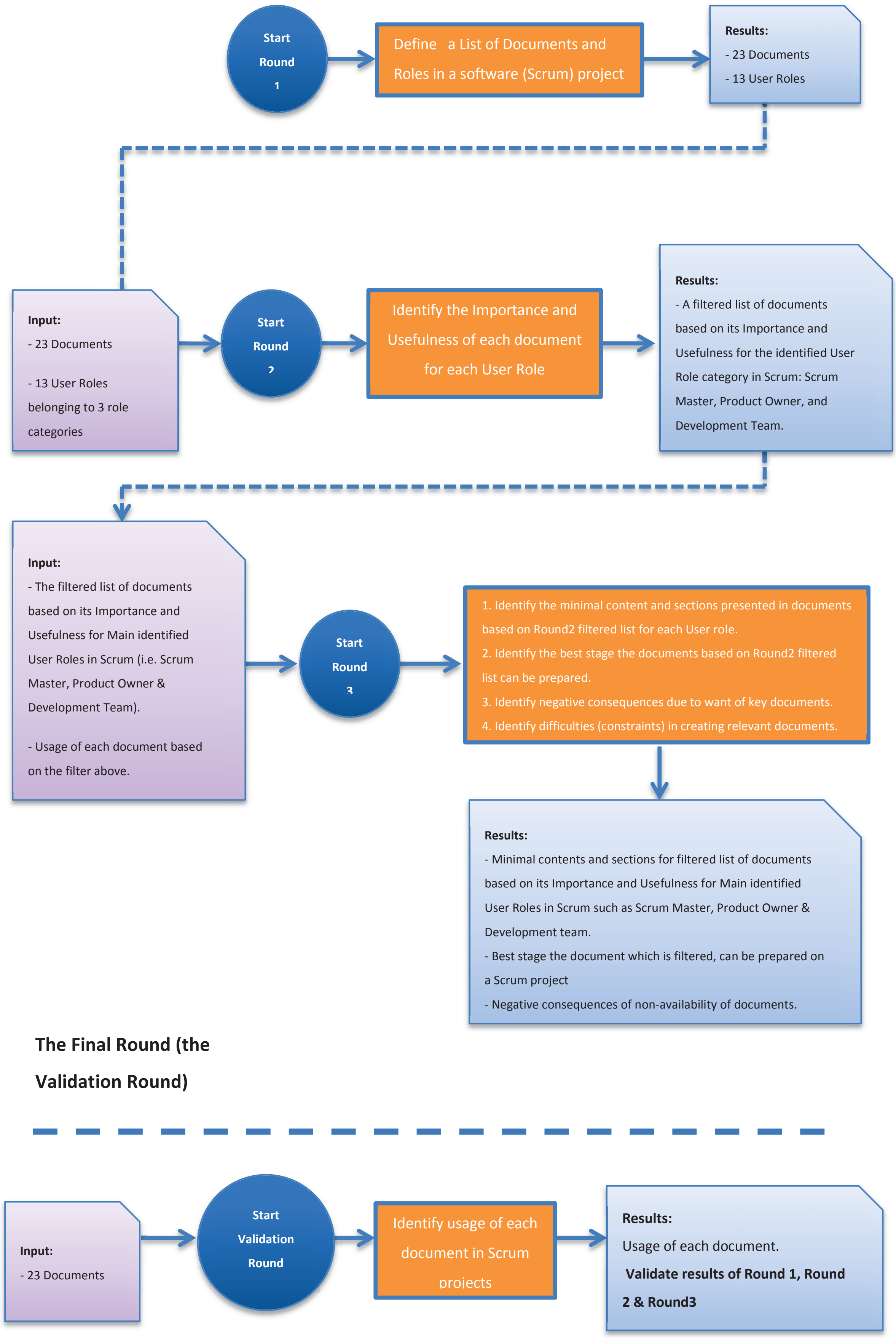

Start
Round
1
Define a List of Documents and Roles in a software (Scrum) project
Results:
- 23 Documents
- 13 User Roles
Input:
- 23 Documents
- 13 User Roles belonging to 3 role categories
Start
Round
2
Identify the Importance and Usefulness of each document for each User Role
Results:
- A filtered list of documents based on its Importance and Usefulness for the identified User Role category in Scrum: Scrum Master, Product Owner, and Development Team.
Input:
- The filtered list of documents based on its Importance and Usefulness for Main identified User Roles in Scrum (i.e. Scrum Master, Product Owner & Development Team).
- Usage of each document based on the filter above.
Start
Round
3
1. Identify the minimal content and sections presented in documents based on Round2 filtered list for each User role.
2. Identify the best stage the documents based on Round2 filtered list can be prepared.
3. Identify negative consequences due to want of key documents.
4. Identify difficulties (constraints) in creating relevant documents.
Results:
- Minimal contents and sections for filtered list of documents based on its Importance and Usefulness for Main identified User Roles in Scrum such as Scrum Master, Product Owner & Development team.
- Best stage the document which is filtered, can be prepared on a Scrum project
- Negative consequences of non-availability of documents.
The Final Round (the Validation Round)
Input:
- 23 Documents
Start
Validation
Round
Identify usage of each document in Scrum projects
Results:
Usage of each document.
Validate results of Round 1, Round 2 & Round3

Figure 3.4: Depicting the sequence of data collection and derivation of results

The next section (section 3.6) describes the construction of questions for the data collection round and associated details.

3.6 Constructing Questions for Data Collection Rounds

This section describes the objective of each data collection round along with the structure of the questions.

3.6.1 Constructing Questions for Round1

Round 1 was an entry level round. The objective of this round was twofold: (a) defining a list of documents that are being used in Scrum projects (at least the participants have seen these being used) and, (b) defining the roles typically being played by team members in Scrum projects (hereafter being called roles). As this round has a broad context (open ended questions) it was decided to select only few experts (5 to be precise) from the case-study organization; the experts were chosen so as to be as diverse as possible (different skills and knowledge levels). As mentioned earlier, the identity of the respondents was disclosed and the respondents were encouraged to brainstorm. The outcome of this round was emergence of a well sorted and defined list of documents as well as roles that can be seen in the development environment (role-plays). That outcome is then used in the second round for further classifications. In-detailed description of Round1 is described in the section 3.5.4.

3.6.2 Constructing Questions for Round2

Outcome of the Round1 was used as the input for the Round2. In Round2 the main focus was to further classify the document list, based on the role-plays, in-terms of the usefulness and importance the documents (i.e. which document is important to whom). For this round 19 participants were chosen of whom 15 were chosen from the case study organization. Thus 4 were chosen out-side the case study organization (from three other organizations that use the Scrum methodology). The composition of the participants and their experience are given in Tables 3.6 and 3.7. The reader should note that out of the 15 respondents chosen from the case study organization, only one did not respond; this was still a good outcome (93% response rate). All 5 respondents outside the case study organization responded.

Table 3.6: Summary of the Participants on Round2 from Case Study Organization

Role	**Number of Participants**	**Experience in the Case Study Organization (Years)**
Product Owner	1	3+
Team lead	2	2.5+
Business Analyst	2	3
Technical lead	1	3.5+
Developer	2	4+
Tester	2	3+
Quality Management Team member	1	4
Implementation consultant	1	4+
Technical writer	1	3.5
Customer support engineer	1	2.5+
Total	14	45.25+ (Weighted Average = 3.4 +)
Note: Details of the respondent who did not respond are not shown in this table.		

Table 3.7: Summary of the Participants on Round 2 from Outside the Case Study Organization

Role	**Number of Participants**	**Average Years of Experience in the Organization**
Team lead	1	2
Technical lead	1	2+
Developer	1	1+
Tester	1	1.5
Technical writer	1	2

Round2 contained two tasks described below.

Task 1: Understanding the Usefulness of Each Document for Each Role-Play

Task 1 focuses the attention of participants on a list of documents (the list being prepared based on information provided by the recipients from Stage 1) used /produced in Scrum projects by different roles plays. The participants were requested to state which document(s) is/are of benefit/useful to perform each role in their day-to-day tasks.

Task 2: Understanding the Importance of Each Document for Each Role

In Task 2, participants were requested to state the importance (in a scale of 1 through to 5) of each document for each role. The 5-point scale used is as follows:

1= extremely important;

2 = Important;

3 = neither important nor unimportant;

4 = Not important;

5= Extremely Unimportant

Table 3.8 depicts a sample response for illustrative purposes. Having received the responses, the researcher classified them based on the main 3 Scrum roles identified in a Scrum projects: Product owner (PO), Scrum master (SM), and Development team (DT).

Table 3.8: A Sample Response for Task2

	Roles												
Document	**Customer / stakeholder**	**Sales people**	**Product owner**	**Project manager or team-lead**	**Business Analysts**	**Technical lead**	**Developers**	**Testers**	**User Interface developers**	**Technical writes**	**Customer support engineers**	**Implementation consultants**	**Quality assurance team members**
Maintained product backlog	**1**	**4**	**1**	**2**	**2**	**2**	**2**	**3**	**3**	**4**	**4**	**2**	**3**

3.6.3 Constructing Questions for Round3

The first task undertaken in Round3was to identify what content should be included in each *important* document (the previous round was designed to identify the important documents for each role-play) for each of the three role categories (PO, SM, and DT), given the QMS requirements stipulated in ISO 9001:2008. The second task undertaken in Round 3 followed from the first task. The objective of the second task was to identify what stage of a Scrum project the documents should be prepared. Round 3 also included accomplishing two other tasks (i.e. Task 3 and Task 4).

To participate for the Round 3, 9 participants were chosen, 3 from each role: PO, SM, and DT. Table 3.9 provides the details of the participants.

Table 3.9: Summary of the Participants on Round2 from Case Study Organization

Role	**Years of Service in the Role Category**	**Years of Service as an IT Professional**
Development Team (DT)		
DT1	10+	10+
DT2	5+	8+
DT3	4+	7+
Scrum Master (SM)		
SM1	2+	10+
SM2	7+	10+
SM3	1+	3+
Product Owner (PO)		
PO1	1+	5+
PO2	1+	4+
PO3	2+	3+

The details on each of the 4 tasks in Round 3 are described below.

Task 1: Identifying what should be written in a document by taking the role category into account

In Task1 participants were requested to state what content (the headings/captions for each content was mentioned in the questionnaire) and sections that they would like to see on each document—identified as "important" in the previous round—when they are playing the role of PO, DT and SM respectively.

The following content (more specifically content headings) for general sections and main sections (it is typical for software document to have general sections as a forerunner to the main sections) were listed (Table 3.10) for each document (identified as important based on Round 2 results); for each document, each respondent was asked to select (by indicating a "yes") which content (more specifically content headings) are relevant to SM, PO, and DT respectively.

Table 3.10: Content Headings the Respondents Were Asked to Choose from as Relevant for Each Document and Each Role Category

General Sections	Main Sections
* Purpose	* User stories
* Author	* Flow charts/diagrams
* Approvers/Reviewers/Sign off	* Development instructions
* Internal communication records	* Release schedules/notes
* Corrective action records	* Known issues
* Preventive action records	* Verification and validation information
* Comments on validity (dated/relevant)	

The reader should note that the content headings were generated based requirements stipulated in ISO 9001:2008 under sub-clause 4.2.4 (Control of Records). Further information is given in section 2.4.13.

Task 2: Identifying when the documents should be prepared in a Scrum project

Task 2 was designed to identify at which stage of a project a document should be produced /written /generated. In this regard, the following 8 stages (these stages cover the full project cycle, as depicted in Figure 3.5) of a Scrum project were listed in the questionnaire and each respondent was asked to indicate the relevant stage (by means of a "Yes") each document should be prepared. The 8 stages listed in the questionnaire were: (1) project initiation, (2) at the commencement of project development, (3) at the commencement of a sprint, (4) whilst the sprint is in progress, (5) at the end of a sprint, (6) at the end of a release, (7) Anytime during the project cycle, and (8) After the end of product release.

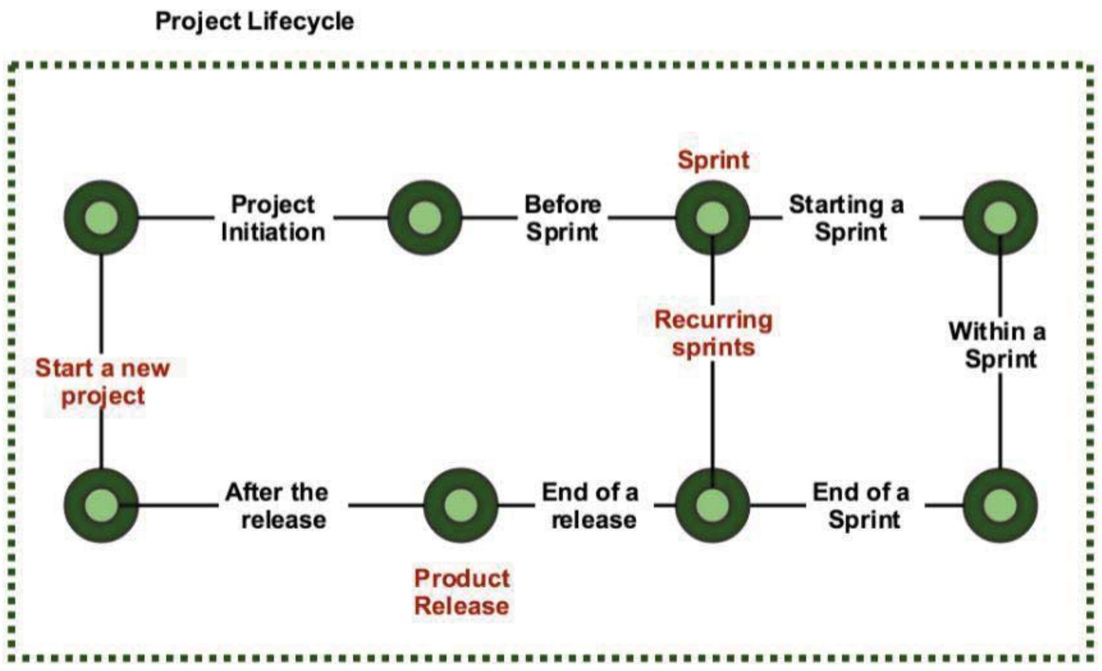


Figure 3.5: Stages of a Scrum project.

Task 3: Recollecting Negative Consequences

Task 3 was designed to enable the researcher (who acts as the facilitator in the 3-stage problem solving process) to focus on collecting information pertaining to situations that the respondents have come across where non-inclusion of relevant/important content (see Table 3.5 in Task 1) for the relevant roles has resulted in product quality issues (e.g. re-work, customer complaints). The researcher collected the relevant information (negative experiences) by way of conducing short interviews (30-45 min) on a one-on-one basis with each of the 9 respondents. The interviews, which initiated through the open-ended question *"tell me some situations that you have come across where non-inclusion of the content which you have rated as relevant and important"*, were recorded using the researcher's iPhone, with the permission of the respondents. The interviews were transcribed (using MS Word) and the transcripts of the 9 interviews (corresponding to 9 responds) were shown to the respective respondent. Similar themes between the 9 transcripts were identified using a separate color for each theme (MS Word's text highlight function was used). These common themes were then tabulated to be produced as results on Task 3.

Task 4: Identifying Difficulties on Creating Relevant Documents in a Scrum Project

The final task of Round 3, Task 4 was designed to understand the difficulties Scrum teams face (as individual members as well as a team) in creating important/relevant documents for the project. The definition of relevant/important documents was the same as that used for Task 3.

The researcher collected the relevant information (difficulties the respondents face in creating important documents) by way of conducing short interviews (30-45 min) on a one-on-one basis with each of the 9 respondents. The approach used for Task 4 was similar to that used in Task 3. The interviews initiated through the open-ended question *"please tell me the difficulties that you/your team faces when producing important documents in your project"*. To facilitate the answer examples of constraints (difficulties) a team may face were given: resource limitations, time pressure, budget, team member motivation or attitudes, company policy, and rules and regulations. As in the case of Task 3, the interviews were recorded using the researcher's iPhone, with the permission of the respondents. The interviews were transcribed (using MS Word) and the transcripts of the 9 interviews (corresponding to 9 responds) were shown to the respective respondent. Similar themes between the 9 transcripts were identified using a separate color for each theme (MS Word's text highlight function was used). These common themes were then tabulated to be produced as results on Task 4.

The reader will note that Task 4 shares many similarities with Task 3. However, Task 4 was executed subsequent to Task 3 on a separate day to eliminate respondent burnout.

3.7 Validation of the Results

After completing the three data collection rounds described above, a final round of data collection was conducted to validate the key results. Objective of the validation round was to collect data about usage of documents in each project. These data were then analyzed and the results were used as a threshold to compare or validate the results received from Round1, Round2 and Round3.

The participants selected to this round were new participants who did not involve in any other data round. There are 20 participants chosen for this round. Out of 20, 15 are from the case-study organization and 5 from 3 different organizations that have Scrum practices in-place.

Selection criteria for the participants were same as Round1 general criteria. Out of the 20 participants, 19 responded which made the response rate 95%. Summary of the participants who responded is shown below.

Since the questionnaire used in the validation round was not complex (just a solitary task described below), the participants were given 1 week to complete the question. All participants completed the questionnaire with in the given time frame; hence no additional time was required.

The Task Assigned to the Panelists in the Validation Round: Understanding the Usage of Each Document

Here the researcher focused their attention on the usage of each document in participant's current project. Initial document list, which was produced in Round1 of data collection was used in the validation round also. If a specific document had been used by a participant, t the participants were asked to state "YES", while if a specific document had not been used the participants were asked to state "NO". However, if the response was a "NO", the participants were given 5 reasons (see below) to choose form for not using the specific document.

The 5 choices given to the respondents were:

a. I never knew this can be considered as a document
b. I don’t see any value using this document
c. With the tight schedule we can’t waste time on writing this
d. We don’t have enough resources
e. Other

After collecting the responses, they were recorded in an Excel spreadsheet. The results received were used as a threshold, to compare the results of Rounds 1, 2 and 3.

Results are detailed in Chapter 4.

Following table (Table 3.11) shows a sample of responses received for the validation round.

Table 3.11: Sample Answer Received for the Validation Round for a Specific Document

DOCUMENT		Is this document being created/used by your current project?	If 'No' the Reason/s (Only the code/s a, b, c, d, e)
1	Maintained product backlog	Yes	

3.8 Summary

Chapter 3 described the research methodology. The researcher justified why the case study approach was chosen for this research, why Delphi-like panel-based data collection was appropriate and how the data collection rounds and the questions were constructed. This chapter also covered data collection methods, sample selection (selection of the participants) and the characteristics of the sample (the expertise and work experience) Finally, the chapter describes how certain key results collected in the 3 rounds of data collection were validated using an additional data collection round known as the validation round.

Following table (Table 3.12) depicts the link between each research questions (RQs) and the methodology followed.

Table 3.12: Research Questions and Data Collection Rounds

Pre-condition: - Given ISO 9001:2008 QMS accreditation is identified as the QMS baseline in the software development company - Given that Scrum is the primary software development method used in the software development company		
ID	**Research Question**	**How Each Research Question Has Been Answered**
RQ1	*What specific operational-level documentations are considered useful and important for a healthcare software development company practicing Scrum to comply with the ISO 9001:2008 QMS standard?*	In Round1, using 5 respondents, the researcher identified as many as 23 potentially useful and important documents and 13 roles in software development (classified under Scrum Master, or Product Owner, or Development Team). In Round2, using 19 respondents, the researcher identified to whom (out of the 13 roles) that each document can be important ad useful.
RQ2	*How can the documents be prepared in a timely and concise manner to maintain good communication among different parties in the development lifecycle?*	In Round3, using 9 respondents, the researcher attempted to identify: what *content* and *sections* should be included in a document based on the role category; at which stage of the Scrum project each document should be prepared.
RQ3	What are the issues that cause documentation processes to be less effective, in a typical Scrum environment?	Also in Round3, the researcher attempted to understand: the product quality issues that arise when a particular document is poorly

Pre-condition: - Given ISO 9001:2008 QMS accreditation is identified as the QMS baseline in the software development company - Given that Scrum is the primary software development method used in the software development company		
ID	**Research Question**	**How Each Research Question Has Been Answered**
		prepared (e.g. less sections or content provided); the practical difficulties the Agile community faces in preparing the useful and important documents. To support with the findings, researcher is using secondary sources such as reports and previous case-studies to draw a conclusion.
Validation To check the validity of the results, Final data collection round is carried out to check the usage of the documents, in Scrum teams. Results are used as a threshold to compare the outputs of previous rounds.		

CHAPTER 4
RESULTS AND DISCUSSION

4.1 Introduction

This Chapter focuses on presenting and discussing the results of each data collection round described in Chapter 3 to answer the three research questions. There are several ways in which the findings of a social research could be presented. Five commonly used criteria are given below (Kumar, 2005; Leszak, Perry, & Stoll, 2000).

- By instrument: If a study involves use of multiple data collection methods (e.g. interviews, surveys, document inspection) the findings can be presented and discussed by each data collection method being used.
- By research question: The findings can be presented by research question (e.g. findings on RQ1, findings on RQ 2 and so on).
- By individual: The findings can be organized by each individual (respondent) being involved in the study. This approach suits well- when the number of participants in a study is limited.
- By group: The findings can be organized based on the participants' common characteristics by forming logical groups (e.g. frontline workers, middle managers, and top managers).
- By issue: The findings can be presented based on the issues being identified.

However, in this research, the findings are presented by research question because data collection rounds tally with research questions quite well. As shown in Table 3.12 in the previous chapter, data collection rounds 1 and 2 were meant to answer the first research question (RQ1), round 3 was meant to answer the second research question (RQ2) and the third research question (RQ3), although secondary sources were also used to answer RQ3. Furthermore, the results from the validation round were used in the discussion of the results.

This chapter consists of 4 main sections. Section 4.1 presents the results on RQ1; section 4.2 presents the results on RQ2; section 4.3 presents the results on RQ3, while section 4.4 presents the results on the validation round, along with an accompanying discussion of the results. Finally, section 4.5 summarizes the findings.

4.2 Results for the Research Question 1 (RQ1)

The first research question (RQ1) asks: *What specific operational-level documentations are considered useful and important for a healthcare software development company practicing Scrum to comply with the ISO 9001:2008 QMS standard?* As mentioned in the previous chapter, this research question is fully answered through two rounds of data collection.

It is important to note that the term 'useful' in the statement above is used to mean benefit(s) of a course of action—in this case, use of appropriate documentation. The term 'import' is used to mean the *value* of a specific course of action (Ghosh, 2015). In the first reading, one may find both terms deceptively similar.[6]

4.2.1 Round 1 of the Study

As stated in the previous chapter (methodology), the objectives of Round1 of data collection were twofold: (i) to generate a list of documents which is perceived by the users to be relevant in Scrum projects, and (ii) to identify the different software development roles being played by team members in Scrum projects.

Question 1 of the survey (see Appendix A) in round 1 required the panel members (*n* = 5) to adopt a consensus process to list: (a) documents that are either currently being used in Scrum projects, or (b) documents that are not being used but might be important or useful for the Scrum projects, or (c) documents that have been used in Agile projects other than Scrum (based in panel members' personal experience or what they have heard from their counterparts elsewhere) that might also be useful in Scrum projects.

[6] As an example for usefulness, Ghosh (2015) cites aspirin; Aspirin reduces symptoms (e.g. pain, fever) and to that extent is to a patient. As an example for importance, Ghosh cites water, in the context of life on earth; without water, life cannot be sustained. Hence water is important to all life.

4.2.1.1 Results for Question 1 (Round 1)

The participants were able to list as many as 23 documents through consensus decision making. Each document is described in turn, along with supporting literature where necessary, to further explain the documents.

Doc_ID_1: Maintained product backlog:
'Maintained Product Backlog' is an ordered list of requirements that might be needed in a product (Schwaber & Sutherland, 2013). It is considered as a single source of requirements, if the product needs any change in the future (Schwaber & Sutherland, 2013). Based on the product backlog, development teams can pick items to be worked on the sprints (short iterations of development).

Doc_ID_2: International Specifications
"International Specifications" are the documents that provide industry requirements and standards world-wide (ISO, 2009). Some of the products developed by the case study organization have to adhere to international specifications to conform to industry standards (e.g. IHE, HL7, and FHIR). Those products which are developed according to the specifications always uses summary of the specifications showing which part of the specification is applicable to the development.

Doc_ID_3: Standards & Regulations
Based on the product and the targeted market, standards and regulations a product has to follow vary. The panel was unanimous in their decision that in order to appraise the development team (and other interested parties) of the relevant industry standards and regulations to be followed must be made available by product owners. As an example of a regulation, the panel mentioned that electronic health products that are sold in the USA from 2013 onwards have to have the MU2 (Meaningful Usage 2) certification. As stated in official USA Healthcare software website, MU2 is a regulation stipulated by USA federal government to "Improve quality, safety, efficiency, and reduce health disparities. Engage patients and family. Improve care coordination, and population and public health. Maintain privacy and security of patient health information" (HealthIT, 2013).

Doc_ID_4: Stakeholder/Customer requirements

Customer/stakeholder requirements are the actual needs and expectations of the customers (IBM, 2005). The panel was unanimous in their decision that customer needs/stakeholder requirements need to be documented (and updated as the project moves along) because the fundamental element of any product design is identifying customer/stakeholder requirements properly. It is important to note that customers/stakeholders do not necessarily have to be external users in traditional sense; they can also be internal customers or colleagues within the same organization who depend on one's team (Radigan, 2015). The panel opined that it is required to document customer/stakeholder needs/requirements in a form, which is available, current, and accessible to the development team. The panel also opined that depending on the teams and the interested parties, the format of the customer/stakeholder requirement document can vary.

Doc_ID_5: Business Flows

Business flow shows the processes that take place in a business (VanBaren, 2015). It shows steps of a process from start to finish. The panel mentioned that although various methods and notations are used in the industry to depict business processes, the "Business Process Model and Notation" (BPMN) is the commonest tool in the IT industry.

Doc_ID_6: Road Map Items (RMI)

Since the road-map Items is a plan that matches short-term and long-term product goals with specific solutions and time-line schedules, participants thought this as a good addition to the list of documents. Such a plan can also be described as feature-level work that encompasses many user stories (Radigan, 2015). This is also referred to as "epics" in some organizations. Often the roadmap is created by looking at the product backlog and upon achieving a road-map, development team considered it as achieving a specific goal.

Doc_ID_7: Features

Features are the smallest units of work. In some organizations features are also called "User-stories". The goal of a feature is to deliver a particular value back to the customer (Radigan, 2015). Without a feature it is hard to define the work that a development team should do, therefore this is again considered as an important document to be added in the list of documents.

Doc_ID_8: Test Verification Report

After developing a feature, it is passed to the testers in the team. They produce a report of the tests carried out on that feature (Mansour, 2013). The report can contain manual or automation test scripts, results and supportive documents such as screen shots. This report can be linked to the same feature, so that verification notes can easily be found (Radigan, 2015). Having a better maintained test verification reports in a development team helps to achieve traceability, hence this is considered as a valid entry to be added to the list of documents.

Doc_ID_9: Sprint demonstration recordings

Sprint demonstrations are usually held at the end of the Sprint (Schwaber & Sutherland, 2013). The Development Team demonstrates the ‘feature work’ that they have completed and clarify any questions with the audience (Pino et al., 2010). It is important to choose the right audience for the demos as this is a place where the development team can engage with the key stakeholders in the development cycle (the engagements are recorded). These recordings can be kept in any form preferred by the organization. The panel argued that it is important to keep the recordings available to be viewed by the interested parties.

Doc_ID_10: Retrospective notes

A Sprint retrospective meeting, typically called the Sprint Retrospective, is an opportunity for the Scrum Team to understand the effective and ineffective processes (Schwaber & Sutherland, 2013). As Scrum teams are self-organizing, they can discuss and see what went well and what went wrong in the previous sprint, and thereby make improvements to be effected during the next sprint (Pino et al., 2010). The Sprint retrospective meetings usually occur after the sprint demo and prior to the next sprint planning meeting (Schwaber & Sutherland, 2013). The panel opined that maintaining notes of each retro meeting is helpful to see how well the team has been able to overcome problems.

Doc_ID_11: Training materials

The training materials are basically provided to the internal staff of the organization. If any team member wants to learn about a new product in the organization, they can follow the training material; these are presented in a form of online-course material. Not having training material will add significant maintenance burden on the development teams after production. Therefore, preparing these material in the development cycle is important and thus included in the list of documents.

Doc_ID_12: Risk Analysis reports

Risk analysis reports are created to understand the risks and suggest mitigation activities (Hossain, Babar, Hye-young, & Verner, 2009). Risk is considered as a danger to an individual, businesses or government due to natural or human events. In Information Technology, risk analysis reports are mostly used to align technology related objectives with business objectives (Hossain et al., 2009). However, in the context of health-IT, most risks are attributable to patient privacy and information safety (HealthIT, 2013).

Doc_ID_13: Technical Design

Technical design includes the design of the product based on the customer requirements (Yunan, 2013). These designs are usually being reviewed by a panel of technical experts as their approval is sought before actual development. The documents can be in a form as long as they are accessible and available to the development teams in the organization.

Doc_ID_14: DB Designs

DB designs (Database) contain detailed information on data models in data base hence it was agreed to add this in to the list of documents. DB Designs can be in a form which is accessible and available to development teams in the organization.

Doc_ID_15: Developer Documentation

Similar to DB designs and technical designs, developers maintain developer documentations, which are a combination of a subset of DB and technical designs. Usually these documents are produced for individual development modules. These documents can be in any form as long as they are accessible and available to development teams in the organization.

Doc_ID_16: Requirements Traceability Matrix

Requirements Traceability Matrix is a tool that is used to show the test coverage for requirements (Seela, 2015). It maps the requirement with corresponding tests done in the code and the functionality level. Also, traceability requirement matrices provide an overview, which is useful to identify coverage gaps in requirements and verification (Seela, 2015). This can be made by testers in particular, and can be made available to all interested parties in a form of a chart or table.

Doc_ID_17: Unit Test and Test Reports

Unit testing is a way of testing the smallest testable parts in an application (Rouse, 2007). Usually these are individual unit of codes which are developed to test a completed function. This is often automated, however manual tests also be done (Rouse, 2007). The test reports are helpful to understand the code coverage, and test coverage for each individual item.

Doc_ID_18: Source Code Files

Source code is the "source" of the program (Thompson, 2007). It consists of all the variable declarations, parameter assignments, loops, instructions, functions, and other statements which direct the program on how to function.

Doc_ID_19: Bug Tickets

Bug is an identified defect in a product. A bug can be either incorrect functionality or unexpected behavior in the product. Usually testers find bugs and report it in a form that developers can easily re-produce and fix them by following the information in the bug ticket. In the case study organization, the format of the bug tickets is pre-defined with content such as ‘environment details’, ‘steps to reproduce the issues’, ‘expected behavior’ and ‘actual behavior’ and any supportive information such as log files and screen shots.

Doc_ID_20: Test Plans

A test plan is a document which consists of planned activities for testing for a product version (Seela, 2015). It is often created at an early stage of the development (Cristal et al., 2008). Test plans contain details such as the test objectives, targeted market and audience, processes for a specific test. It is usually presented to a mixed group of audience, with a view to seek approval in a meeting called 'Test entry meeting'; this to ensure that everyone in the development team knows what is going to be tested and how.

Doc_ID_21: Bug Verification Report

Bug verification reports are done to show the information and results of the tests done to verify a bug (Nashwille, 2015). These reports can include information about the tests, screenshots, exploratory tests and regression to show that no existing function is regressed by the bug fix (Nashwille, 2015). It is always recommended to attach the verification report to the bug ticket, so that all the information can be found in one place.

Doc_ID_22: User Manuals

User manuals are the guidance documents that help the users. The purpose of these documents is to provide knowledge and details to the customers, so that they can use the products with minimal help and supervision from development teams or technical staff (Cajander, Larusdottir, & Gulliksen, 2013).

Doc_ID_23. Companywide Quality Procedures

In-order to prevent mistakes and defects in the products, most companies follow companywide quality procedures (McCluskie, 2013). The companywide quality procedures work as an umbrella, under the hood, development teams are following sub-set of procedures and principals depending on their product and the targeted market (McCluskie, 2013). These procedures are documented in a way that it is accessible by the whole organization. Also, companywide quality procedures are very dynamic (they change rapidly) due to international regulations and market changes. The case-study organization requires its employees to undergo awareness training in-case of a major change of quality procedures.

4.2.1.2 Results for Question 2 (Round 1)

The panel identified 13 important roles relevant to software development (Scrum) projects. The roles are named using the terminology used in the case study organization as majority of the participants in the study come from the case study organization. The 13 roles are described as follows, along with supporting literature. Each role has given a title which labelled as “Role_ID_#” and description of the role is given along with its title.

Role_ID_1: Product Owner

Product Owner is the person who is responsible of enhancing the value of the product and the work of the Development Team by introducing new features according to the customer requirements and current market trends. The Product Owner directs the development team by conveying his/her vision to the team, outlining work in the Product Backlog, and prioritizing it based on business value (Schwaber & Sutherland, 2013).

Role_ID_2: Customer / Stakeholder

A stakeholder /customer is any person, organization, social group who has a vital interest in the business or its activities (Grimsley, 2003). Customers/stakeholders do not necessarily have to be external users in a traditional sense; they can also be internal customers or colleagues within the same organization who depend on one’s team (Radigan, 2015). Stakeholders can affect a business, or be affected by a business or be both affected by a business and affect a business (Grimsley, 2003).

Role_ID_3: Project Manager or Team-lead

In the case study organization, most of the time Project managers are acting as team leads. However, in most of the teams they have a team-lead who is not a project Manager. Because of that reason, in this study both roles have combined together. A project manager/team lead is the person who is responsible of leading a project from its commencement to execution (PMI, 2011). The project manager/team lead manages tasks including planning, execution and managing the people, resources and scope of the project.

Role_ID_4: Business Analysts

Business analysts "identify and articulate the need for change in how organizations work, and to facilitate that change by defining needs and recommending solutions that deliver value to stakeholders" (IIBA, 2010).

Role_ID_5: Technical Lead

A Technical Lead is a developer who is responsible for leading a development team (Kua, 2014). They have to have the right balance on technical skills and leadership skills. Usually a technical lead is assigned to one or more teams.

Role_ID_6: Developers

Developers are the people who write the code and implement the functions. They are engaged in functions such as design, actual implementation, and unit tests.

Role_ID_7: Testers

Testers are the people who check the application to find bugs. Testing is a way of validating the software over the requirements before it is being released for production.

Role_ID_8: User Interface Developers

These developers basically focus on the user interface design. They are responsible of making attractive yet user friendly designs.

Role_ID_9: Technical Writes

Technical writers engage on technical writing which requires direction, instruction and explanation to its users. The documents produced by technical users provide an efficient and clear way of explaining something and how it works (Parker, 2015).

Role_ID_10: Implementation Consultants

Implementation consultants work as consultants in customer sites at the early stages of the production. They have skills and knowledge of the products produced by the company; implementation consultants require good negotiation and problem solving skills.

Role_ID_11: Customer Support Engineers

Customer support engineers play a role similar to implementation consultants. However, support engineers do not work in customer sites. They work in the organization; they keep in contact with the on- going customers and help to resolve technical issues.

Role_ID_12: Quality Assurance Team Members

Members of the quality assurance team are the ones who make processes, standards and documentations around companywide QMS procedures.

Role_ID_13: Sales Staff

Sales staff work closely with the customers. Their responsibility is to sell the products and gain company revenue.

The list of roles mentioned above, as identified by the panel, are the common roles pertaining to the case-study organization. However, in the context of Scrum, there are only 3 primary roles present: The Product Owner, the Scrum Master, and the Development Team (Schwaber & Sutherland, 2013). Role descriptions for each of the primary roles are as follows.

Product Owner (PO)

In Scrum, PO is responsible of enhancing the value of the product and the work of the Development Team by introducing new features according to the customer requirements and current market trends. The Product Owner directs the development team by conveying his/her vision to the team, outlining work in the Product Backlog, and prioritizing it based on business value (Schwaber & Sutherland, 2013).

Scrum Master (SM)

Scrum Master is responsible for ensuring that Scrum practices are understood well within the Scrum team. This is done by ensuring that the Scrum team adheres to theories, practices, and rules presented with Scrum (Schwaber & Sutherland, 2013) and by helping to resolve impediments during sprints. However, in the case study organization, there is no specific role defined as the 'Scrum Master'. The responsibilities are shared across members of the development team particularly among the Team Lead, Project Manager, and the Business Analyst. This is the reason why the role 'Scrum Master' did not immerge in response to round 1 data collection (the questionnaire in Appendix-B). For this reason, in the subsequent rounds of data collection, several roles were consolidated under the role 'Scrum Master' (Table 4.1).

<u>*Development team (DT)*</u>

The Development Team consists of professionals (from different functional areas) who perform work that lead to delivery of a potentially releasable product at the end of each Sprint (Schwaber & Sutherland, 2013). The development Team is also known as the Cross-functional team. In cross-functional Agile team members are not designated with unique job titles; rather, they share responsibilities of all the necessary roles to successfully deliver the software. Table 4.1 shows the composition of the team members who have been classified under the 'Development Team'.

Table 4.1: Categorizing Roles under Main Scrum Roles

Product Owner	**Scrum Master**	**Development Team**
- Product Owner - Customer / Stakeholder	- Project manager or team-lead - Business Analysts	- Technical lead - Developers - Testers - User Interface developers - Technical writes - Implementation consultants - Customer support engineers - Quality assurance team members
Out of the 13 roles named by the respondents (based on the questionnaire given to them in round 1), the role 'sales staff' is excluded because they are not an integral part of a Scrum Team.		

Based on the above categorization, the five participants in round 1 represent the roles shown below (Table 4.2).

Table 4.2: Round 1 Participants Breakdown by Their Role.

Product Owner	**Scrum Master**	**Development Team**
Round 1		
Strategic Project Manager	Development team lead	Solution Architect Test Engineer Quality Manager

4.2.2 Round 2 of the Study

As defined in the previous chapter, the objective of Round 2 is to identify the usefulness and importance of each document, based on a given role. As mentioned in the previous chapter, in round 2, a questionnaire (Appendix-B) was distributed among 19 respondents to accomplish the following two tasks: identifying the usefulness of each document for each role (Task 1) and identifying the importance of each document for each role (Task 2).

The 19 participants in round 2 represent the roles shown below (Table 4.3).

Table 4.3: Round 2 Participants Breakdown by Their Role

Product Owner	**Scrum Master**	**Development Team**
Round 2		
Product Owner (n =1)	Team lead* (n = 3) Business Analyst (n = 2)	Technical lead* (n = 2) Developer* (n = 3) Tester* (n = 3) Quality Management team member (n = 1) Implementation consultant (n = 1) Technical writer* (n = 2) Customer support engineer (n = 1)
* One member outside the case study organization.		

4.2.2.1 Obtaining Results for Task 1: Identifying the Usefulness of Each Document

Based on the responses received for task1 through the questionnaire (Appendix B), where each respondent was presented the list of 23 documents (identified from round 1of data collection) to be assessed for usefulness of each document, the following heuristics were used to identify the degree of usefulness of each document for each role.

Heuristic 1: If a particular document receives 75% or more votes for a certain role, then, that document is considered to be a *very useful* document for that particular role.

Heuristic 2: If a particular document receives less than 75% but 50% or more votes for a certain role, that document is considered to be a *useful* document for that particular role

Heuristic 3: If a particular document receives less than 50% votes for a certain role, that document is considered to be a *not useful* document for that particular role (hence not considered in the analysis report).

The votes received for each document, against each role, were counted and divided by the total number of respondents to determine the proportions (the percentage figures) to apply the above heuristics. Table 4.4 shows a portion of researcher's data collection sheet, containing the votes. Four documents and three roles are recorded as a sample.

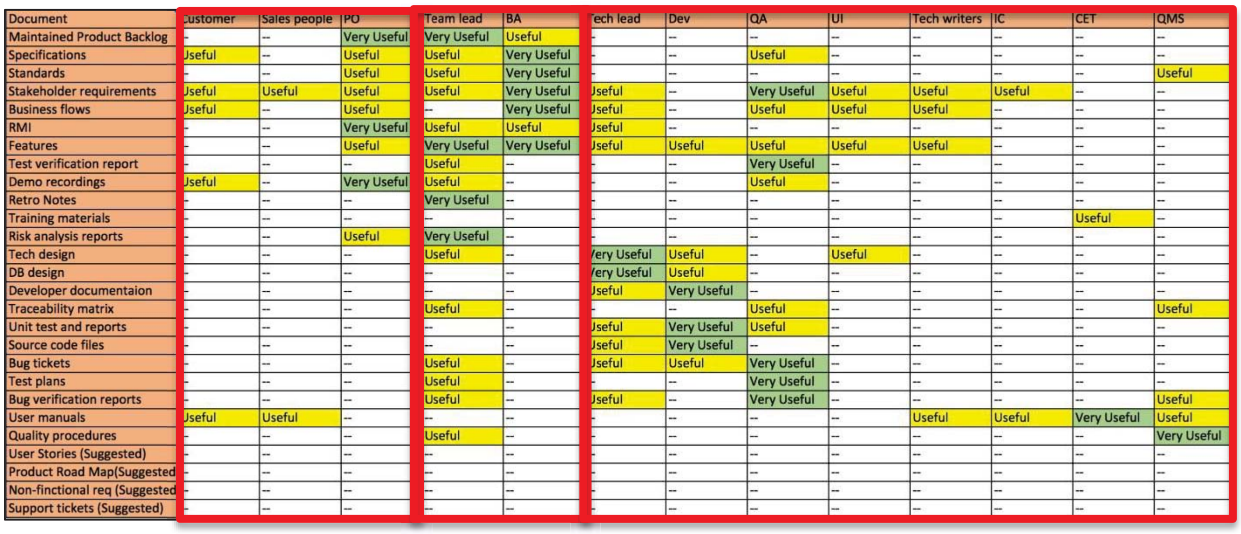
Table 4.4: Sample of Total Counts Received for Four Documents for Each Role

Document	Customer	Product Owner	Team Lead
Maintained Product Backlog	8	16	15
Specifications	10	10	10
Standards	8	10	10
Stakeholder requirements	13	13	11

In determining the proportions, although the panel size was 19, the effective number of respondents had to be considered as 18, because the responses given by one respondent was thoroughly incomplete; these incomplete responses were not counted in composing the spreadsheet containing the votes. As an example, the proportion of votes received for the document 'Maintained Product Backlog' as being useful for customers was 8/18, which is 4.4%, hence not deemed a useful document for customers, based on the heuristics used by the researcher.

Table 4.5, which is a screen dump from the researcher's spreadsheet, depicts the usefulness of each document for each role. The reader should note that the blank cells (dash) in Table 4.5 denote documents that are *'not useful'* to a particular role.

Table 4.5: The Degree of Usefulness of Each Document for Each Role



Product Owner Scrum Master ←---------------------------Development Team --------------------------------→

Document	Customer	Sales people	PO	Team lead	BA	Tech lead	Dev	QA	UI	Tech writers	IC	CET	QMS
Maintained Product Backlog	-	--	Very Useful	Very Useful	Useful	-	--	--	--	--	--	--	--
Specifications	Useful	--	Useful	Useful	Very Useful	-	--	Useful	--	--	--	--	--
Standards	-	--	Useful	Useful	Very Useful	-	--	--	--	--	--	--	Useful
Stakeholder requirements	Useful	Useful	Useful	Useful	Very Useful	Useful	--	Very Useful	Useful	Useful	Useful	--	--
Business flows	Useful	--	Useful	--	Very Useful	Useful	--	Useful	Useful	Useful	--	--	--
RMI	-	--	Very Useful	Useful	Useful	Useful	--	--	--	--	--	--	--
Features	-	--	Useful	Very Useful	Very Useful	Useful	Useful	Useful	Useful	Useful	--	--	--
Test verification report	-	--	--	Useful	--	-	--	Very Useful	--	--	--	--	--
Demo recordings	Useful	--	Very Useful	Useful	--	-	--	Useful	--	--	--	--	--
Retro Notes	-	--	--	Very Useful	--	-	--	--	--	--	--	--	--
Training materials	-	--	--	--	--	-	--	--	--	--	--	Useful	--
Risk analysis reports	-	--	Useful	Very Useful	--	-	--	--	--	--	--	--	--
Tech design	-	--	--	Useful	--	Very Useful	Useful	--	Useful	--	--	--	--
DB design	-	--	--	--	--	Very Useful	Useful	--	--	--	--	--	--
Developer documentaion	-	--	--	--	--	Useful	Very Useful	--	--	--	--	--	--
Traceability matrix	-	--	--	Useful	--	-	--	Useful	--	--	--	--	Useful
Unit test and reports	-	--	--	--	--	Useful	Very Useful	Useful	--	--	--	--	--
Source code files	-	--	--	--	--	Useful	Very Useful	--	--	--	--	--	--
Bug tickets	-	--	--	Useful	--	Useful	Useful	Very Useful	--	--	--	--	--
Test plans	-	--	--	Useful	--	-	--	Very Useful	--	--	--	--	--
Bug verification reports	-	--	--	Useful	--	Useful	--	Very Useful	--	--	--	--	Useful
User manuals	Useful	Useful	--	--	--	-	--	--	--	Useful	Useful	Very Useful	Useful
Quality procedures	-	--	--	Useful	--	-	--	--	--	--	--	--	Very Useful
User Stories (Suggested)	-	--	--	--	--	-	--	--	--	--	--	--	--
Product Road Map(Suggested	-	--	--	--	--	-	--	--	--	--	--	--	--
Non-finctional req (Suggested	-	--	--	--	--	-	--	--	--	--	--	--	--
Support tickets (Suggested)	-	--	--	--	--	-	--	--	--	--	--	--	--

The next step is to combine the results in Table 4.5 on the basis of three key role categories—Product Owner, Scrum Master, and Development Team—in Scrum (see Table 4.1). Both Product Owner and Scrum Master Categories contain two roles (hence two cell entries in Table 4.5) while the category Development Team contains eight roles (hence eight cell entries in Table 4.5). The two cell entries (each cell correspond to a particular role) on the degree of usefulness for each category, Product Owner and Scrum Master (Table 4.3), were combined into one entry using the "AND" operation where two inputs get resolved into one output (Batten, 1973). The operation works similar to "ADD" operation, which takes two values and produces one result (Janikow & Michalewicz, 1991). In this technique, precedence is given to the value which has high value (e.g. Very Useful over Useful). Table 4.6 depicts the criteria used for combining two cell entries in Table 4.5 into one output (entry). As regards the category Development team (Table 4.1), since there are eight cell entries corresponding to eight roles, the criteria (Table 4.6) were used in pairs to combine into a single resultant output (i.e. 8 cell entries → 4 cell entries → 2 cell entries → a single cell entry).

Table 4.6: Cell Entry Combining Criteria

Cell 1 Entry	Cell 2 Entry	Combined Output
Useful	Useful	Useful
Useful	Very Useful	Very Useful
Useful	Not Useful	Useful
Very Useful	Not Useful	Very Useful

Table 4.7 depicts the results after combining the information in Table 4.5 using the combination criteria described above. Accordingly, Table 4.7 depicts the degree of usefulness of each document to each key role category in Scrum.

Table 4.7: The Degree of Usefulness of each Document for the Key Scrum Roles

Document	**Degree of Usefulness for each Role**		
	Scrum Master	**Product Owner**	**Development Team**
1. Maintained product backlog	**	**	
2. International Specifications	**	**	*
3. Standards & Regulations	**	**	*
4. Stakeholder/Customer	**	**	**
5. Business Flows	**	**	*
6. Road Map Items (RMI)	*	**	*
7. Features	**	**	*
8. Test Verification Report	*	*	**
9. Sprint demonstration recordings	*	*	*
10. Retrospective notes	**	**	
11. Training materials			
12. Risk Analysis reports	**	**	
13. Technical Design	*	*	**
14. DB designs			**
15. Developer documentation			**
16. Traceability Matrices	*	*	*
17. Unit Test and test reports			**
18. Source codes files			**
19. Bug tickets	*	*	**
20. Test plans	*	*	**
21. Bug Verification report	*		**
22. User manuals			**
23. Companywide quality procedures	*	*	**
Notes: ***** Indicates a **Useful** document based on the votes received ****** Indicates a **Very useful** document based on the votes received No asterisk (*) indicates being **not Useful**, based on the votes received			

4.2.2.2 Obtaining Results for Task 2: Identifying the Importance of Each Document

In task 2 of round 2, through the questionnaire (Appendix B), each respondent was presented the list of 23 documents identified from round 1of data collection (with option for the respondent to add more documents to the list) and was asked to rate each document for each role using a 5 point rating system: 1 being extremely important, 2 being important, 3 being neither important nor unimportant, 4 being not important, and 5 being extremely unimportant; a numbering (ranking) system is used to judge the importance of documents because unlike usefulness, importance is a *value judgement* (see section 4.2 for the definitions of usefulness and importance). As a sample, Table 4.8 shows the votes received for 1s (extremely important) through to 5s (extremely unimportant) for four documents (there are 23 documents altogether) and three roles (there are 13 roles altogether). Based on these votes (values for 1s through to 5s) the weighted average score was calculated for each document for each role. These weighted averages were then and rounded-off to the nearest integers to determine the degree of importance of each document for each role. As an example, the weighted average value (rounded-off to the nearest integer) for the document 'Product Backlog' for the role 'Customer' is 3, which translates to 'neither important nor unimportant', based on the scale used. The following sample calculation.

$$\begin{aligned} Weighted..Average_{\Pr oductBacklog,Customer} &= [(1*5)+(2*4)+(3*7)+(4*1)+(5*2)]/19 \\ &= 48/19 = 2.52 = 3\,(\text{Rounded}) \end{aligned}$$

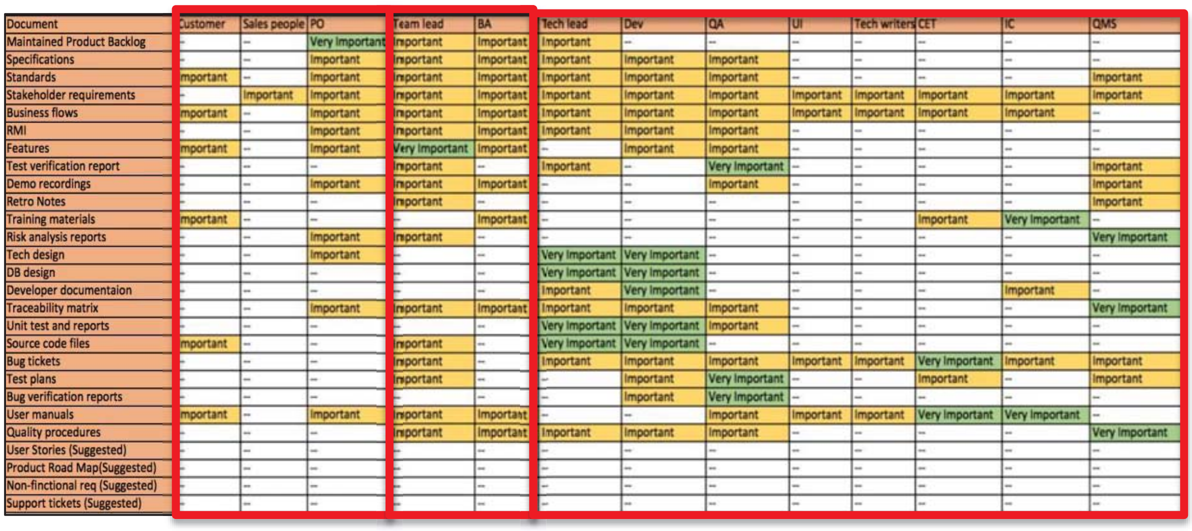
Table 4.8: Sample Scores Received for Each Document

Document	**Customer**	**Product Owner**
Maintained Product backlog	1=5; 2=4 ;3=7 ;4=1,5=2	1=17; 2=0 ;3=0 ;4=0,5=2
Specifications	1=4; 2=6 ;3=6 ;4=1,5=2	1=6; 2=11 ;3=1 ;4=0,5=1
Standards	1=5; 2=8 ;3=2 ;4=1,5=3	1=8; 2=7 ;3=2 ;4=0,5=2
Stakeholder requirements	1=14; 2=2 ;3=1 ;4=5,5=2	1=17; 2=0 ;3=1 ;4=0,5=1
Business flows	1=8; 2=6 ;3=1 ;4=2,5=2	1=10; 2=7 ;3=0 ;4=0,5=2

In this study, the focus is on identifying important documents (rounded-off weighted average score of 4) and very important documents (rounded-off weighted average score of 4). Therefore, any rounded off weighted average score of 3, 2 or 1 rating of importance for a particular document for as particular role was treated as an unimportant document.

Table 4.9, which is a screen dump from the researcher's spreadsheet, depicts the degree of importance of each document for each role. The reader should note that the blank cells (dash) in Table 4.9 denote documents that are *unimportant* to a particular role.

Table 4.9: The Degree of Importance of Each Document for Each Role



	Product Owner			Scrum Master		←---------------------------Development Team ----------------------------→							
Document	Customer	Sales people	PO	Team lead	BA	Tech lead	Dev	QA	UI	Tech writers	CET	IC	QMS
Maintained Product Backlog	–	–	Very Important	Important	Important	Important	–	–	–	–	–	–	–
Specifications	–	–	Important	Important	Important	Important	Important	Important	–	–	–	–	–
Standards	mportant	–	Important	Important	Important	Important	Important	Important	–	–	–	–	Important
Stakeholder requirements	–	Important	Important	Important	Important	Important	Important	Important	Important	Important	Important	Important	Important
Business flows	mportant	–	Important	Important	Important	Important	Important	Important	Important	Important	Important	Important	–
RMI	–	–	Important	Important	Important	Important	Important	Important	–	–	–	–	–
Features	mportant	–	Important	Very Important	Important	–	Important	Important	–	–	–	–	–
Test verification report	–	–	–	Important	–	Important	–	Very Important	–	–	–	–	Important
Demo recordings	–	–	Important	Important	Important	–	–	Important	–	–	–	–	Important
Retro Notes	–	–	–	Important	–	–	–	–	–	–	–	–	Important
Training materials	mportant	–	–	–	Important	–	–	–	–	–	Important	Very Important	–
Risk analysis reports	–	–	Important	Important	–	–	–	–	–	–	–	–	Very Important
Tech design	–	–	Important	–	–	Very Important	Very Important	–	–	–	–	–	–
DB design	–	–	–	–	–	Very Important	Very Important	–	–	–	–	–	–
Developer documentaion	–	–	–	–	–	Important	Very Important	–	–	–	–	Important	–
Traceability matrix	–	–	Important	Important	Important	Important	Important	Important	–	–	–	–	Very Important
Unit test and reports	–	–	–	–	–	Very Important	Very Important	Important	–	–	–	–	–
Source code files	mportant	–	–	Important	–	Very Important	Very Important	–	–	–	–	–	–
Bug tickets	–	–	–	Important	–	Important	Important	Important	Important	Important	Very Important	Important	Important
Test plans	–	–	–	Important	–	–	Important	Very Important	–	–	Important	–	Important
Bug verification reports	–	–	–	–	–	–	Important	Very Important	–	–	–	–	–
User manuals	mportant	–	Important	Important	Important	–	–	Important	Important	Important	Very Important	Very Important	–
Quality procedures	–	–	–	Important	Important	Important	Important	Important	–	–	–	–	Very Important
User Stories (Suggested)	–	–	–	–	–	–	–	–	–	–	–	–	–
Product Road Map(Suggested)	–	–	–	–	–	–	–	–	–	–	–	–	–
Non-finctional req (Suggested)	–	–	–	–	–	–	–	–	–	–	–	–	–
Support tickets (Suggested)	–	–	–	–	–	–	–	–	–	–	–	–	–

Table 4.10 depicts the degree of importance of each document to each role category (i.e. Scrum Master, Product Owner, and Development Team) after combining the information in Table 4.9 using the combination criteria that was used to combine similar information on the degree of usefulness of documents. Accordingly, Table 4.10 depicts the degree of importance of each document to each key role category.

Table 4.10: Degree of Importance of each Document for Each Key Scrum Roles

Document	Degree of Importance for Each Role		
	Scrum Master	Product Owner	Development Team
1. Maintained product backlog	*	*	*
2. International Specifications	*	*	*
3. Standards & Regulations	*	*	*
4. Stakeholder/Customer requirements	*	*	*
5. Business Flows	*	*	*
6. Road Map Items (RMI)	*	*	*
7. Features	**	**	*
8. Test Verification Report	*	*	**
9. Sprint demonstration recordings	*	*	**
10. Retrospective notes	*	*	*
11. Training materials	*	*	
12. Risk Analysis reports	*	*	
13. Technical Design			**
14. DB designs			**
15. Developer documentation			**
16. Traceability Matrices	*	*	**
17. Unit Test and test reports			**
18. Source codes files	*	*	**
19. Bug tickets	*	*	**
20. Test plans	*	*	**
21. Bug Verification report			**
22. User manuals	*	*	**
23. Companywide quality procedures	*	*	**

Notes: * Indicates Important document based on the votes received
** Indicates Very Important based on the votes received
No asterisk (*) indicates being Not Important, based on the votes received

4.2.3 Conclusion for RQ1

As mentioned earlier, in order to answer RQ1, the results of round 1 and round 2 have to be consolidated. After analyzing and presenting useful and important operational-level documents for each Scrum role separately, the final result is displayed in Table 4.11. This table shows that all 23 documents are important in Scrum. While some documents are useful and important to more than one role category (e.g. International Specifications), some are either important or useful top only one role category. This information can be used to prioritize documents and/or to tailor the documents to the requirements of the relevant role category. Finally, the results in Table 4.11 will be used in round 3, to answer RQ2, by way of identifying minimal content and sections needed in the documents.

Table 4.11: Usefulness and Importance of Each Document for Each Key Scrum Roles

DOCUMENT	Role Category					
	Scrum Master		Product Owner		Development Team	
	Useful	Important	Useful	Important	Useful	Important
1. Maintained product backlog	**	*	**	*		*
2. International Specifications	**	*	**	*	*	*
3. Standards & Regulations	**	*	**	*	*	*
4. Stakeholder/Customer requirements	**	*	**	*	**	*
5. Business Flows	**	*	**	*	*	*
6. Road Map Items (RMI)	*	*	**	*	*	*
7. Features	**	**	**	**	*	*
8. Test Verification Report	*	*	*	*	**	**
9. Sprint demonstration recordings	*	*	*	*	*	**
10. Retrospective notes	**	*	**	*		*
11. Training materials		*		*		
12. Risk Analysis reports	**	*	**	*		
13. Technical Design	*		*		**	**
14. DB designs					**	**
15. Developer documentation					**	**
16. Traceability Matrices	*	*	*	*	*	**
17. Unit Test and test reports					**	**
18. Source codes files		*		*	**	**
19. Bug tickets	*	*	*	*	**	**
20. Test plans	*	*	*	*	**	**
21. Bug Verification report	*				**	**
22. User manuals		*		*	**	**
23. Companywide quality procedures	*	*	*	*	**	**

Notes: * Indicates either being **Useful** or **Important,** based on your votes received
Indicates either being **Very useful or **Very Important**, based on your votes received
No asterisk (*) indicates being not Useful or Important, based on your votes received

4.3 Results for the Research Question 2 (RQ2)

The second research question asks *"how can the documents be prepared in a timely and concise manner to maintain good communication among different parties in the development lifecycle?"* The conciseness aspect of documentation relates to identifying which document is important to whom (readily covered in RQ1), and what content should be included in each document. The timeliness aspect of documentation relates to identifying exactly when each document should be prepared to maintain good cross-functional communication. These are what that are being addressed in this section. As mentioned earlier, the second research question (RQ2) is answered through the results of round 3 data collection. The following sections cover the results of round 3 data collection and the corresponding discussion.

4.3.1 Round 3 of the Study Relevant to RQ2

As defined in the methodology section, round 3 of data collection was designed to address RQ2 and RQ3. The first task in this round was identifying the minimal content to be presented in a document (in the previous rounds, 23 documents were identified as being useful and important in Agile) so as to look adequate, in terms of depth of details in the documentation to meet the ISO 9001:2008 international QMS standard. The second task in this round was to find the best stage of a Scrum project to produce each document. Completion of these two tasks addresses RQ2.

4.3.1.1 Analysis of Task1 Results

In this task, the respondents were asked select the relevant sections (the purpose section, the author section and so forth) and content (user stories, flowcharts/diagrams and so forth) for each of the three key role categories (i.e. the Development Team, the Scrum Master, and the Product Owner), from a list provided to them (see Appendix C). As mentioned in the previous chapter, the nine-member panel utilized in round 3 consisted of three Development Team member roles, three Scrum Master Roles, and three Product Owner roles. The results received are shown in Tables 4.13 through to 4.16. Participants of Round 3 can break-down in to their roles as follow.

Table 4.12: Round 3 Participants Breakdown by their Role

Product Owner	Scrum Master	Development Team
Round 3		
Product Owner *(n = 3)*	Team Lead *(n = 1)* Business Analyst (n=2)	Developer *(n = 1)* Test Engineer *(n = 1)* Member of the Quality Assurance team *(n = 1)*

4.3.1.1.1 Identify the Sections and Content Relevant to the Development Team

Table 4.13 depicts the sections and content being viewed as important by the three Development Team members, for their role, given each document. The documents shown in the table are the documents, which are marked as Important/Very Important and Useful/Very Useful in Table 4.11 (usefulness and importance of each document for Scrum roles) for the Development Team.

The results shown in Table 4.13 are based on the choices (votes) made by individual team members in the Round 3 questionnaire. The results were finalized using following decision criterion. If no member or only 1 member (out of 3) agreed that a particular section or content is relevant to their role, such a section or content was considered as not relevant. If only 2 members agreed that a particular section or content is relevant to their role, such a section or content was considered as relevant based on majority vote. This approach was found to be more practical than arranging follow-up meetings to reach a consensus decision, which is the standard approach used in Delphi-like decision making environments.

Table 4.13: The Sections and Content Being Viewed as Relevant by the Development Team

Document	Sections							Content					
	1. Purpose	2. Author	3. Approvers/Reviewers/Sign off	4. Internal communication records	5. Corrective action records	6. Preventive action records	7. Comments on validity (out-of-date/relevant)	a. User stories	b. Flow charts/diagrams	c. Development instructions	d. Rerelease schedules/notes	e. Known issues	f. Verification and validation information
International Specifications	X	X	X	X	X	X	X	X	X	X	X	X	X
Standards & Regulations	X	X	X	X			X		X	X			X
Stakeholder requirements	X	X	X	X		X	X	X	X		X		
Business flows	X		X	X				X	X				
RMI	X	X	X	X		X	X	X	X		X		
Features	X	X	X	X	X	X	X	X	X	X			X
Test verification report	X	X	X	X	X	X	X	X		X	X	X	X
Demo recordings	X		X	X	X			X					
Training materials	X		X				X				X		
Tech design	X	X	X	X		X	X	X	X	X		X	X
DB design	X	X	X	X		X	X		X	X	X	X	X
Developer documentation	X	X	X		X	X	X			X	X	X	X
Traceability matrix					X						X		
Unit test and reports	X	X	X	X	X	X	X	X		X	X	X	X
Source code files	X	X	X		X	X	X	X		X		X	X
Bug tickets	X	X	X	X	X	X	X	X		X	X	X	X
Test plans	X		X							X		X	X
Bug verification reports	X	X	X	X	X		X	X		X	X	X	X
User manuals	X	X					X						
Quality procedures	X	X							X	X			X
Maintained Product backlog	X	X	X	X			X	X			X	X	X
Retro Notes	X			X	X				X	X			
#of Votes (22)	2	1	1	1	1	1	1	1	1	1	1	1	1

Notes:

'X' indicate relevance; blanks indicate non-relevance

The results are based on the responses given by the three Development Team members

4.3.1.1.2 Identify the Sections and Content Relevant to Scrum Master

Table 4.14 depicts the sections and content being viewed as important by the 3 Scrum Masters, for their role, given each document. The documents shown in table are the documents, which are marked as Important/Very Important and Useful/Very Useful in Table 4.11 (the usefulness and importance of each document for Scrum Master category). The results shown in table are based on the choices (votes) made by individual members in Round 3 questionnaire, using the criterion of vote counting as used previously.

Table 4.14: The Sections and Content Being Viewed as Relevant by the Scrum Master

Document	Sections							Content					
	1. Purpose	2. Author	3. Approvers/Reviewers/Sign off	4. Internal communication records	5. Corrective action records	6. Preventive action records	7. Comments on validity (out-of-date / relevant)	a. User stories	b. Flow charts/diagrams	c. Development instructions	d. Rerelease schedules/notes	e. Known issues	f. Verification and validation information
Maintained Product Backlog		X	X	X			X	X	X		X	X	
International Specifications	X	X	X	X			X	X	X			X	X
Standards & Regulations	X	X	X				X					X	X
Stakeholder requirements	X	X	X	X			X	X	X				X
Business flows	X	X	X										
RMI	X	X	X				X	X	X			X	X
Features	X	X	X				X	X	X			X	X
Test verification report	X	X	X	X	X	X	X					X	X
Demo recordings	X	X	X				X	X					X
Retro Notes	X	X			X	X							
Risk analysis reports	X	X	X	X		X	X						X
Bug tickets	X	X	X										X
Test plans	X	X	X										
Traceability matrix	X	X	X										X

Document	Sections							Content					
	1. Purpose	2. Author	3. Approvers/Reviewers/Sign off	4. Internal communication records	5. Corrective action records	6. Preventive action records	7. Comments on validity (out-of-date / relevant)	a. User stories	b. Flow charts/diagrams	c. Development instructions	d. Rerelease schedules/notes	e. Known issues	f. Verification and validation information
Quality Procedure	X	X	X			X							X
User manuals	X	X	X				X						
Training Materials	X	X	X										
Source code files	X	X	X										X
Tech Design	X	X	X	X			X			X			X
Bug verification reports	X	X	X	X			X					X	X
# of Votes (20)	19	20	19	7	2	4	12	6	5	1	1	7	14

Notes:

‘X’ indicate relevance; blanks indicate non-relevance

The table has been based on the responses given by the three Scrum Masters

4.3.1.1.3 Identify the Sections and Content Relevant to Product Owner

Table 4.15 depicts the sections and content being viewed as important by the 3 Product Owners, for their role, given each document. The documents shown in the table are the documents, which are marked as Important/Very Important and Useful/Very Useful in Table 4.11 (usefulness and importance of each document for Scrum roles for the Product Owner category). The results shown in table are based on the choices (votes) made by individual members in Round 3 questionnaire, using the criterion of vote counting as used previously.

Table 4.15: The Sections and Content Being Viewed as Relevant by the Product Owner

Document	Sections							Content					
	1. Purpose	2. Author	3. Approvers/Reviewers/Sign off	4. Internal communication records	5. Corrective action records	6. Preventive action records	7. Comments on validity (out-of-dat	a. User stories	b. Flow charts/diagrams	c. Development instructions	d. Rerelease schedules/notes	e. Known issues	f. Verification and validation inform
Maintained Product Backlog	X	X	X	X			X	X					X
International Specifications	X	X	X				X	X	X				X
Standards & Regulations	X	X					X						X
Stakeholder requirements	X	X	X	X			X	X	X				X
Business flows									X				
RMI	X	X	X	X			X	X			X		X
Features	X	X	X	X			X	X	X				X
Demo recordings	X						X	X					
Risk analysis reports	X	X	X	X	X	X	X		X		X	X	X
User manuals	X		X				X		X				
Tech design	X	X	X	X					X				
Traceability matrix	X	X	X						X				X
Training Materials	X		X						X				
Source code files	X	X							X				
#of Votes (14)	13	10	10	6	1	1	8	6	10	0	2	1	8

Notes:

'X' indicate relevance; blanks indicate non-relevance

The table has been based on the responses given by the three Product Owners

4.3.1.1.4 Summary of Task 1

By reviewing the results shown in the above three tables, in regards to the sections- in general, for all listed documents, *'purpose', 'author', 'approvers / reviewers / sign-off', and 'comments on validity'* seems to be important and relevant to all 3 roles. However, *'internal communication records' and 'preventive action records'* seem to be more important and relevant to the Development Teams rather than the Scrum Masters and the Product Owners.

In regards to the document content in general, for all listed documents, the *'verification and validation information'* seem to be important and relevant to all 3 roles. However, again *'flow charts/diagrams'* seem to be important and relevant to Development Teams and the Product Owner rather than the Scrum Master, while the *'known issues'* seem to be mostly relevant to the Development Team only.

4.3.1.2 Analysis of Task2 Results

In task 2, the objective is to identify the most appropriate stage to produce documents. Based on the responses received from the 9 participants, following stages are identified. When finalizing the stages, majority vote was considered because of difficulty in arraigning meetings to reach consensus decisions.

Table 4.16 shows the most appropriate stage identified in a Scrum project to produce the document.

Table 4.16: Best Stage to Produce the Documents

Document	**Most appropriate stage to produce the document**
Maintained Product backlog	Project initiation stage Before sprint
Standards & Regulations	Project initiation stage Before sprint
Business flows	Before sprint
RMI	Before sprint
Features	Before sprint
Demo recordings	At the end of a sprint

Document	Most appropriate stage to produce the document
User manuals	Within a sprint
Tech design	Starting of a sprint
Source code files	Within a sprint
Training materials	At the end of a release
International Specifications	Project initiation stage Before sprint Within a sprint At the end of a sprint
Stakeholder requirements	Project initiation stage Before sprint
Traceability matrix	Before sprint Starting of a sprint Within a sprint At the end of a release
Retro Notes	At the end of a sprint
Bug tickets	Within a sprint
Test plans	Before sprint Starting of a sprint Within a sprint
Test verification report	Within a sprint At the end of a sprint
Quality Procedure	Project initiation stage Before sprint
Bug verification reports	At the end of a sprint At the end of a release
Risk analysis reports	Before sprint At the end of a sprint At the end of a release After the release
Unit test and report	Within a sprint At the end of a sprint

Document	**Most appropriate stage to produce the document**
DB design	Before sprint Starting of a sprint Within a sprint
Developer documentation	Within a sprint

Project initiation stage is the stage in which the software development team finalizes the development work. In order to do the work, information in Product Backlog, Standards & Regulations, International Specifications, Stakeholder Requirements and Quality Procedures are helpful. It is seen from the above table that those documents should be therefore created at the project initiation stage.

Once the project is finalized, the Development Team finalizes their scope of work and number of sprints required for the development. In order to help doing that documents such as the Product Backlog, Standards & Regulations, International Specifications, Stakeholder Requirements and Quality Procedure, Business Flows, RMI, Features, Test Plans, DB Design and Risk Analysis Reports should be ready before starting a sprint or at the time of the starting a sprint.

Within the sprint, with the development effort, teams can create the User Manual, Source Code Files, Bug Tickets, Test Verification Report, Unit Tests, and Developer Documentation. At the end of each sprint, development teams can create Demo Recordings, Retro Notes and complete work with their Test Verification Notes, Unit Tests and Risk Analysis Reports.

After multiple iterations, when the product is completely ready for production, the development teams can create Training Materials, Traceability Matrices and Risk Notes. After the release, depending on the product, the development team can still maintain Risk Analysis Reports. However, it is important to note that Bug Tickets can be created by any member who is using the software at any stage of the development, as it concerns product quality.

4.3.2 Conclusion for RQ2

As per the results shown is tables 4.13, 4.14, and 4.15, the most relevant section for the Development Team (DT) is the 'Purpose' of the document, followed by 'Approver/Reviewer sign-off'. It seems that 'Corrective and Preventive' action records are the least relevant sections for the development teams in the documents. In terms of the contents, the DT seems to be keen to see 'Development Instructions' and 'Verification and Validation Instructions’ on the documents that they refer to. Least relevant content for the DT seems to be the 'Flow Charts /Diagrams' (Table 4.13).

By looking at the results for Scrum Master (SM) category, it seems that the 'Author' of the document is the important section to them, followed by 'Purpose' and the 'Approver/Reviewer Sign-off' of the document. 'Corrective Action Records' seem to be the least relevant section for the SM. In-terms of the content, 'Verification and Validation Information’ are the most important sections while 'Development Instructions' and 'Release Schedules' are the least important sections for the SM.

For the Product Owner (PO), 'Purpose' of the document seems to be the most relevant section, followed by the 'Author' and the 'Approver/ Reviewer Sign-off', while the least relevant section seems to be the 'Corrective and Preventive Action Records'. As regards the content, the POs seem to be interested in seeing 'Flow-Chats and Diagrams' in the documents, followed by 'Verification and Validation' Information, while the 'Developer Instructions' seem to be the section that they are least concerned with.

Also, by looking at Table 4.16, it becomes evident that most of the documents including Developer Documentation, DB design, Unit Test and Report, and the Test Verification Report can be created within the sprints. Documents that should be ready before the sprint starts (e.g. the Quality Procedure, Test Plans, and Stakeholder Requirements) can either be created at the project initiation stage, before sprint starts, or at the start of the sprint itself. However, some documents (e.g. Demo Recordings and Retro Notes) can only be created at the end of a sprint.

Producing the artifacts mentioned above shall be created as part of the project activities, not keeping it to the last minute. Keeping the documentation tasks separate to the project activities can result in more effort as mentioned in section 2.4. Hence producing the documentation at the most appropriate phase can minimize additional effort and maximum benefit to its audience.

4.4 Results for Research Question 3 (RQ3)

The third research question asks *"what are the issues that cause documentation processes to be less effective, in a typical Scrum environment?* For ease of explanation, as mentioned earlier, the researcher has divided documentation effectiveness issues into two categories: technical and practical. The aim of task 3 (in round 3 of data collection) is to uncover the technical issues concerning documentation effectiveness, while the aim of task 4 is to uncover the practical issues concerning documentation effectiveness.

4.4.1 Round 3 of the Study Relevant to RQ3

4.4.1.1 Analysis of Task 3 Results

In task 3, as mentioned earlier, the nine respondents were asked (through a mini 30-minute interview with each respondent) to recollect negative experiences they have had with documentation (see Appendix-C). In other words, the respondents were asked to describe instances in which the documentation was found to be ineffective due to technical reasons. The responses provided by the respondents (*n* = 9) were synthesized and classified under seven themes. Each theme is described in turn. The parenthesized codes shown below are the respondent codes—for example, DT1 is the first respondent (in the alphabetical order of the family name) representing the Development Team category.

1. **Non- technical customers find it harder to understand the documents**

a. The documents which are presented to customers such as User Manuals and Admin Manuals are often written from a developer's perspective. This makes it harder for the non-technical customers to understand and interpret the instructions. As a result of this, customer support engineers get very busy in explaining the contents in the documents. The effectiveness of documents presented to the customers can be improved considerably [DT1] [SM2] [PO1].

b. When analyzing the bugs raised by the customers, it becomes clear that the bugs could have been easily prevented, if the Development Team has provided correct and comprehensive documents [DT2] [DT1] [PO1] [PO3].

2. Sometimes, the purpose of a document is not clearly stated

a. Sometimes, the purpose of a document is not clearly stated in the document itself; this increases the support requests received by customers [PO1] [SM1] [SM3] [DT3]

SM1 said:

"If we fix a bug, we state that the identified bugs are fixed for the release. But we haven't bothered to mention the information in the release notes." Missing information often related to questions such as: (i) "what is the bug?" (ii) "Why is it fixed?" (iii) "Where was it fixed?" (iv) "Which part of the functionality is affected?", and (v) "who are the potential customers who need an upgrade?"

Also, for internal documents, not having stated the purpose of the document makes it harder to capture *why the document has been created* [SM2] [PO2].

b. The documents should always provide the relevant context. For example, a document should provide an explanation as to why the document is needed. Failing to provide the necessary context of a document may mislead the users [DT2] [PO3].

3. Not following up with client feedback

The development team is often unaware about the documents that are important to the customers [DT3] [PO2] [PO1] [DT1] [PO2] [SM3].

DT 3 said:

"If we do a proper analysis and follow up with customer feedback, we could have offered them better documents than we are offering them now".

Two respondents mentioned that often, users are logging clarification support requests on some of the functionalities of the system. They mentioned that even though it seems very fundamental, when they look at the support requests logged by the customers, it becomes evident that the Development Teams have to provide complete and easily understandable documents to the customer [DT1] [SM3].

4. **Increased amount of re-work due to requirement changes**

a. Due to poorly understood documented requirements and business cases, the development teams have to re-work on the code and the test cases, after the customer seeks certain clarifications of the product [PO1] [PO2] [SM3] [DT1].

b. Excessive rework is an issue and this can be partially mitigated if the “purpose section” and the reviewer’s/approvers sections of a document is done well; this will help to minimize the re-work of pre-built product functionalities. When doing development, it is hard to make sure who defined the requirements in some cases as the author/purposes is not defined at the right time; this is specially the case when requirements are defined earlier by a different team and given to another team for development [PO2] [SM3] [DT1].

c. Stakeholder requirements need to be clearly specified and signed-off. When this does not happen, the product requirement is considered to have not been validated by the stakeholder, before the commencement of development [PO3] [SM2] [DT3].

5. Not mentioning the validity of the documents increase the number of less useful documents pertaining to the product

a. Not specifying the purpose of a document is a common case; this makes harder for the development teams to define and keep track of the version the document(s) and its validity at a particular point in time [PO3] [SM1] [DT3].

b. Not having authorship and internal communications committed within the team has resulted in misguided vision and motivation for the team to deliver work [PO3] [SM1].

c. Having too many documents in an organization always require proper reasons being provided to validate the existence of a document [PO1] [SM2] [DT2].

6. Lack of risk analysis reports

Lack of risk analysis reports results in a release having risk issues and release restrictions with two consequent releases [PO2] [PO1] [SM3].

7. Disruptions in the development work

a. Having no maintained product backlog means that the team(s) start on a certain body of work have to abandon the work mid-cycle to work on another body of work. This causes disruption in the flow and upsets the team morale [DT3] [SM1] [DT2].

b. Having no properly defined sprint backlog increases the amount of re-design, development and testing of the product. This will affect negatively on the teams' velocity towards succeeding sprints [DT3] [SM2] [DT3].

4.4.1.2 Analysis of Task 4 Results

In task 4, as mentioned earlier, the nine respondents were asked (through a mini 30-minute interview with each respondent) to recollect issues that they might have had in creating important documentation due to non-technical/practical problems (a definition of non-technical/practical problems was given, as evidenced in Appendix-D). The respondents (n = 9) were synthesized and classified under seven themes. Each theme is described in turn.

1. Lack of skilled document writers

a. Due to the existence of cross-functional teams, most teams do not have a dedicated technical writer to prepare documents tailored to the client. There have been many instances in which the teams had to work without skilled and qualified document writers; this hinders the production useful and valuable documents [PO3] [SM1] [DT3].

b. Not having a dedicated person to write documents often leads to having no or few (less than optimal) documents being produced; this is due to all resource personnel of the project being busy on their assigned tasks (no one has enough time to maintain proper documentation) [DT1] [DT2] [SM3].

c. The ‘release notes’ are often not written in a way that is useful or meaningful to the client. These documents do not really explain how an issue might affect the client or how the issue could be fixed. According to the respondents (see below) this might be due to lack of regulations or passion, on the part of the writers [PO1] [DT2] [SM3].

d. Those who have a passion to write documents do not know the technical/ implementation detail and the people who know such details do not like or do not know how to write documents in a readable manner [DT1] [DT2] [SM3] [PO2].

e. Often, people are not interested in articulating how a problem was solved. For the most part, nontechnical people do not see the bigger picture; therefore, it becomes harder to explain the real content [DT2] [SM2] [PO2].

2. Keep documentation till the last minute

a. Leaving till the last minute to create the documents results in having minimal content in the documents (rush work). This often leads to omission of important information, which could have been captured parallel to development [PO1] [SM3] [PO2].

b. The feature document is something that is left till the last minute and time constraints mean that it ends up being brief (or not being very useful); this means rework, as the document needs to be re-written later [DT2] [SM1] [SM2] [PO2].

3. Incorrect understanding of creating Agile concepts

a. There is a common misconception among some Scrum practitioners that Agile means no documentation! [SM1] [SM2] [PO1].

b. When people think that Agile projects should not involve documentation, it becomes hard to convince them to write proper and useful documents [SM1] [SM2] [PO1].

4. **No common documentation standard is maintained**

a. Quite a number of documents being produced by one team becomes ambiguous to another team; there is little common language/framework being used [SM1] [SM2] [PO1] [DT1].

b. Having a companywide strategy to write/produce the documents makes it easier for teams to refer documents [SM1] [SM2] [PO1] [DT1] [DT2].

5. **Producing documents is considered to be a burden**

a. Cultivating passion and motivation to create documents is the hardest challenge. It is commonly labelled as tedious and uninteresting, yet it has a significant place in delivery. Most teams find it tedious to document what they have done/considered during development [SM2] [PO3] [DT3].

b. Documentation is not considered a deliverable to some stakeholders unfortunately, hence it becomes difficult to justify time and effort spent on these activities [SM1] [PO2] [DT3].

c. Due to strict deadlines/timelines and shortage of time, people cut down the focus on documents; this results in problems later on. Also, it becomes even harder to do it later, as it is more time consuming [SM3] [PO3] [DT3].

d. Sometimes the deadline/resources or the budget can become a hurdle to adhere to the standards [SM2] [PO2] [DT1].

6. **Poor tooling for documentation**

Some tools are either not fit for purpose or the development team has yet to optimize or automate [DT1] [DT3].

7. **Not having the right people at the right time**

Not having the right people around at the right time, makes production of documents harder. Often, the product owner is not at hand to provide sufficient details and/or the architects aren’t available to guide the team in the right direction [DT2] [DT3] [SM2].

4.4.2 Utilizing Secondary Data

As mentioned in section 3.5.4.4 (Secondary Source Information) in the previous chapter, complaints received on issues related to the 'User Manual' with in past 2 years were recorded and analyzed to identify underlying issues. The objective of the secondary analysis was to strengthen the findings based on tasks 3 and 4 in Round 3 (primary) data collection. In other words, the objective of secondary analysis was to effect triangulation of evidence.

Table 4.17 provides an overview of the issues raised by the users, the main problem, and the proposed resolution for each issue. The researcher observes that for the period between 1st January 2014 through to 1st October 2015, 15 bugs have been raised concerning the 'User Manuals', for the applications that were live in the production.

The reader should note that some information pertaining to the bugs have omitted due the information security policy of the case study organization.

Table 4.17: Overview of the Bugs Raised Against User-Manuals

ID	**Stage**	**Problem Overview**	**Diagnosis**	**Resolution**
#1	Non-production (UAT level)[1]	Invalid information regarding the supported operating systems and DB versions.	It is identified in the user manual, incorrect version of Windows operating system is mentioned as supportive, but it is actually not supportive to successfully run the application, Hence the DB version mentioned also not supportive.	Documentation is correctly updated. No change in the software was required.
#2	Production	Hyperlinks in the documentation	It is found that in one place of the documentation,	Documentation is correctly updated with working

ID	Stage	Problem Overview	Diagnosis	Resolution
		cannot be viewed. Therefore, information cannot be found.	hyperlink is broken, therefore users cannot proceed with their functions.	hyper-links. No change in the software was required
#3	Non-production (UAT[1] level)	Supported browser information is incorrect.	The application was supposed to be working fine in IE (Internet Explorer) 11 browser. However, the user manual has no information related to this and it is considered as misleading to the users.	Documentation is correctly updated with supported browser information. No change in the software was required
#4	Non-production (UAT[1] level)	Behavior of an application function is different from the documented behavior.	One of the functions of an application is in-correctly documented. This misleads the users as documented behavior is different from actual behavior. This is then identified as a bug in the system as the behavior shown is incorrect.	Since it is found that the actual behavior is incorrect, product change was required to the application.
#5	Production	After upgrading to a new version, there is	It is found that after the software is upgraded to its newer	Documentation is updated with additional

ID	Stage	Problem Overview	Diagnosis	Resolution
		an issue and the documentation does not have instructions to fix it.	version, issue arise with connecting to a required connector. Since documentation does not say what additional configuration is required to fix this user can't proceed.	instructions and inform all the client about the new change. No change in the software was required
#6	Non-production (UAT[1] level)	Dependencies of the software applications are incorrectly documented.	It was identified that incorrect dependencies of an application are documented. (E.g. to work Software A, it requires 0.2 version of software B and 0.4 version of Software C. But it is documented as it requires, 0.1 version of Software B)	Documentation is updated with correct dependencies. No change in the software was required.
#7	Non-production (UAT[1] level)	Missing configuration information.	Documentation does not include complete configuration information, which results in erroneous state of the software.	Documentation is updated with completed configuration steps. No change in the software was required.
#8	Production	How to handle empty fields of	It was identified, documentation has a	Documentation is updated with

ID	Stage	Problem Overview	Diagnosis	Resolution
		a network route if not properly instructed in the documentation.	step missed on stating how to handle an empty or no-value field of a network route configuration. As clients are usually depending on the documentation, they don’t want to take the risk and do their own step to handle it.	correct configuration step. No change in the software was required.
#9	Production	Documentation structure is complex thus hard to find information.	One of the clients made a compliant asking to change the structure of the documentation as they felt it complex to navigate through the pages.	No change to documentation structure was made as it is the standard structure of the organization. However, customer support engineer helped the client to be familiarize with the documentation.
#10	Production	Documented behavior of handling date-time is different than its actual behavior.	Found a bug in the system and it was planned to fix it immediately.	Documentation was correct. This required an urgent change in the product.
#11	Production	Incorrect language	Found the language definition used in the	Documentation was correct.

ID	Stage	Problem Overview	Diagnosis	Resolution
		definition is used in an application, compared to the documentation.	actual product is in-correct therefore product change is required.	This required a change in the product.
#12	Production	Missing documentation	Information seems to be missing in an application behavior.	Documentation is updated with missed information. No change in the software was required.
#13	Non-production (UAT[1] level)	Instructions missing.	Instructions in resetting default user name and password of an application is missing from the document.	Documentation is updated with missed information. No change in the software was required.
#14	Non-production (UAT[1] level)	Misleading information on supported function.	It is seen that one of the products following an International standard does not properly state what is supported and what not supported based on the specification. This is misleading the users.	Documentation is updated with correct information. No change in the software was required

ID	Stage	Problem Overview	Diagnosis	Resolution
#15	Non-production (UAT[1] level)	Known issues are not stated in document.	It is identified that some of the known issues the product had when it is releasing are not mentioned in the document.	Updated the known issues section in the documentation. No change in the product was required urgently however the known issues are added in to product backlog for future reference.

[1] UAT (User acceptance testing): Before starting the production, almost all the applications go through a UAT round. It involves Implementation Consultants from the software development company as well as the technical staff from the client organization; both parties sit together and perform installation, configuration and running test cases, which are almost representative of actual scenarios (Hambling & Van Goethem, 2013). In some cases, they replicate the load of the data with a large volume of transactions to test whether the system would fail at any point. This has been a very useful approach in reducing or eliminating issues arising in production, as well as new changes that the software company faces concerning the existing software (Hambling & Van Goethem, 2013)

By referring to the bugs raised against the applications, it can be seen that most of the time, the bugs are raised due to incompleteness (#2, #5, #7, #8, #12, #13, #14, #15) of documentation or incorrect information (#1, #3, #6) provided in the documents. This reinforces one of the issues identified by the respondents, namely, "not following up customer feedback". If the feedback is followed up properly, recurring bugs, that reflect incompleteness and incorrectness on the part of software, could have been prevented. Also, when non-technical clients refer to the documents generated by technical staff, they seem to find some pieces of information missing, arguably due to the fact that technical staff has limited understanding about the context. This was also identified as a quality issue by the respondents (see section 4.4.1.2).

However, the issues mentioned in this section, for the most part, are caused due to the difficulties mentioned by the respondents in section 4.4.1.2. Not having skillful document writers, leaving documentation till the last minute, and still worse, people perceiving document generation as a burdensome activity are always problematic, when producing documents meant to the client. Also, not having the right people at the right time makes it harder to present quality documents with fewer bugs.

If there is a company-wide standard and a common understanding on how documents should be prepared, and if the staff are educated on the importance of documentation, these issues can be minimized.

4.4.3 Conclusion for RQ3

This section of the research study targets on identifying issues and collate negative experiences on documentation. In section 4.4.1.1, multiple factors that affect (cause issues) production of documents were identified. Most issues were found to be due to less attention being given to documentation in Agile. The study found that most issues could be mitigated if Scrum teams pay much attention to it. The two key points based on the results are: (i) not stating a clear purpose of the document (using complex technical terminology could make it harder for readers to understand the document) and (ii) not following customer feedback to improve the documentation (this can simply be attributable to less priority over documentation). Writing documents before identifying requirements was found to be another failure in documentation in Scrum. Incorrect documentation creates a project risk and increased effort to make changes and re-work affects quality.

In section 4.4.1.2, it was shown that having lack of skilled writers and having lack of passion on writing documents are two major limitations when producing the documentation. Having the documentation tasks scheduled up until last minute was found to be another factor that contributes to poor documentation practices in Scrum teams. In the later stages of the project most Scrum teams perceive documentation as a burden. As a consequence, they were found to be producing less documentation with poor quality or no-documentation at all! Also, leaving documentation until last most could cause incorrect or incomplete information as teams become unable to find the right people at that time. Not having proper documentation standards or tooling was found to be another irritant in regards to documentation in Scrum. If the Scrum teams can decide the documentation tasks, estimate and schedule it in their sprints, the problem can be easily mitigated as documentation is one of the major artifacts that should be delivered with software solutions.

Secondary data sourced from customer complaints (section 4.4.2) confirmed the importance of having documentation. Commonly identified causes were having fewer reviews on the content; incomplete and incorrect information; poor style of writing, and complex documents.

Since documentation is a vital artifact in software deliverables, the researcher recommends that it is necessary to have a standard approach and a strategic level solution to address the issues on poor documentation practices in Scrum. Proposed approaches covered in earlier sections (4.2 and 4.3) would be useful to mitigate most of the issues discussed on this section.

4.5 Results of the Validation Round

As mentioned in section 3.7, a final data collection round was carried out to validate the results obtained from the three main rounds (Questionnaire used in the validation round is shown in Appendix D). The objective of this round was to collect data about usage of documents in each project. Results from the 19 respondents were analyzed and these results were used as a threshold to compare or validate the results received from Round1, Round2 and Round3. Details of the participants in the validation round are shown in Table 4.18.

Table 4.18: Validation Round Participants Breakdown by their Role.

Product Owner	**Scrum Master**	**Development Team**
Validation Round		
Product Owner (n =1)	Team lead* (n = 3) Business Analyst (n = 2)	Technical lead* (n = 2) Developer* (n = 3) Tester* (n = 3) Quality Management team member (n = 1) Implementation consultant (n = 1) Technical writer* (n = 2) Customer support engineer (n = 1)
* One member outside the case study organization.		

The votes received for each document, against its usage, were counted and divided by the total number of respondents to determine the proportions (the percentage figures) to apply the blow heuristics. Table 4.19 shows a portion of researcher's spreadsheet containing the votes, as a screen dump. Twelve documents and Votes (Yes or No, if not the reason) are captured in this screen dump.

Table 4.19: Sample of Scores Received for Each Document

			No				
		Yes	a	b	c	d	e
1	Maintained Product Backlog	19	0	0	0	0	0
2	Specifications	12	1	1	0	0	4
3	Standards	16	1	0	0	0	1
4	Stakeholder requirements	14	0	0	0	1	4
5	Business flows	15	0	0	0	1	3
6	RMI	18	0	1	0	0	0
7	Features	19	0	0	0	0	0
8	Test verification report	15	1	0	1	1	0
9	Demo recordings	18	0	0	0	0	1
10	Retro Notes	15	1	1	0	1	1
11	Training materials	14	0	0	0	2	3
12	Risk analysis reports	10	2	1	0	2	5

Heuristic 1: If a particular document receives 75% or more votes for its usage, then, that document is considered to be a *widely used* document in the Scrum team.

• **Heuristic 2:** If a particular document receives 75%-50% votes for its usage, then, that document is considered to be a *used* document in the Scrum team.

• **Heuristic 3:** If a particular document receives less than 50% votes for its usage, then, that document is considered to be a *least used* document in the Scrum team. That documents are accompanied with possible reasons.

Based on the rules, the usage of each document in Scrum Project can be shown as below.

Table 4.20: The Degree of Usage of each Document in Scrum project

Document	Usage	Reason given by the respondent(s) if document is least used
Maintained Product Backlog	Widely Used	
Standards	Widely Used	
Business flows	Widely Used	
RMI	Widely Used	
Features	Widely Used	
Test verification report	Widely Used	
Demo recordings	Widely Used	
Retro Notes	Widely Used	
Tech design	Widely Used	
Source code files	Widely Used	
Bug tickets	Widely Used	
Test plans	Widely Used	
User manuals	Widely Used	
Stakeholder requirements	Widely Used	
Specifications	Used	
Training materials	Used	
Risk analysis reports	Used	
Bug verification reports	Used	
Unit test and reports	Least Used	Other: Reports are not publicised
DB design	Least Used	"I never knew this can be considered as a document"
Developer documentation	Least Used	"I don't see any value using this document"
Traceability matrix	Least Used	"With the tight schedule we can't waste time on writing this"
Quality procedures	Least Used	"Quality team shall look after this"

Table 4.21 juxtaposes the results of the Validation round with the results (usage level) from Round 2.

Table 4.21: The Degree of Usage of each Document in Scrum Projects (Validation Round) Against Usefulness and Importance Identified Earlier (Round 2)

DOCUMENT	Validation Round Results	Round 2 Results					
		Role Category					
	USAGE LEVEL	Scrum Master		Product Owner		Development Team	
		Useful	Important	Useful	Important	Useful	Important
1. Maintained product backlog	(W)	**	*	**	*		*
2. International Specifications	(U)	**	*	**	*	*	*
3. Standards & Regulations	(W)	**	*	**	*	*	*
4. Stakeholder /Customer requirements	(W)	**	*	**	*	**	*
5. Business Flows	(W)	**	*	**	*	*	*
6. Road Map Items (RMI)	(W)	*	*	**	*	*	*
7. Features	(W)	**	**	**	**	*	*
8. Test Verification Report	(W)	*	*	*	*	**	**
9. Sprint demonstration recordings	(W)	*	*	*	*	*	**
10. Retrospective notes	(W)	**	*	**	*		*
11. Training materials	(U)		*		*		
12. Risk Analysis	(U)	**	*	**	*		

DOCUMENT	Validation Round Results	Round 2 Results					
		Role Category					
	USAGE LEVEL	Scrum Master		Product Owner		Development Team	
		Useful	Important	Useful	Important	Useful	Important
reports							
13. Technical Design	(W)	*		*		**	**
14. DB designs	(L)					**	**
15. Developer documentation	(L)					**	**
16. Traceability Matrices	(L)	*	*	*	*	*	**
17. Unit Test and test reports	(L)					**	**
18. Source codes files	(W)		*		*	**	**
19. Bug tickets	(W)	*	*	*	*	**	**
20. Test plans	(W)	*	*	*	*	**	**
21. Bug Verification report	(U)	*				**	**
22. User manuals	(W)		*		*	**	**
23. Companywide quality procedures	(L)	*	*	*	*	**	**

DOCUMENT	Validation Round Results	Round 2 Results					
	USAGE LEVEL	Role Category					
		Scrum Master		Product Owner		Development Team	
		Useful	Important	Useful	Important	Useful	Important

Notes:

* Indicates either being **Useful** or **Important,** based on the votes received

Indicates either being **Very useful or **Very Important**, based on the votes received

No asterisk (*) indicates being not Useful or Important, based on your votes received

(W) indicates **Widely Used,** (U) indicates **Used**, (L) indicates **Least Used**

According to the results, in general documents that were identified as less important and useful for Scrum roles (e.g. company-wide quality procedures, DB designs, Developer documentation and Traceability records) were found to be the documents that are least used in Scrum projects, which vindicates the results in Round 2 (in the main). However, Risk Analysis reports, Bug Verification reports and Training materials—documents that were identified as not very important or useful for all the roles (Round 2) were found to be having high usage levels (Category "U" in Table 4.21) documents in Scrum projects. Documents that were found to be important and useful for all the roles (e.g. Business Flows, Maintained Product Backlog and Stakeholder/Customer requirements) were found to be having very high/wide usage levels (Category "W" in Table 4.21). Hence it can be concluded that widely used documents across Scrum teams are important and useful most Scrum teams based. Equally importantly, in the main, the results found in the validation round were consistent with those found in the main rounds of data collection, thus cross-validating the original findings to increase the generalizability of the findings.

CHAPTER 5
Conclusion

5.1 Introduction

This research was undertaken to study the apparent incompatibility between lack of emphasis (or discouragement) on documentation in Agile methodologies and emphasis on documentation in quality assurance, in relation to software development in *healthcare applications*. ISO 9001:2008 quality management system (QMS) standard was taken as the baseline for quality assurance. This study pivoted on the following research overall (overarching) question:

> *What would be the minimum level of documentation that would be acceptable for a Health-IT organization pursuing Scrum, if they are to maintain an internationally recognized QMS standard such as ISO 9001:2008?*

The literature review on the above overarching research question lead to three research questions, which have one-to-one correspondence with three specific research objectives (Table 5.1).

The study is important because Agile methodologies have now become the norm in software development and Agile practitioners have been, to some extent, misinformed by proponents of Agile software development that these methodologies should place a zero emphasis on documentation. Documentation on the other hand is a critical component in quality assurance, both from an audit standpoint and quality management standpoint. This is the reason why the internationally recognized ISO 9001:2008 QMS standard places such an emphasis on documentation.

Table 5.1: The Link between Research Questions and Research Objectives

The Overall Research Question: *What would be the minimum level of documentation that would be acceptable for a Health-IT organization pursuing Scrum, if they are to maintain an internationally recognized QMS standard such as ISO 9001:2008?*	
Research Question	**Research Objective**
RQ1 *Given that ISO 9001:2008 QMS accreditation is identified as the baseline and given that Scrum is the working method of the health-care software development company, what specific operational-level documentations appear to be useful and important for such a company to comply with quality standards?*	**Objective 1** To identify the critically important documents for quality assurance, more specifically, to (seemly) meet the documentation requirements of ISO 9001:2008.
RQ2 *How concise and when can the documents be prepared to maintain good communication among different parties in the development lifecycle?*	**Objective 2** The content that should be included in each document and the stage of the project at which each document should each be prepared.
RQ3 *What are the issues that cause documentation processes to be less effective, in a typical Scrum environment?*	**Objective 3** To identify the issues surrounding documentation.

5.2 How the Research Questions Were Addressed

In order to answer the research questions, and thereby achieve the research objectives, data were collected from panels of experts—personnel experienced in Agile software development in healthcare, particularly the “Scrum” approach, which is the focus of the study—using a *Delphi-like approach* through three rounds. The primary data collection instrument in each round was a questionnaire. Data required to answer the research questions were collected in sequential fashion (findings in one stage being an input to the next stage), converging towards answers for each research question. Figure 5.1 depicts the nexus between the research questions, data collection rounds and the research findings. Majority of panelists (15 out of 20) for the *three main rounds* study came from a leading healthcare software developer based in Auckland (the case study organization). The findings to research questions were validated by collecting fresh data from a different panel of experts in a *validation round*. For the validation round, a sizable proportion of experts outside the case study organization (5 out of 20 respondents) were involved to increase the external validity (generalizability) of the findings. Findings in the validation round were similar to the findings that came from the three main data collection rounds, which confirmed the external validity of the findings.

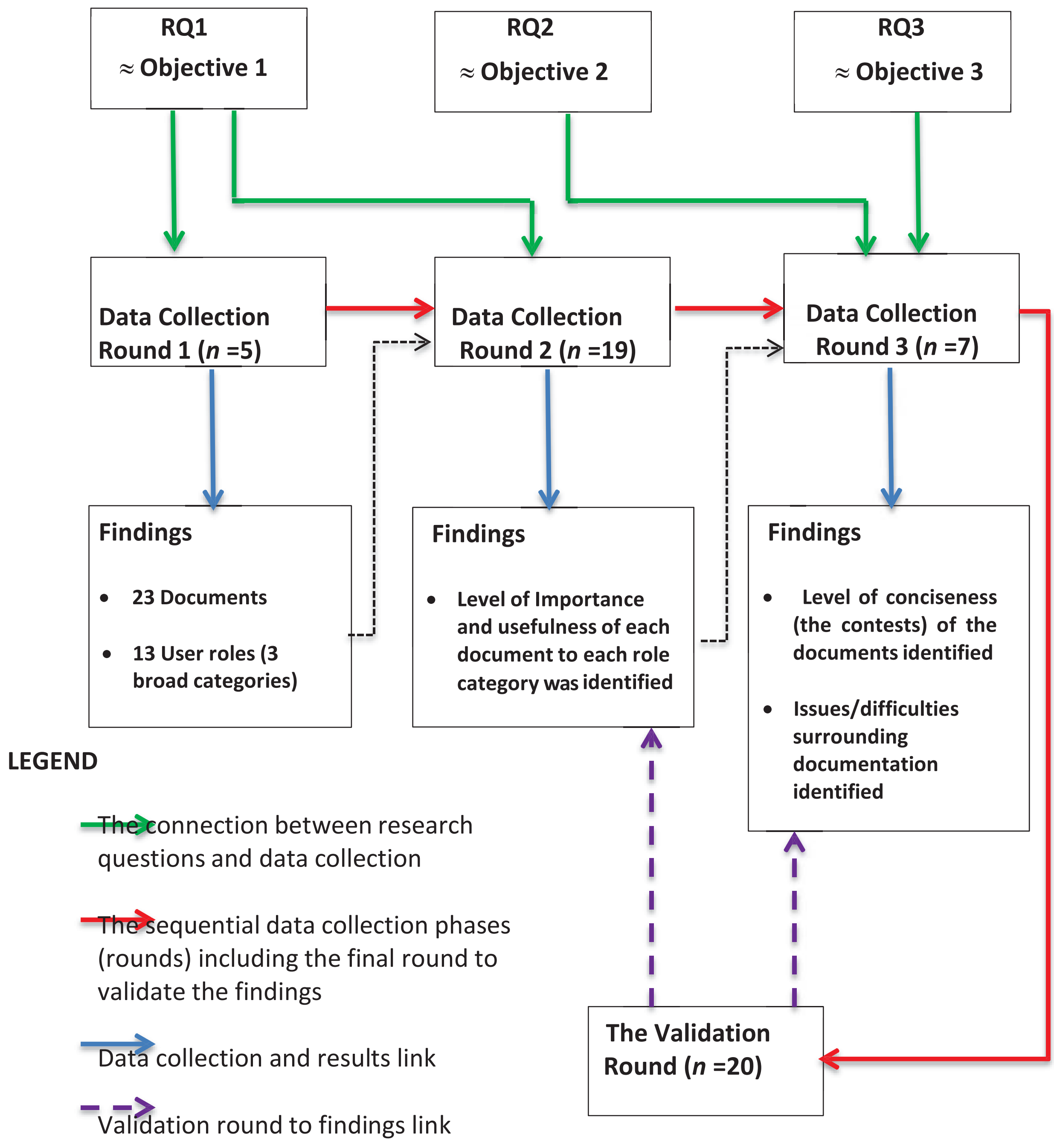

RQ1
≈ Objective 1
RQ2
≈ Objective 2
RQ3
≈ Objective 3
Data Collection Round 1 (n =5)
Data Collection Round 2 (n =19)
Data Collection Round 3 (n =7)
Findings
23 Documents
13 User roles (3 broad categories)
Findings
Level of Importance and usefulness of each document to each role category was identified
Findings
Level of conciseness (the contests) of the documents identified
Issues/difficulties surrounding documentation identified
The Validation Round (n =20)
LEGEND
The connection between research questions and data collection
The sequential data collection phases (rounds) including the final round to validate the findings
Data collection and results link
Validation round to findings link

Figure 5.1: The nexus between research questions, data collection, and findings

5.3 Key Findings and Achievement of Research Objectives

In the first round of the study, the panelists (*n* = 5) identified 23 documents that could be identified as being useful and/or important in Scrum, from a quality assurance (ISO 9001:2008) standpoint (details in section 4.2.1, Chapter 4). The panel also identified 13 different Scrum roles (the product owner, stakeholder, team lead, business analysist etc.) and that these roles could be represented by three board role categories: The Product Owner, the Scrum Master, and the Development Team.

In the second round of the study, in which researcher completed data/information collection to answer RQ1, the panelists (*n* = 19)[7] identified the degree of usefulness and importance of each document to each role category (the heuristics used and other details along with the summary of the findings are given in section 4.2.2, Chapter 4).
Given the researcher's definition of usefulness of a document and importance of a document—useful means possible benefits to the organization while importance means the risk/impact on the organization for not having the document—the researcher recommended that the focus should initially be on documents that are most important to each role, so that these documents should be concisely prepared at the right stage of the project.

In the third round of the study (details in sections 4.3 and 4.4, Chapter 4), in which researcher completed data/information collection to answer the remaining research questions (RQ2 and RQ3), the panelists identified the following:

(a) The relevant sections and content that should be included in each document to suit the three role categories (results summarized in Tables 4.13 through to 4.15, Chapter 4).

(b) The most appropriate stage to produce each document (results summarized in Table 4.16, Chapter 4).

(c) Negative experiences Scrum practitioners have had on documentation due to technical deficiencies in the documents.

[7] Of the 20 panelists, the responses of one member were discarded due to substantial proportion of missing responses.

(d) Problems Scrum practitioners face (due to nontechnical/practical reasons) in creating important documents.

Regarding (a) above (details in section 4.3.1.1, Chapter 4), the researcher found that in general, the *‘purpose’, ‘author’, ‘approvers / reviewers / sign-off’, and ‘comments on validity’* are important and relevant to all 3 roles, in the case of all 23 documents listed. Researcher also found that the *‘internal communication records’ and ‘preventive action records’* are more important and relevant to Development Teams rather than Scrum Masters and Product Owners. Likewise, in regards to the document content in general, for all listed documents, the *‘verification and validation information’* were found to be important and relevant to all 3 roles. However, again *‘flow charts/diagrams’* were found to be important and relevant to Development Teams and Product Owners rather than the Scrum Master, while the *‘known issues’* was found to be mostly relevant to Development Teams only. These findings can be used to make documents as concise as possible, this helping to exclude nonvalue adding components in documents.

Regarding (b) above (details in section 4.3.1.2, Chapter 4), the researcher found that several documents including Developer Documentation, DB design, Unit Test and Report, and the Test Verification Report can be created within the sprints. Documents that were found to be needed when the sprint begins (e.g. the Quality Procedure, Test Plans, and Stakeholder Requirements) can be created either at the project initiation stage (before sprint begins) or at the beginning of the sprint itself. However, some documents (e.g. Demo Recordings and Retro Notes) were found to be needed at end of a sprint. Needless to say, producing the documentation at the most appropriate phase can minimize additional effort, giving the maximum benefits to its audience, in keeping with the spirit of Scrum.

Regarding (c) above (details in section 4.4.1.1, Chapter 4), researcher found seven negative experiences Scrum practitioners encounter in relation to documentation: (i) nontechnical customers finding the documents difficult to understand; (ii) purpose of a document being not explicit; (iii) not following-up with client’s feedback; (iv) excessive rework due to frequent changes in requirements; (v) deficiencies in the validity of the documents; (vi) lack of risk analysis reports; and (vii) disruptions in the software development work. These are all due to technical deficiencies in the documents. The ramifications of each deficiency were discussed to enable practitioners to address these deficiencies.

Regarding (d) above (details in section 4.4.1.2, Chapter 4), again researcher found seven issues practitioners encounter in creating important documents: (i) lack of skilled document writers; (ii) waiting till the last minute to prepare the documents; (iii) misunderstanding of the purpose and intent of Agile; (iv) lack of a common documentation standard; (v) perceiving that preparing documents is a burden; (vi) poor tooling for documentation; and (vii) not having the right people at the right time. All these can be classified as nontechnical/practical problems. The ramifications of each issue were discussed to enable practitioners to address these issues.

5.4 Specific Conclusions

Given the above findings and the accompanying discussion in the previous chapter, the following can be concluded in the light of the overarching research question: *what would be the minimum level of documentation that would be acceptable for a Health-IT organization pursuing Scrum, if they are to maintain an internationally recognized QMS standard such as ISO 9001:2008?*

- High-level documentation such as quality policy and quality objectives as well as the quality manual do not seem to conflict with Agile/Scrum. The conflict exists at operational-level documents.

- At operational level, 23 sets of documents seem to be useful and important for health software developers adopting the Scrum methodology, in order to comply with the documentation requirements of ISO 9001:2008.

- To what extent each document becomes useful and important depends on the particular user (three user categories were identified in the study). What is important to the *development team* in general was not found to be important to the *product owner* and the *scrum master* and vice versa.

- The content in a particular document should fit the requirements of the user to whom the document is most useful and important. In addition, the documents need to be up-to-date and concise and needs to be introduced at the right stage to increase value, so as to maintain the spirit of Agile/Scrum.

- Institutional barriers to document implementation (details in section 4.4.1.2, Chapter 4) need to be eliminated to integrate QMS standards such as ISO 9001:2008 in Agile/Scrum because documentation is an integral part of quality assurance.

5.5 Limitations of the Study and Future Directions

The findings of this study are based on a case study. Although multiple participants were utilized to collect data and the validation round of data collection cross-validated the findings to increase the external validity (generalizability) of the results to a great extent, findings of a case study should be generalized with caution. The findings of this may not be generalizable across the universe of Health-IT organizations pursuing Scrum. More over due to time and other resource constraints, a relatively small sample was selected to administer the survey instrument, which reduced the internal validity of the results also. Another reason what could have affected the internal validity of the findings could have been the exposure of the panel members to Agile/Scrum. The researcher had to select a purposive sample and although the respondents recruited as panel members in the study have had adequate experience, they were not the most experienced Scrum practitioners in the organization.

Using the findings of this study as the background, future research could be conducted using a larger sample (also using more experienced Scrum practitioners) drawn from several healthcare originations. These studies could also focus on identifying specific issues salient to healthcare software development to answer the third research question more comprehensively. Also ISO 9001:2008 QMS standard has just been replaced by the ISO 9001:2015 QMS standard, which places more emphasis on quality risk assessment, which in turn may have a different implication on documentation. As such future research should consider the requirements stipulated in the new standard.

5.6 Final thoughts

Like most disciplines, quality management and operations management approaches such as Agile are evolving. There is a great deal of interplay between quality management (e.g. quality management system standards) and operations management (e.g. Agile), with each discipline being cross-fertilized by the other. More specifically, Agile methodologies (e.g. Scrum) do not favor documentation. In addition, Agile looks at managing/mitigating risks more effectively. The newest quality management system standard ISO 9001:2015 takes these aspects into consideration. For example, documentation requirements now depend on the situation, and documents such as quality manuals are no longer mandatory. In addition, the new standard attempts to address risk more explicitly. However, this study identified that in Agile/Scrum, documentation is needed not only to verify compliance with quality management system standards but also for smooth functioning of the day-to-day business. Thus the researcher expects Agile methodologies to embrace documentation as an important facet to satisfy the needs of internal and external stakeholders.

References

Abrahamsson, P. (2002). *Agile Software Development Methods: Review and Analysis (VTT publications)*.

Akif, R., & Majeed, H. (2012). Issues and challenges in Scrum implementation. *International Journal of Scientific & Engineering Research, 3*(8), 1-4.

Ambler, S. W. (2006). Agile Model Driven Development (AMDD) Scaling Agile Software Development. *IBM Coperation: Rational Software*.

Ambler, S. W. (2012). Best Practices for Agile/Lean Documentation. *Agile Modeling.* from http://www.agilemodeling.com/essays/agileDocumentationBestPractices.htm

Anderson, C. L., & Agarwal, R. (2011). The digitization of healthcare: boundary risks, emotion, and consumer willingness to disclose personal health information. *Information Systems Research, 22*(3), 469-490.

Archer, J. (2005). *Agile design*

Armour, Q., & Thizy, D. (2010). Developing Successful Healthcare Software: 10 Critical Lessons. Retrieved from MACADAMIAN website:

AS/NZS, S. (2000). AS/NZS ISO 9001:2000. *Standards Australia and Standards New Zealand.*

Atahualpa, W. (2014). Waterfall-model
from http://www.waterfall-model.com/

Australian and New Zealand Standard. (2006). Quality Management Systems *AS/NZS ISO 9000:2006*. Australia/New Zealand: Standard Austrailia , New Zealand.

Avision, D., & Fitzgerald, G. (2003). Where now for development methodologies. *Communications of the ACM, 46*(1), 78-82.

Awad, M. (2005). A Comparison between Agile and Traditional Software Development Methodologies. *School of Computer Science and software Engineering, The University of Western Australia,* .

Babbie, E. (2015). *The practice of social research*: Cengage Learning.

Bach, J. (2003). Exploratory Testing Explained. 2.

Basil, V. R., & Turner, A. J. (1975). Iterative enhancement: A practical technique for software development. *Software Engineering, IEEE Transactions on*(4), 390-396.

Batten, A. H. (1973). Binary and multiple systems of stars.

Beck, K., Beedle, M., Bennekum, A. v., Cockburn, A., Cunningham, W., Fowler, M., . . . Thomas, D. (2001). Agile Mainfesto. from http://agilemanifesto.org/

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., . . . Jeffries, R. (2001). Manifesto for agile software development.

Benbasat, I., Goldstein, D. K., & Mead, M. (1987). The case research strategy in studies of information systems. *MIS quarterly*, 369-386.

Benbasat, I., & Weber, R. (1996). Research commentary: Rethinking “diversity” in information systems research. *Information systems research, 7*(4), 389-399.

Blaikie, N. (2009). *Designing social research*: Polity.

Brennan, V. (2010). Controlling your documents and records.

Brewer, M., Reis, H., & Judd, C. (2000). Research design and issues of validity. *Handbook of research methods in social and personality psychology*, 3-16.

Bryman, A. (2012). *Social research methods*: Oxford university press.

Cajander, Å., Larusdottir, M., & Gulliksen, J. (2013). Existing but not explicit-The user perspective in Scrum projects in practice *Human-Computer Interaction–INTERACT 2013* (pp. 762-779): Springer.

Campbell, D. T., Stanley, J. C., & Gage, N. L. (1963). *Experimental and quasi-experimental designs for research*. Houghton Mifflin Boston.

Cavaye, A. L. (1996). Case study research: a multi‐faceted research approach for IS. *Information systems journal, 6*(3), 227-242.

Cellucci, L. W., Trimmer, K., Farnsworth, T., & Waldron, A. (2011, 4-7 Jan. 2011). *The Implementation of a Health-IT Academic Focus: A Case Study.* Paper presented at the System Sciences (HICSS), 2011 44th Hawaii International Conference on.

Chang, T., & Lee, M. (2005). Applying TQM, CMM and ISO 9001 in knowledge management for software development process improvement. *International Journal of Services and Standards, 2*(1), 101-115.

Cho, J. (2008). Issues and Challenges of agile software development with SCRUM. *Issues in Information Systems, 9*(2), 188-195.

Cianfrani, C. A., Tsiakals, J. J., & West, J. (2009). ISO 9001:2008 explained (3rd ed.). *Milwaukee, WI: ASQ Press*.

Cohen, D., Lindvall, M., & Costa, P. (2003). Agile software development. *DACS SOAR Report*(11).

Coleman, G., & O'Connor, R. (2008). Investigating software process in practice: A grounded theory perspective. *Journal of Systems and Software, 81*(5), 772-784.

Conboy, K., Wang, X., & Fitzgerald, B. (2009). Creativity in Agile Systems Development: A Literature Review *Information Systems–Creativity and Innovation in Small and Medium-Sized Enterprises* (pp. 122-134): Springer.

Creswell, J. W. (2014). *A concise introduction to mixed methods research*: SAGE Publications.

Cristal, M., Wildt, D., & Prikladnicki, R. (2008, 17-20 Aug. 2008). *Usage of SCRUM Practices within a Global Company.* Paper presented at the Global Software Engineering, 2008. ICGSE 2008. IEEE International Conference on.

Dalkey, N. (1972). *Studies in the quality of life; Delphi and decision making.*

Dalkey, N., Brown, B., & Cochran, S. (1969). *The Delphi method: An experimental study of group opinion* (Vol. 3): Rand Corporation Santa Monica, CA.

Darke, P., Shanks, G., & Broadbent, M. (1998). Successfully completing case study research: combining rigour, relevance and pragmatism. *Information systems journal, 8*(4), 273-289.

David Avison, & Fitzgerald, G. (2003). *Information systems development: methodologies, techniques and tools*: McGraw Hill.

De Vaus, D. A., & de Vaus, D. (2001). *Research design in social research*: Sage.

De Vries, H. J. (2001, 2001). *Standardization-a new discipline?* Paper presented at the Standardization and Innovation in Information Technology, 2001 2nd IEEE Conference.

Dehling, T., & Sunyaev, A. (2014, 6-9 Jan. 2014). *Information Security and Privacy of Patient-Centered Health IT Services: What Needs to Be Done?* Paper presented at the System Sciences (HICSS), 2014 47th Hawaii International Conference on.

Denzin, N. K. (2010). Moments, mixed methods, and paradigm dialogs. *Qualitative inquiry*.

Dubé, L., & Paré, G. (2003). Rigor in information systems positivist case research: current practices, trends, and recommendations. *MIS quarterly*, 597-636.

Eden, D., & Aviram, A. (1993). Self-efficacy training to speed reemployment: Helping people to help themselves. *Journal of applied Psychology, 78*(3), 352.

Eisenhardt, K. M. (1989). Building theories from case study research. *Academy of management review, 14*(4), 532-550.

Elke, H., & Roland, M. (2008). *Agile Process Myths*. Paper presented at the Proceedings of the 2008 international workshop on Scrutinizing agile practices or shoot-out at the agile corral, Leipzig, Germany.

Esker. (2008). Agile approach to solution delivery In H. i. works (Ed.).

Feagin, J. R., Orum, A. M., & Sjoberg, G. (1991). *A case for the case study*: UNC Press Books.

Filipe , F. C., Ademar, A., Hugo , S. F., & Nuno, F. (2009). *Patterns for consistent software documentation*. Paper presented at the Proceedings of the 16th Conference on Pattern Languages of Programs, Chicago, Illinois.

Flyvbjerg, B. (2006). Five misunderstandings about case-study research. *Qualitative inquiry, 12*(2), 219-245.

Forkner-Dunn, J. (2003). Internet-based patient self-care: the next generation of health care delivery. *Journal of Medical Internet Research, 5*(2), e8.

Fraenkel, J. R., Wallen, N. E., & Hyun, H. H. (1993). *How to design and evaluate research in education* (Vol. 7): McGraw-Hill New York.

Frost, J. H., & Massagli, M. P. (2008). Social uses of personal health information within PatientsLikeMe, an online patient community: what can happen when patients have access to one another's data. *Journal of Medical Internet Research, 10*(3).

Gagnon, M. P., & Scott, R. E. (2005). Striving for evidence in e-health evaluation: lessons from health technology assessment. *Journal of telemedicine and telecare, 11*(suppl 2), 34-36.

Ghezzi, C., Jazayeri, M., & Mandrioli, D. (2002). *Fundamentals of software engineering*: Prentice Hall PTR.

Ghosh, R. (2015). What is the difference between 'usefulness' and 'importance'?

Gibbert, M., Ruigrok, W., & Wicki, B. (2008). Research notes and commentaries what passes as a rigorous case study. *Strategic management journal, 29*(13), 1465-1474.

Grimsley, S. (2003). Stakeholder in Business.

Hambling, B., & Van Goethem, P. (2013). *User acceptance testing*: BCS Learning & Development.

Hampton, D. M. (2014). *A step forward. Quality Progress* (Vol. 47(3)).

Harvey, J. (1998). Service quality: a tutorial. Journal of Operations Management. *16*(5), 583-597.

HealthIT. (2013). Basics of Health IT. from http://www.healthit.gov/patients-families/basics-health-it

Heeager, L. T. (2012). Introducing agile practices in a documentation-driven software development practice: A case study. *Journal of Information Technology Case & Application Research, 14*(1), 3-24.

Hooper, J. H., Cianfrani, C. A., Tsiakals, J. J., & West, J. E. (2001). The process approach to QMS in ISO 9001 and ISO 9004*,* 70.

Hossain, E., Babar, M., Hye-young, P., & Verner, J. (2009, 1-3 Dec. 2009). *Risk Identification and Mitigation Processes for Using Scrum in Global Software*

Development: A Conceptual Framework. Paper presented at the Software Engineering Conference, 2009. APSEC '09. Asia-Pacific.

Hoyle, R. H., Harris, M. J., & Judd, C. M. (2002). *Research methods in social relations*: Thomson Learning.

Hsu, C., & Sandford, B. (2007). The Delphi Technique: Making Sense Of Consensus.

IBM. (2005). *Defining Customer and Business Requirements*

IIBA. (2010). Business Analysis.

ISO. (1994). *ISO 8402: 1994: Quality Management and Quality Assurance-Vocabulary*: International Organization for Standardization.

ISO. (2009). ISO 9001:2008. Switzerland: International Standard Organization.

James, B. C. (1989). Quality management for health care delivery. *The Hospital Research and Educational Trust*.

Janikow, C. Z., & Michalewicz, Z. (1991). *An experimental comparison of binary and floating point representations in genetic algorithms.* Paper presented at the ICGA.

Jansen, B. J., Spink, A., & Saracevic, T. (2000). Real life, real users, and real needs: a study and analysis of user queries on the web. *Information processing & management, 36*(2), 207-227.

Jepsen, T., & Jepsen, M. (2008). Health IT: A Roundtable Discussion with Healthcare Information Technology Executives. *IT Professional, 10*(2), 53-55. doi: 10.1109/MITP.2008.27

Joaquim, B. (2008). *Agile documentation with uScrum*. Paper presented at the Proceedings of the 26th annual ACM international conference on Design of communication, Lisbon, Portugal.

Juran, J. M., & Gryna, F. M. (1988). *Juran's quality control handbook / J.M. Juran, editor-in-chief, Frank M. Gryna, associate editor*: New York : McGraw-Hill, c1988

4th ed.

Kaiser, H., & Dickman, K. (1962). Sample and population score matrices and sample correlation matrices from an arbitrary population correlation matrix. *Psychometrika, 27*(2), 179-182. doi: 10.1007/BF02289635

Kaplan, B., & Duchon, D. (1988). Combining qualitative and quantitative methods in information systems research: a case study. *MIS quarterly*, 571-586.

Karsh, B. (2004). Beyond usability: designing effective technology implementation systems to promote patient safety. *Quality and Safety in Health Care, 13*(5), 388-394.

Kleppe, A. G., Warmer, J., Bast, W., & Explained, M. (2003). The model driven architecture: practice and promise: Addison-Wesley Longman Publishing Co., Inc., Boston, MA.

Kotrlik, J., & Higgins, C. (2001). Organizational research: Determining appropriate sample size in survey research appropriate sample size in survey research. *Information technology, learning, and performance journal, 19*(1), 43.

Ktata, O., & Lévesque, G. (2010). *Designing and Implementing a Measurement Program for Scrum Teams: What do agile developers really need and want?* Paper presented at the Proceedings of the Third C* Conference on Computer Science and Software Engineering.

Kua, P. (2014). *Talking with Tech Leads*.

Kumar, R. (2005). *Research Methodology* (Second edition ed.). India: Dorling Kindersley.

Lee, G., & Xia, W. (2010). Toward agile: An integrated analysis of quantitative and qualitative field data on software development agility. *MIS Quarterly 34*(1), 87-114.

Leszak, M., Perry, D. E., & Stoll, D. (2000). *A case study in root cause defect analysis*. Paper presented at the Proceedings of the 22nd international conference on Software engineering, Limerick, Ireland. http://dl.acm.org/citation.cfm?doid=337180.337232

Leung, H. K., Liao, L., & Qu, Y. (2007). Automated support of software quality improvement. *International Journal of Quality & Reliability Management, 24*(3), 230-243.

Levinson, W., Kao, A., Kuby, A., & Thisted, R. (2005). Not all patients want to participate in decision making. *Journal of general internal medicine, 20*(6), 531-535.

Lincoln, Y. S., Lynham, S. A., & Guba, E. G. (2011). Paradigmatic controversies, contradictions, and emerging confluences, revisited. *The Sage handbook of qualitative research, 4*, 97-128.

Lindgreen, A., Hingley, M., Stavros, C., & Westberg, K. (2009). Using triangulation and multiple case studies to advance relationship marketing theory. *Qualitative Market Research: An International Journal, 12*(3), 307-320.

Mandi, K. D., & Lee, T. H. (2002). Integrating medical informatics and health services research. *Journal of the American Medical Informatics Association, 9*(2), 127-132.

Mansour, S. (2013). agile-requirements-documentation. Retrieved from http://blogs.atlassian.com/2013/07/agile-requirements-documentation-a-guide/

Mathison, S. (1988). Why triangulate? *Educational researcher, 17*(2), 13-17.

May, T. (2011). *Social research*: McGraw-Hill Education (UK).

McCluskie, C. (2013). Benefits of an Organisation-Wide Quality Management Solution.

McConnell, H. (2003). International efforts in implementing national health information infrastructure and electronic health records. *World hospitals and health services: the official journal of the International Hospital Federation, 40*(1), 33-37, 39-40, 50-32.

McMichael, B., & Lombardi, M. (2007). ISO 9001 and Agile Development. *AGILE 2007 (AGILE 2007)*, 262.

Melis, M., Ambu, W., Pinna, S., & Mannaro, K. (2004). *Requirements of an ISO Compliant XP Tool.* Universit´a di Cagliari. (3092(Springer, Heidelberg))

Merriam, S. B. (1998). *Qualitative Research and Case Study Applications in Education. Revised and Expanded from" Case Study Research in Education."*: ERIC.

Miller, G. G. (2001). *The characteristics of agile software processes.* Paper presented at the tools.

Ming, H., Verner, J., Liming, Z., & Babar, M. A. (2004, 28-30 Sept. 2004). *Software quality and agile methods.* Paper presented at the Computer Software and Applications Conference, 2004. COMPSAC 2004. Proceedings of the 28th Annual International.

Mohammed, A. H. (2006). Quality management system in construction.

Momtahan, L., Lloyd, S., & Simpson, A. (2007, 20-22 June 2007). *Switched Lightpaths for e-Health Applications: Issues and Challenges.* Paper presented at the Computer-Based Medical Systems, 2007. CBMS '07. Twentieth IEEE International Symposium on.

Morgan, D. L. (2014). Pragmatism as a paradigm for social research. *Qualitative Inquiry*, 1077800413513733.

Myers, M. D., & Avison, D. (1997). Qualitative research in information systems. *Management Information Systems Quarterly, 21*, 241-242.

Myhrberg, E. (2009). *A practical field guide for ISO 9001: 2008*: ASQ Quality Press.

Nashwille, M. (2015). Bugdays/Bug-verification. *MozillaWiki*.

Nawrocki, J. R., Jasinski, M., Walter, B., & Wojciechowski, A. (2002). Combining Extreme Programming with ISO 9000. *LECTURE NOTES IN COMPUTER SCIENCE*(2510), 786-794.

Oakland, J. S. (2013). *Total quality management and operational excellence : text with cases / John Oakland*: New York : Routledge, 2013
Fourth Edition.

Olivier, G., & Pierre, N. R. (2013). *A process practice to validate the quality of reused component documentation: a case study involving open-source components*. Paper presented at the Proceedings of the 2013 International Conference on Software and System Process, San Francisco, CA, USA.

Orlikowski, W., & Baroudi, J. (2008). Studying information technology in organizations : research approaches and assumptions (pp. 146-173).

Osteen, O., & Robbert, G. (1995). Interpreting the requirements in ISO 9001 for software development and maintenance An ISO 9000 Approach to Building Quality Software. *Prentice Hall*.

Pagliar, C. (2007). Design and Evaluation in eHealth: Challenges and Implications for an Interdisciplinary Fiel. *Journal of Medical Internet Research*

Parker, T. (2015). Technical Writing.

Pino, F. J., Pedreira, O., García, F., Luaces, M. R., & Piattini, M. (2010). Using Scrum to guide the execution of software process improvement in small organizations. *Journal of Systems and Software, 83*(10), 1662-1677. doi: http://dx.doi.org/10.1016/j.jss.2010.03.077

PMI. (2011). Who are Project Managers?

Poksinska, B., Jörn Dahlgaard, J., & Eklund, J. A. (2006). From compliance to value-added auditing–experiences from Swedish ISO 9001: 2000 certified organisations. *Total Quality Management & Business Excellence, 17*(7), 879-892.

Punch, K. F. (2013). *Introduction to social research: Quantitative and qualitative approaches*: Sage.

Qumer, A., & Henderson-Sellers, B. (2006). *Measuring agility and adoptability of agile methods: a 4-dimensional analytical tool.* Paper presented at the Procs. IADIS International Conference Applied Computing 2006.

Radigan, D. (2015). Epics, stories, versions, and sprints.

Ragin, C. (1989). The Comparative Method: Moving Beyond Qualitative and Quantitative Strategies: UNIV NORTH CAROLINA PRESS BOX 2288, CHAPEL HILL, NC 27515-2288.

Rashina, H., James, N., & Stuart, M. (2010). *How much is just enough?: some documentation patterns on Agile projects*. Paper presented at the Proceedings of the 15th European Conference on Pattern Languages of Programs, Irsee, Germany.

Rasmussen, R., Hughes, T., Jenks, J. R., & Skach, J. (2009, 24-28 Aug. 2009). *Adopting Agile in an FDA Regulated Environment.* Paper presented at the Agile Conference, 2009. AGILE '09.

Ratanotayanon, S., Kotak, J., & Sim, S. E. (2006). *After the Scrum: Twenty Years of Working without Documentation.* Paper presented at the SEKE.

Rico, D. (2005). Short history of software methods. 1,3,4,5.

Rico, D. (2010). Agile Methods and Software Documentation

Rising, L., & Janoff, N. S. (2000). The Scrum software development process for small teams. *IEEE software*(4), 26-32.

Ritchie, J., Lewis, J., Nicholls, C. M., & Ormston, R. (2013). *Qualitative research practice: A guide for social science students and researchers*: Sage.

Rottier, P. A., & Rodrigues, V. (2008, 4-8 Aug. 2008). *Agile Development in a Medical Device Company.* Paper presented at the Agile, 2008. AGILE '08. Conference.

Rouse, M. (2007). Unit Testing Definition. *Software Quality*.

Rüping, A. (2003). *Agile Documentation: A Pattern Guide to Producing Lightweight Documents for Software Projects*. The Atrium, Southern Gate, Chichester, West Sussex PO19 8SQ, England: John Wiley & Sons Ltd, .

Sampaio, P., Saraiva, P., & Guimarães Rodrigues, A. (2009). ISO 9001 certification research: questions, answers and approaches. *International Journal of Quality & Reliability Management, 26*(1), 38-58.

Sarantakos, S. (2012). *Social research*: Palgrave Macmillan.

Schlickman, J. (2003). *ISO 9001: 2000 Quality management system design*: Artech House.

Schwaber, K., & Beedle, M. (2002). Agile Software Development with Scrum.

Schwaber, K., & Sutherland, J. (2013). *The Scrum guide: The Definitive Guide to Scrum: The Rules of the Game*

Seela, S. (2015). How to Create Requirements Traceability Matrix. *Sofware testing*.

Sergio, C., Nicolas, A., & Thia, O. (2005). *A study of the documentation essential to software maintenance*. Paper presented at the Proceedings of the 23rd annual international conference on Design of communication: documenting &amp; designing for pervasive information, Coventry, United Kingdom.

Sinha, M., & Hernandez, H. (2010). Quality audit as a driver for compliance to ISO 9001: 2008 standards. *The TQM Journal, 22*(4), 454-466.

Skulmoski, G. J., Hartman, F. T., & Krahn, J. (2007). The Delphi Method for Graduate Research. *Journal of Information Technology Education, 6*.

Stalhane, T., & Hanssen, G. (2008). The Application of ISO 9001 to Agile Software Development. *Lecture Notes in Computer Science* (5089), 371-385.

Stalhane, T., & Hanssen, G. (2009). The application of ISO 9001 to agile software development *The Norwegian University of Science and Technology*

Stober, T., & Hansmann, U. (2010). Traditional Software Development *Agile Software Development* (pp. 15-33): Springer.

Stoica, M., Mircea, M., & Ghilic-Micu, B. (2013). Software Development: Agile vs. Traditional. *Bucharest University of Economic Studies, Romania 17*(4).

Summers, D. C. S. (2010). *Quality.* 5th ed. Boston : Prentice Hall.

Sutherland, J. (2004). Agile development: Lessons learned from the first Scrum. *Cutter Agile Project Management Advisory Service: Executive Update, 5*(20), 1-4.

Sutherland, J., Viktorov, A., Blount, J., & Puntikov, N. (2007). *Distributed Scrum: Agile project management with outsourced development teams.* Paper presented at the System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on.

Taylor, R. N., Medvidovic, N., & Dashofy, E. M. (2009). *Software architecture: foundations, theory, and practice*: Wiley Publishing.

Thompson, P. (2007). Source Code. *Technical Terms*.

Tony, C. (2003). Documentation and agile methods: striking a balance (Vol. 35, pp. 12-13): ACM.

Tricker, R. (2014). *ISO 9001: 2008 for Small Businesses*: Routledge.

VanBaren, J. (2015). Business Flow Diagram.

Venkatesh, V., & Davis, F. D. (2000). A theoretical extension of the technology acceptance model: Four longitudinal field studies. *Management science, 46*(2), 186-204.

Venkatraman, N., & Ramanujam, V. (1986). Measurement of business performance in strategy research: A comparison of approaches. *Academy of management review, 11*(4), 801-814.

Voss, C., Tsikriktsis, N., & Frohlich, M. (2002). Case research in operations management. *International journal of operations & production management, 22*(2), 195-219.

Wealleans, D. (2005). *The quality audit for ISO 9001: 2000: a practical guide*: Gower Publishing, Ltd.

Weiner, J. (2007). Measurement: Reliability and Validity Measures.

Whittemore, R., Chase, S. K., & Mandle, C. L. (2011). Validity in Qualitative Research. *QUALITATIVE HEALTH RESEARCH, 11*(4).

Williams, L. (2003). Guest Editor's Introduction: The XP Programmer--The Few-Minutes Programmer. *IEEE Software*(3), 16-20.

Wirth, N. (2008). A Brief History of Software Engineering. 2.

Wohlin, C., Höst, M., & Henningsson, K. (2003). Empirical research methods in software engineering *Empirical methods and studies in software engineering* (pp. 7-23): Springer.

Wright, G. (2003). *Achieving ISO 9001 Certification for an XP Company*.

Yin, R. (1999). Enhancing the quality of case studies in health services research. *Health Services Research, 34*(5 Pt 2), 1209-1224.

Yin, R. (2013). *Case study research: Design and methods*: Sage publications.

Yunan, C. (2013, 20-24 May 2013). *Towards collaborative-oriented health IT systems design.* Paper presented at the Collaboration Technologies and Systems (CTS), 2013 International Conference on.

Appendix A

Round 1 Questionnaire

Shanuka Wickramasinghe
3 Landow Place
Henderson
Auckland 0612
30th October, 2014

Dear valued panel member,

Seeking Expert Opinion on Documentation Requirements in ISO 9001:2008 Quality Management System Standard: Delphi Survey Round 1

First of all, thank you very much for consenting to participate in the above survey as an expert in a software sub-discipline. ISO 9001:2008 certification has become a necessary requirement in some customer environments. Agile practitioners we know the challenges of getting all the documentation right for ISO 9001:2008 certification (the standard requires a certain minimum level of documentation). This research is carried out to identify documents likely be needed for ISO 9001:2008 certification of a vendor whose scope involves healthcare software development in a Scrum project environment. The research is part of the requirement of the *Master of Philosophy in Quality Systems degree I am pursuing at Massey University (part-time).*

As most of the Scrum teams do not prefer to write down or refer to lengthy documents, this research focuses on defining the most important documents and what sections of these should be included or excluded and what documents need to be totally eliminated from the current procedures, for the sake of ISO 9001:2008 certification. I am using a proven panel-based iterative problem solving method known as the Delphi method, which seeks to reach a high degree of census (among a diverse group of subject experts) within a few rounds of data collection. Data provided by individual panel members in each round will be collated by me and redistributed to the members as aggregate information with a view to improve the outcomes in the subsequent round. Nearly always, consensus is reached in about three rounds in Delphi studies.

The targeted audience for these questions are members of software development teams who work in Scrum environments, most specifically focusing on health care software development. The questions will be asked in 3 rounds (the most).

This first round of my survey consists of three general questions (areas) related to ISO 9001:2008, based on my interpretation of the above standard (I have also consulted a few auditors who are familiar with the standard). These questions can be viewed as terms of reference of the Delphi panel in the first round of the survey. In the second and third rounds the questions (the terms of reference of the Delphi panel) will become more specific, based on the responses received from the previous round. If you are happy to proceed with round 1, please answer to the following questionnaire.

Thank you for your support in advance.

Yours sincerely
Shanuka Wickramasinghe

BSc (Hons),
Business Analyst, Orion Health

MASSEY UNIVERSITY - THE ENGINE OF THE NEW NEW ZEALAND.
New Zealand is emerging with a new confidence on a global stage. As a country we have the desire, innovation and creativity to take on the best the world has to offer. Through our students, staff and alumni, Massey is the engine that is making a practical contribution to turning the wheels in this exciting new chapter in New Zealand. http://www.engine.ac.nz

Part A - Panel Member's General Information *(this information will not be circulated in the next round to preserve confidentiality and any bias in the subsequent rounds).*

Your Project:	
Your Job Tittle:	
Number of years of Orion Health Experience:	
Number of years of Industry Experience:	

Part B – Define list of document and Roles in Scrum projects.
Brainstorm and consolidate the results to define lists for Question1 and Question2.

Question 1: List documents that you are either,
- Currently using or
- might think as important / useful or
- seen in other projects hence would like to use in the future Scrum projects.

Question 2: List commonly seen software development roles in Scrum projects, so that they can be listed under theoretical Scrum projects who are PO, DT and SM

Appendix B

Round 2 Questionnaire

Shanuka Wickramasinghe
3 Landow Place
Henderson
Auckland 0612
10th November, 2014

Dear valued panel member,

Seeking Expert Opinion on Documentation Requirements in ISO 9001:2008 Quality Management System Standard: Delphi Survey Round 2

First of all, thank you very much for consenting to participate in the above survey as an expert in a software sub-discipline. ISO 9001:2008 certification has become a necessary requirement in some customer environments. Agile practitioners we know the challenges of getting all the documentation right for ISO 9001:2008 certification (the standard requires a certain minimum level of documentation). This research is carried out to identify documents likely be needed for ISO 9001:2008 certification of a vendor whose scope involves healthcare software development in a Scrum project environment. The research is part of the requirement of the *Master of Philosophy in Quality Systems degree I am pursuing at Massey University (part-time).*

As most of the Scrum teams do not prefer to write down or refer to lengthy documents, this research focuses on defining the most important documents and what sections of these should be included or excluded and what documents need to be totally eliminated from the current procedures, for the sake of ISO 9001:2008 certification. I am using a proven panel-based iterative problem solving method known as the Delphi method, which seeks to reach a high degree of census (among a diverse group of subject experts) within a few rounds of data collection. Data provided by individual panel members in each round will be collated by me and redistributed to the members as aggregate information with a view to improve the outcomes in the subsequent round. Nearly always, consensus is reached in about three rounds in Delphi studies.

The targeted audience for these questions are members of software development teams who work in Scrum environments, most specifically focusing on health care software development. The questions will be asked in 3 rounds (the most).

This round of my survey consists of three general questions (areas) related to ISO 9001:2008, based on my interpretation of the above standard (I have also consulted a few auditors who are familiar with the standard). These questions can be viewed as terms of reference of the Delphi panel in the first round of the survey. In the next rounds the questions (the terms of reference of the Delphi panel) will become more specific, based on the responses received from the previous round. If you are happy to proceed with round 2, please answer to the following questionnaire.

Thank you for your support in advance.

Yours sincerely
Shanuka Wickramasinghe

BSc (Hons),
Business Analyst, Orion Health

MASSEY UNIVERSITY - THE ENGINE OF THE NEW NEW ZEALAND.
New Zealand is emerging with a new confidence on a global stage. As a country we have the desire, innovation and creativity to take on the best the world has to offer. Through our students, staff and alumni, Massey is the engine that is making a practical contribution to turning the wheels in this exciting new chapter in New Zealand. http://www.engine.ac.nz

Part A - Panel Member's General Information *(this information will not be circulated in the next round to preserve confidentiality and any bias in the subsequent rounds).*

Your Project:	
Your Job Tittle:	
Number of years of Orion Health Experience:	
Number of years of Industry Experience:	

Part B – Terms of Reference in Round 1

Task 1: Understanding the Benefit of Each Document for Each Role In task 1, I am focusing your attention on 23 specific documents (Table 1) used in software development by different role players which was defined as a result of Round1. Please state which document(s) is/are of benefit in each role listed in Table 2. A sample row is filled up in Table 2 for illustrative purposes.

Table 1: List of Documents under Consideration

DOCUMENT	
1	Maintained product backlog
2	International Specifications
3	Standards & Regulations
4	Stakeholder/Customer requirements
5	Business Flows
6	Road Map Items (RMI)
7	Features
8	Test Verification Report
9	Sprint demonstration recordings
10	Retrospective notes
11	Training materials
12	Risk Analysis reports

13	Technical Design
14	DB designs
15	Developer documentation
16	Traceability Matrices
17	Unit Test and test reports
18	Source codes files
19	Bug tickets
20	Test plans
21	Bug Verification report
22	User manuals
23	Companywide Quality Procedure

Note: In Table 1, please add any document you think needs to be added to this list (I have given 3 blank spaces).

Table 2: Roles and Useful Documents
(The list of roles are defined as a result of Round1)

Role	**Useful documents (Please indicate the reference number given in Table 1)**
Sample role X	**1, 20**
Customer / Stakeholder	
Sales people	
Product owner	
Project manager or team-lead	
Business Analysts	
Technical lead	
Developers	
Testers	
User Interface developers	
Technical writes	
Implementation consultants	

Customer support engineers	
Quality assurance team members	

Task 2: Understanding the Importance of Each Document for Each Role In Task 2, I am your focusing your attention on the importance of each document in each role. In Table 3 shown below, please rate the importance of each document based for role given, using the 5-point scale given below. Use 1, 2, 3, 4 or 5 in each cell in Table 3. Three spare rows are given for you to inset any additional document that you may have mentioned in task 1).

1= extremely important;
2= Important;
3= neither Important nor Un-important;
4 = Not important;
5= Extremely Unimportant

Table 3: Rating Each Document for Each Role

Document	**Role**												
	Customer / Stakeholder	**Sales people**	**Product owner**	**Project manager or team-lead**	**Business Analysts**	**Technical lead**	**Developers**	**Testers**	**User Interface developers**	**Technical writes**	**Customer support engineers**	**Implementation consultants**	**Quality assurance team members**
Maintained product backlog													
International Specifications													
Standards & Regulations													
Stakeholder/Customer requirements													
Business Flows													
Road Map Items (RMI)													
Features													
Test Verification													

Report													
Sprint demonstration recordings													
Retrospective notes													
Training materials													
Risk Analysis reports													
Technical Design													
DB designs													
Developer documentation													
Traceability Matrices													
Unit Test and test reports													
Source codes files													
Bug tickets													
Test plans													
Bug Verification report													
User manuals													
Companywide Quality Procedure													

Appendix C
Round 3 Questionnaire

From: Shanuka Wickramasinghe

20, April 2015

To: Mr/MS XXX

Dear Mr/MS XXXX

The Panel Review to Identify the Essential Documentation Needed for ISO 9001:2008 Accreditation and Minimizing Quality Risks in a Scrum Software Development Environment- The Final Round

First of all, thank you for your valuable input in the previous round of inquiry in connection with the above matter. We have now come to the final round of review and once again I expect your support in responding to the specific queries outlined in section three (page 3). Let me first begin with the following background.

1. Introduction and Background

The aim of this project is to identify the following, in a Scrum software development environment: (a) the essential (minimum) documents required for ISO 9001:2008 quality management system accreditation, without compromising the quality risk, and (b) the specific stage of the (Scrum) project each document should be prepared.

In the first round (not all of you were involved) we were able to identify a catalogue of 23 documents (Table 2) that are thought be essential for a Scrum project, given the ISO 9001:2008 quality management system standard and the potential quality risk of not having these documents. In the previous round (the second round), based on your responses, it was possible to identify which of the 23 documents are useful and important to the 13 roles involved in Scrum projects. In analyzing the data, for the purpose of summarizing the results, I have assigned the 13 roles into three broad categories: the *Scrum master*, the *product owner*, and the *development team* (Table 1).

Note that I excluded sales people from my data analysis, thus limiting the number of roles to 12.

Table 1: Allocation of the Roles into Three main identified Scrum Roles

Scrum Master	**Product Owner**	**Development Team**
Project manager or team-lead	Product Owner	Technical lead
Business Analysts	Customer / Stakeholder	Developers
		Testers
		User Interface developers
		Technical writes
		Implementation consultants
		Customer support engineers
		Quality assurance team members

The objective of this third and final round of data collection is to:

1. Identify what *content* should be included in a document based on the role category (Table 2).

2. Identify at which stage of the Scrum project each document should be prepared.

3. Identify negative consequences related to documentation.

4. Identify the practical difficulties the Agile community faces in preparing important (and useful) documents.

The details of the tasks that you have to undertake, in the light of the findings of the previous round (a summary is provided in the next section), are given in section 3.

2. A Summary of Findings from the Previous Round (Round 2)

Table 2 shows which documents were being viewed by you—based on the % votes received and the criteria that I have used to filter information—as being useful and important for the three broad role categories (Table 1). To recollect, for the purpose of this study *a useful document* is defined as a document that is being viewed as having some place in Agile (e.g. for internal and external customer satisfaction, ISO 9001), while an important document is a document that is being viewed as having an impact on the quality performance of the software product that is being offered via a Scrum project. Thus you should view “important documents” as documents which needs to be prepared and implemented appropriately to minimize quality risks.

Table 2: The Usefulness and Importance of Each Document to Each Role Category

DOCUMENT	**Role Category**					
	Scrum Master		**Product Owner**		**Development Team**	
	Useful	**Important**	**Useful**	**Important**	**Useful**	**Important**
1. Maintained product backlog	**	*	**	*		*
2. International Specifications	**	*	**	*	*	*
3. Standards & Regulations	**	*	**	*	*	*
4. Stakeholder/Customer requirements	**	*	**	*	**	*
5. Business Flows	**	*	**	*	*	*
6. Road Map Items (RMI)	*	*	**	*	*	*
7. Features	**	**	**	**	*	*
8. Test Verification Report	*	*	*	*	**	**
9. Sprint demonstration recordings	*	*	*	*	*	**
10. Retrospective notes	**	*	**	*		*
11. Training materials		*		*		
12. Risk Analysis reports	**	*	**	*		
13. Technical Design	*		*		**	**
14. DB designs					**	**
15. Developer documentation					**	**
16. Traceability Matrices	*	*	*	*	*	**
17. Unit Test and test reports					**	**
18. Source codes files		*		*	**	**
19. Bug tickets	*	*	*	*	**	**
20. Test plans	*	*	*	*	**	**
21. Bug Verification report	*				**	**
22. User manuals		*		*	**	**
23. Companywide quality procedures	*	*	*	*	**	**

Notes: * Indicates either being **Useful** or **Important,** based on your votes received
Indicates either being **Very useful or **Very Important**, based on your votes received
No asterisk (*) indicates being not Useful or Important, based on your votes received

3. The Specific Tasks Relevant to This Final Round

The four specific tasks that you need to undertake are as follows.

Task 1: Identifying what should be written in a document by taking the role category into account

In Task 1, I would like you to identify what sections and content that you would like to see on a document when you are playing the given role (Table 1 depicts the 3 role categories).

Based on ISO 9001:2008 requirements for "4.2.4 Control of records" there are 2 main places found in a document which can maintain the records of a document. They are:

1. **The General Sections:** This is useful for a user to get more value out of the document. General sections are
   - viii. Purpose
   - ix. Author
   - x. Approvers / Reviewers / Sign off
   - xi. Internal communication records
   - xii. Corrective action records
   - xiii. Preventive action records
   - xiv. Comments on validity (Out of–date / relevant)

2. **Content of the document:** Apart from the general sections the versatile part on a document is it content. Depending on the user their preference for content can be changed. The content of a document can be,
   - g. User stories
   - h. Flow charts / diagrams
   - i. Development instructions
   - j. Release schedules / notes
   - k. Known issues
   - l. Verification and validation information

Sub task 1.1 Select which sections and content are relevant for the Scrum Master

Following is a list of documents that were identified as useful and/or important to the Scrum Master (see Table 2). Out of the sections and content defined above, assuming that you are playing the role of a Scrum Master, please select (by responding “Yes”) which sections and content that you would like to see in a document.

Document	**General Sections**							**Content**					
	1	**2**	**3**	**4**	**5**	**6**	**7**	**a**	**b**	**c**	**d**	**e**	**f**
Sample Document	**Yes**	**-**	**Yes**	**-**	**-**	**-**	**Yes**	**-**	**Yes**			**Yes**	**-**
Maintained Product Backlog													
Specifications													
Standards													
Stakeholder requirements													
Business flows													
RMI													
Features													
Test verification report													
Demo recordings													
Retro Notes													
Risk analysis reports													
Bug tickets													
Test plans													
Traceability matrix													
Quality Procedure													
User manuals													
Training Materials													
Source code files													
Tech Design													
Bug verification reports													

Sub task 1.2 Select which sections and content are relevant for the Product Owner

Following is a list of documents that were identified as useful and/or important to the Product Owner (see Table 2). Out of the sections and content defined above, assuming that you are playing the role of a Product Owner, please select (by responding “Yes”) which sections and content that you would like to see in a document.

Document	**General Sections**							**Content**					
	1	**2**	**3**	**4**	**5**	**6**	**7**	**a**	**b**	**c**	**d**	**e**	**f**
Sample Document	**Yes**	**-**	**Yes**	**-**	**-**	**-**	**Yes**	**-**	**Yes**			**Yes**	**-**
Maintained Product Backlog													
Specifications													
Standards													
Stakeholder requirements													
Business flows													
RMI													
Features													
Demo recordings													
Risk analysis reports													
User manuals													
Tech design													
Traceability matrix													
Training Materials													
Source code files													

Sub task 1.3 Select which sections and content are relevant for the Development Team

Following is a list of documents that were identified as useful and/or important to the Development Team (see Table 2).Out of the sections and content defined above, assuming that you are playing the role of a Development Team, please select (by responding "Yes") which sections and content that you would like to see in a document.

Document	**General Sections**							**Content**					
	1	**2**	**3**	**4**	**5**	**6**	**7**	**a**	**b**	**c**	**d**	**e**	**f**
Sample Document	**Yes**	**-**	**Yes**	**-**	**-**	**-**	**Yes**	**-**	**Yes**			**Yes**	**-**
Specifications													
Standards													
Stakeholder requirements													
Business flows													
RMI													

Features													
Test verification report													
Demo recordings													
Training materials													
Tech design													
DB design													
Developer documentation													
Traceability matrix													
Unit test and reports													
Source code files													
Bug tickets													
Test plans													
Bug verification reports													
User manuals													
Quality procedures													
Maintained Product backlog													
Retro Notes													

Sub task 1.4 Recollecting Negative Experiences

This sub-task involves a mini interview. Tell me some situations that you have come across where non-inclusion of the content (to which you have answered as “yes”) in any of the documents listed above—for either role category—has resulted in product quality issues (e.g. re-work, customer complaints).

Task 2: Identifying when the documents should be prepared in a Scrum project

Task 2 is designed to identify at which stage of a project a document should be produced/written/generated. Scrum project has the following main stages, which is identified as follows.

Stages of a Scrum project:
1. Project initiation stage
2. With the start of project development
3. Starting of a sprint
4. Within a sprint
5. At the end of a sprint
6. At the end of a release
7. Anytime on the project cycle
8. After the release

The following documents have been identified as “most important and useful documents”, **“**important and useful documents”, and “not very important and not very useful documents” for each role category. Please identify (by responding “Yes”) which you think is the relevant stage at which each document listed below should be prepared in a Scrum project.

Document	Stage							
	Project initiation stage	With the start of the project development	Starting of a sprint	Within a sprint	At the end of a sprint	At the end of a release	Anytime on the project cycle	After the release
Sample Document	Yes			Yes			Yes	
Most Important and Useful Documents								
Maintained Product backlog								
Standards								
Business flows								
RMI								
Features								
Demo recordings								
User manuals								
Tech design								
Source code files								
Training materials								
Specifications								
Stakeholder requirements								
Traceability matrix								
Important and Useful Documents								
Retro Notes								
Bug tickets								
Test plans								
Test verification report								
Quality Procedure								
Bug verification reports								
Risk analysis reports								
Not Very Important and Not Very Useful documents								

Unit test and report								
DB design								
Developer documentation								

Task 3: Recollecting Negative Experience

In this section, I want you to write-down as vividly as possible at least two different negative experiences (more preferred) that you have come across in preparing (or wanted to be prepared by a different person) key documents in a Scrum project. Reckon documents listed in Table 2 as the set of key documents required.

Negative Experience 1:

Negative Experience 2:

Negative Experience 3:

Negative Experience 4:

Negative Experience 5:

Negative Experience 6:

Note that I may interview you (see Task 4 below) if I need more clarification/elaboration on your above responses.

Task 4: Identifying difficulties on creating important documents (Table 2) in a Scrum Project.

This final task involves a mini interview. Please indicate to me the difficulties that you/your team is facing when producing important documents in your project. You can use following guidelines to provide feedback. Difficulties in terms of

- resources
- time / deadline
- budget
- people's passion
- regulations

Thank you and I look forward to receiving your responses within two weeks from receipt. For the tasks that involve mini-interviews, I hope you will give me 40 minutes of your valuable time for me to interview you.

Yours sincerely
Shanuka Wickramasinghe

Appendix D

Validation Round Questionnaire

Shanuka Wickramasinghe
3 Landow Place
Henderson
Auckland 0612
10th November, 2014

Dear valued panel member,

Seeking Expert Opinion on Documentation Requirements in ISO 9001:2008 Quality Management System Standard: Delphi Survey Round 2

First of all, thank you very much for consenting to participate in the above survey as an expert in a software sub-discipline. ISO 9001:2008 certification has become a necessary requirement in some customer environments. Agile practitioners we know the challenges of getting all the documentation right for ISO 9001:2008 certification (the standard requires a certain minimum level of documentation). This research is carried out to identify documents likely be needed for ISO 9001:2008 certification of a vendor whose scope involves healthcare software development in a Scrum project environment. The research is part of the requirement of the *Master of Philosophy in Quality Systems degree I am pursuing at Massey University (part-time).*

As most of the Scrum teams do not prefer to write down or refer to lengthy documents, this research focuses on defining the most important documents and what sections of these should be included or excluded and what documents need to be totally eliminated from the current procedures, for the sake of ISO 9001:2008 certification. I am using a proven panel-based iterative problem solving method known as the Delphi method, which seeks to reach a high degree of census (among a diverse group of subject experts) within a few rounds of data collection. Data provided by individual panel members in each round will be collated by me and redistributed to the members as aggregate information with a view to improve the outcomes in the subsequent round. Nearly always, consensus is reached in about three rounds in Delphi studies.

The targeted audience for these questions are members of software development teams who work in Scrum environments, most specifically focusing on health care software development. The questions will be asked in 3 rounds (the most).

These round is specifically designed to gain understanding on how documents are used in Scrum projects. Objective of this round is to use the results for validation purposes. If you are happy to proceed with this round, please answer to the following question.

Thank you for your support in advance.

Yours sincerely

Shanuka Wickramasinghe

BSc (Hons),
Business Analyst, Orion Health

MASSEY UNIVERSITY - THE ENGINE OF THE NEW NEW ZEALAND.
New Zealand is emerging with a new confidence on a global stage. As a country we have the desire, innovation and creativity to take on the best the world has to offer. Through our students, staff and alumni, Massey is the engine that is making a practical contribution to turning the wheels in this exciting new chapter in New Zealand. http://www.engine.ac.nz

Task 1: Understanding the Usage of Each Document

In this task, I am focusing your attention on the usage of each document in your current project. In Table 4 below, please state the usage of each document in your current project. If the answer to the question in the second column is 'No', please give the reason/s from 5 of the following, in the third column (you may give more than one reason/code). Three spare rows are given in Table 4 for you to inset any additional document that you may have mentioned in task 1)

Reasons for 'No'

a. I never knew this can be considered as a document
b. I don't see any value using this document
c. With the tight schedule we can't waste time on writing this
d. We don't have enough resources
e. Other

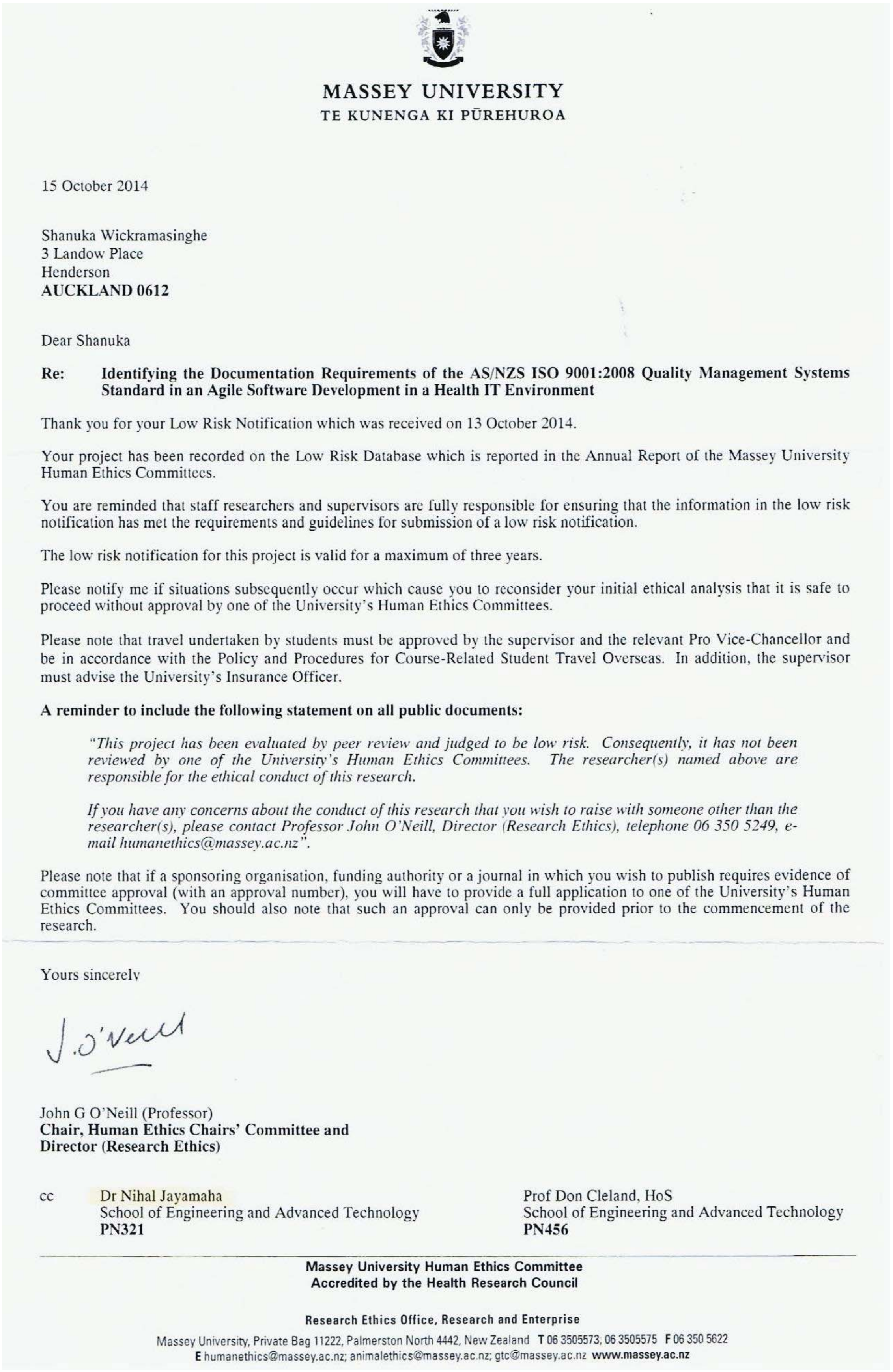
Table 4: Document Usage Information

DOCUMENT		**Is this document being created/used by your current project?**	**If 'No' the Reason/s (Only the code/s a, b, c, d, e)**
1	Maintained product backlog		
2	International Specifications		
3	Standards & Regulations		
4	Stakeholder/Customer requirements		
5	Business Flows		
6	Road Map Items (RMI)		
7	Features		
8	Test Verification Report		
9	Sprint demonstration recordings		
10	Retrospective notes		

11	Training materials		
12	Risk Analysis reports		
13	Technical Design		
14	DB designs		
15	Developer documentation		
16	Traceability Matrices		
17	Unit Test and test reports		
18	Source codes files		
19	Bug tickets		
20	Test Plans		
21	Bug Verification report		
22	User manuals		
23	Companywide Quality Procedure		

Appendix E

Massey University Human Ethics Low Risk Notice



MASSEY UNIVERSITY
TE KUNENGA KI PŪREHUROA

15 October 2014

Shanuka Wickramasinghe
3 Landow Place
Henderson
AUCKLAND 0612

Dear Shanuka

Re: Identifying the Documentation Requirements of the AS/NZS ISO 9001:2008 Quality Management Systems Standard in an Agile Software Development in a Health IT Environment

Thank you for your Low Risk Notification which was received on 13 October 2014.

Your project has been recorded on the Low Risk Database which is reported in the Annual Report of the Massey University Human Ethics Committees.

You are reminded that staff researchers and supervisors are fully responsible for ensuring that the information in the low risk notification has met the requirements and guidelines for submission of a low risk notification.

The low risk notification for this project is valid for a maximum of three years.

Please notify me if situations subsequently occur which cause you to reconsider your initial ethical analysis that it is safe to proceed without approval by one of the University's Human Ethics Committees.

Please note that travel undertaken by students must be approved by the supervisor and the relevant Pro Vice-Chancellor and be in accordance with the Policy and Procedures for Course-Related Student Travel Overseas. In addition, the supervisor must advise the University's Insurance Officer.

A reminder to include the following statement on all public documents:

"This project has been evaluated by peer review and judged to be low risk. Consequently, it has not been reviewed by one of the University's Human Ethics Committees. The researcher(s) named above are responsible for the ethical conduct of this research.

If you have any concerns about the conduct of this research that you wish to raise with someone other than the researcher(s), please contact Professor John O'Neill, Director (Research Ethics), telephone 06 350 5249, e-mail humanethics@massey.ac.nz".

Please note that if a sponsoring organisation, funding authority or a journal in which you wish to publish requires evidence of committee approval (with an approval number), you will have to provide a full application to one of the University's Human Ethics Committees. You should also note that such an approval can only be provided prior to the commencement of the research.

Yours sincerely

J. O'Neill

John G O'Neill (Professor)
Chair, Human Ethics Chairs' Committee and
Director (Research Ethics)

cc Dr Nihal Jayamaha
School of Engineering and Advanced Technology
PN321

Prof Don Cleland, HoS
School of Engineering and Advanced Technology
PN456

Massey University Human Ethics Committee
Accredited by the Health Research Council

Research Ethics Office, Research and Enterprise
Massey University, Private Bag 11222, Palmerston North 4442, New Zealand **T** 06 3505573; 06 3505575 **F** 06 350 5622
E humanethics@massey.ac.nz; animalethics@massey.ac.nz; gtc@massey.ac.nz **www.massey.ac.nz**