Summer 2010

Requirement Specification Stage of the Project Lifecycle of Computerized Systems & the Standards that Can Be Implemented

Nicola Grace
Regis University

Regis University
College for Professional Studies Graduate Programs
Final Project/Thesis

Disclaimer

Abstract

Understanding requirement specifications was an integral part of information systems design and was critical to the success of interactive systems. However, specifying these requirements was not simple to achieve. This research, including a literature review, describes general methods to support requirement specification analysis that can be adapted into a range of situations in accordance with relevant standards. The main techniques discussed were risk management, stage-based lifecycle models and frameworks. Additionally, as part of the methodology and project history, the methods for implementation, process improvements and schedule of the research was examined. A case study with statistical analysis was described to illustrate how these techniques, methods and standards have been applied in practice and the advantages and disadvantages experienced.

Acknowledgements

I would like to acknowledge with thanks everyone who helped and supported me in my studies. Especially, I would like to thank my parents for their support throughout this process. I would like to acknowledge my willing participants, for their patience and assistance. Finally, I would like to thank my thesis advisor, Mike Prasad, for all his guidance and support during this process, and my thesis facilitator, Don Ina, for his clear guidance throughout the thesis journey.

Table of Contents

List of Figures

List of Tables

Chapter 1 – Introduction

Requirements and specifications are very important components in the development of any information system. Requirement specification analysis is the first step in the system design process, where user requirements should be clarified and documented to generate the corresponding specifications. Therefore, if this area of the project lifecycle is neglected the project will suffer severely in future phases of the project lifecycle, because 60% of errors originate within the requirements activity. Consequently, this thesis will research and examine the requirement specification gathering procedures, in combination with relevant standards.

1.1 Background Information

In the early 1970's, according to Brooks, F. (1995), ~~"t~~he hardest single part of building a software system is deciding precisely what to build. No other part of the conceptual work is as difficult as establishing the detailed technical requirements, including all the interfaces to people, to machines and to other software systems. No other part of the work cripples the resulting systems if done wrongly, and no other part is more difficult to rectify later. Therefore the most important function that software builders do for their clients is the iterative extraction and refinement of the product requirements."

The most promising of the current technological efforts are the development of approaches, and tools for rapid prototyping of systems as part of the iterative specification of requirements. A prototype software system is one that simulates the important interfaces, and performs the main functions of the intended system, while not being necessarily bound by the same hardware speed, size, or cost constraints. To examine this topic further, in the present-day software specification, acquisition procedures rest upon the assumption that one must design a satisfactory system in advance, get bids for its construction, have it built and install it. This assumption is fundamentally wrong, and many software acquisitions problems spring

from fallacy. Hence it cannot be fixed without fundamental changes within the development and specification of documents and procedures, and hence the reason for proceeding with this work within the thesis topic.

1.2 Scope of work and thesis significance.

The main area of significance is to demonstrate the errors and problems caused by neglecting requirement specifications. While it is a common tendency for designers to be anxious about starting the design and implementation, discussing requirements with the customer is vital in the construction of systems. Activities in this first stage have significant impact on the downstream results in the system lifecycle. For example, errors developed during the requirements and specifications stage may lead to errors in the design stage. When this error is discovered, the engineers must revisit the requirements, and the specifications required to fix the problem. This leads not only to more time wasted, but also to the possibility of other requirements and specification errors. Therefore, the understanding, capturing and documenting of user requirements is a vital part of information systems design, and is critical to the success of interactive systems.

It is now widely understood that successful systems and products begin with an understanding of the needs and requirements of the users. As specified in the IEEE 15288, IEEE12207, and IEEE 1233 standards, user requirement specification gathering and documenting begins with a thorough understanding of the needs and requirements of the users. The benefits can include increased productivity, enhanced quality of work, reductions in support and training costs, and improved user satisfaction. However, requirement analysis is not a simple process. Particular problems faced by the analyst are:

1. Addressing complex organisational situations with many stakeholders.
2. Users and designers thinking along traditional lines, reflecting the current system and processes, rather than being innovative.

3. Users not knowing in advance what they want from the future system.
4. Rapid development cycles, reducing the time available for user needs analysis.
5. Representing user requirements in an appropriate form.

This research considers how these problems can be addressed, by selecting appropriate methods and standards, to support the process of user requirements gathering and documenting. The main tools to be used to aid in this process are risk management, stage-based lifecycle models and frameworks. The research, using a case study, will also describe and suggest standards and methods, which demonstrate how each contributes to the requirements process. The main aim of the research is to demonstrate the importance of a requirements document for a system, as it defines a set of acceptable implementations of that system, including any implementation constraints. Such a document should be clear, precise, easy to modify, and easy to check for completeness and consistency, in accordance with international and industrial standards. When properly written, a requirements document has many uses: as a contract between buyer and seller, that specifies what is to be built; as a metric that management can use to measure progress; as a standard for determining the correctness of an implementation; and as a guide that those developers can use to formulate and evaluate various design and implementation alternatives.

The poor quality of most real world requirement documents seriously reduces their usefulness, due to many companies neglecting the requirement specification stage of the project lifecycle. Other authors have emphasised the ambiguity, imprecision, and inconsistency that one often finds in such documents. This research focuses on these issues and also on insufficient importance being placed on the requirement specification stage of the project lifecycle. It also examines an additional problem of how many requirement documents say too little about what to do, and too much about how to do it. As a result,

designers are constrained by poor design decisions that have been built into the requirements or forced to collect essential system features from a mass of extraneous detail.

1.3 Acknowledgment of previous work

The main work to date on this topic is split between the two general areas of the requirement specification stage of the project lifecycle in software engineering and the general standards applicable to software engineering. The research in relation to requirement specification dates from late 1970‘s to the present.

The area of requirement specification in project lifecycle was researched and written about in detail from the early the 1970s to 1995. This research was related to the following main factors for requirement specification:

1. An approach to producing abstract requirement specifications that applies to a significant class of real-world systems.

2. Methodologies for requirement discovery and organisation according to competence areas available on the project and involved risks.

3. Methodology for managing change of' requirements during the software development project to enable fruitful project conclusion even if major requirement changes occur.

4. Writing requirement specifications for process-control systems.

The research on standards from 1980 to 1995 is mainly on to the following topics.

Importance of standards, according to Walker (1998), –the internal quality systems audits are compliance requirements of the ISO 9001, but there is little common approach to auditing of a company against the compliance requirements, other than that which may be traced to ISO 9001 which is a basic standard”. The main conclusion drawn from this area is that there are standards in place but not implemented in a correct or efficient manner.

The software engineering project standards and their importance was also discussed in 1982 by Branstad & Powell, in relation to progression of standards through the project lifecycle and their relevance.

The research and papers dating for the past five years in relation to standards within the software engineering community again express the issues in relation to current and up-to-date standards being accessible to engineers within the industry. One paper of interest discusses the ~~"in~~integration of software lifecycle process standards with engineering activities for security purposes" (Lee, Y., Lee, J. & Lee, Z. 2002, p. 350). Another issue is the delays in implementation of standards, according to the Software Engineering Body of Knowledge (SWEBOK) conference of 1992, ~~"th~~e introduction and constant reviewing and updating of standards were of hindrance to the industry" (Bourque & Lethbridge, 2002, p.1).

The research in the past five years in relation to the requirement specifications is:

1. Methods and procedures for gathering requirement specification and managing according to Carew, D., Exton, C. and Buckley, J. (2005).

2. Risk management in relation to requirement specification according to Glinz, M. (2008).

3. Lifecycle frameworks for requirement specification according to Pozagj, Z., Sertie, H. and Boban, M. (2003).

1.4 Road Map of thesis

The thesis research is structured as follows: Chapter 1 is the introduction to the thesis topic and scope, Chapter 2 describes the literature review for the requirement specifications stage of the project lifecycle and the relevant standards required. This chapter also demonstrates techniques for requirement specification, including risk management, stage-based lifecycle model, and requirement specification stages frameworks. The chapter concludes by discussing standards, and requirement specification stages, in combination.

Chapter 3 presents the methodology that drove the analysis of the data collection. Chapter 4 presents the results and analysis via figures, tables and statistics. Chapter 5 discusses the project history and finally, Chapter 6 concludes the research. The thesis also contains a number of appendices including Appendix A which is the annotated bibliography. Appendix B which is an example of the questionnaire distributed to the participants. Appendix C is the history of the project plan and a glossary.

1.5 Beginning of thesis project workload

The main aim of the thesis is to prove that requirement specification documenting is a neglected phase of the project lifecycle. This thesis will discuss how to discover the requirements, and determine their precise nature, and use these requirements in combination with standards to contribute largely to the project lifecycle, and aid in proceeding through the project lifecycle of a component or system. This thesis will present a set of techniques for gathering, confirming, organising, and documenting the requirements for a product. It also discusses how you can come to an understanding of the requirements, and how you might write them down so that the constructors and the future generations of maintenance people can understand them.

The main aim of the thesis is to demonstrate the importance of requirement specifications throughout the project lifecycle, and the benefits and advantages achieved by implementing this phase in combination with relevant and reflective standards.

Chapter 2 – Review of Literature and Research

2.1 General Introduction

The correct requirements come from understanding the work that the products are intended to support. Only when you know the correct requirements can you design and build the correct product, which in turn enables the product‘s users to do their work in a way that satisfies their business needs. The cost of good requirement gathering and system analysis is minor compared to the cost of poor requirements.

Sadly, the requirements are not always correctly understood. Jackson (1995) provides statistics that show that as many as ~~–6~~0 percent of errors originate with the requirements activity”. Clearly, although developers of products have the opportunity to eliminate a large category of errors they choose, or even worse their manager chooses, to rush headlong into constructing the wrong products. As a result, they pay many times the price for the product that they would have if the requirements and analysis had been done correctly in the first place. Poor quality is passed on.

By implementing relevant standards to the requirements phase of a project, the cost and poor quality can be significatly reduced. The main aim of this thesis is to demonsrrate this point. The thesis topic was further decided on by researching current topics in the industry via papers, internet, university courses and postgraduate opportunities. A lot of the relevant papers date from two decades ago and include the following:

1. Loucopoulos & Champion (1990) discuss ~~–s~~oftware development activities and argues that requirement specification is the most critical of all software development activities.” The paper further proposes an approach of system development methods which address the task in the context of regulatory approaches.

2. Jang (1994) discusses the ~~–f~~formal requirement specification language for requirement specification analysis". The main conclusion of the paper was to emphasise the importance of knowledge based analysis for requirement specification analysis.

3. Yau and Liu (1988) examine ~~–a~~n approach to software requirement specification using a structure inheritance network."

4. Yau, Bae and Yeon (1994) discuss the ~~–f~~act that errors in source code can be traced back to errors in requirement specifications." The paper examines an approach to allow verification of requirement specifications in software development for distributed computing systems. The main aim of the paper is to demonstrate the use of formal specification languages. The main conclusion drawn from the paper is that ambiguities in the requirement specification statements may lead to different interpretations of the software system, thus making verification more difficult.

5. The last paper of significance during the research of the thesis topic in relation to requirement specification was Hunt (1997) which describes ~~–a~~ radically new approach to producing re-usable requirement specification which achieves levels of clarity and precision hitherto unattainable". The paper further discusses the problem area in requirement specification. The author then proceeds to methodology, including the general basic framework, evolution, real application areas and finally the conclusion. The main conclusion and aim of the paper was to draw attention to the fact that it is possible to considerably improve the quality of a requirement specification prior to the development of the system by designers and software engineers.

The main paper to be fully applicable to my thesis topic from the beginning was Hesslink (1995) as the paper discusses ~~–t~~he standard and its relevance to requirement specification within the project lifecycle, project documentation, configuration management, verification and validation and quality assurance." The author accesses a number of

standards (4) and their relevance to requirement specification. He further discusses the standards and their application to project lifecycle. He briefly discussed relevant standards and requirement specifications, but few papers discussed requirement specification and relevant standards in combination.

Therefore the thesis research examined the combination of requirement specification and standards in combination, the main current papers supporting these objectives were:

1. Glinz (2008) discussed the quality requirements and their availability to stakeholders. The main aim of the author was to demonstrate the three kinds of problems experienced during requirement specification. The main conclusion found was the importance of supplier organisations and customers to communicate during requirement specifications.

2. The second paper of relevance to this time period and topic was Aliport & Isazadeh (2008) who discussed ~~t~~he changes experienced in the gathering of requirement specifications in the past three decades."

3. The final paper of relevance during this time period was Glass (2009) which discussed ~~t~~he importance of a standard for requirement documents." This supported my belief in the importance of requirement specification and standards in combination.

2.2 Standards

The main standards of interest to this thesis research are as follows:

1 IEEE STD 15288™-2008, Systems and software engineering —System lifecycle processes

2 IEEE STD 12207™-2008, Systems and software engineering —System lifecycle processes.

3 IEEE STD 1233™-1998, IEEE Guide for Developing System Requirement Specifications.

4 European Space Agency (ESA) Board for Software Standardisation and Control (BSSC), 1991, ESA Software Engineering Standards

The major standards are IEEE STD 15288 and IEEE STD 12207, as both discuss the system and software engineering principles, including design and requirement generation. IEEE STD 1233 is specific to the requirement specification which was very relevant to the thesis, as it discussed the specific requirements which was a major part of the research performed.

2.2.1 IEEE standards

The IEEE STD 15288™-2008 (Systems and software engineering —System lifecycle processes) establishes ~~–a~~ common framework for describing the lifecycle of systems created by humans." It defines a set of processes and associated terminology. These processes can be applied at any level in the hierarchy of a system's structure. Selected sets of these processes can be applied throughout the lifecycle for managing and performing the stages of a system's lifecycle. This International Standard also provides processes that support the definition, control and improvement of the lifecycle processes used within an organisation, or of a project. The standard concerns those systems that are man-made and may be configured with one or more of the following: hardware, software, data, humans, processes (e.g. processes for providing services to users), procedures (e.g. operator instructions), facilities, materials and naturally occurring entities.

This standard is applicable to the thesis as it discusses in detail the lifecycle concepts process models as per Section 5 and Section 6 the System Lifecycle Processes. Annex E is of major influence to the thesis research as it discusses the process alignment of ISO/IEC 15288 and ISO/IEC 12207. The alignment of the processes is straightforward as both standards use the same process names and same clause number for the individual processes. The two standards use slightly different names for these processes. In some cases, the process in

ISO/IEC 12207 is a software specialisation of the process in ISO/IEC 15288. —In other cases, the process in ISO/IEC 12207 merely contributes to the achievement of one or more outcomes of the corresponding process in ISO/IEC 15288. —(Singh, 2000, p. 10)

IEEE STD 12207 establishes a common framework for software lifecycle processes, with well defined terminology, that can be referenced by the software industry. It contains processes, activities and tasks that are to be applied during the acquisition of a software product or service and during the supply, development, operation, maintenance and disposal of software products. Software includes the software portion of firmware. This International Standard applies to the acquisition of systems and software products and services, to the supply, development, operation, maintenance and disposal of software products and the software portion of a system, whether performed internally or externally to an organisation. Those aspects of system definition needed to provide the context for software products and services are included.

> This International Standard also provides a process that can be employed for defining, controlling, and improving software lifecycle processes. The processes, activities and tasks of this International Standard, either alone or in conjunction with ISO/IEC 15288, may also be applied during the acquisition of a system that contains software (IEEE STD 12207™, 2008, p. 75).

The purpose of this standard is to provide a defined set of processes to facilitate communication among acquirers, suppliers and other stakeholders in the lifecycle of a software product. The standard is written for acquirers of systems and software products, and services and for suppliers, developers, operators, maintainers, managers, quality assurance managers and users of software products. This international standard is intended for use in a two-party situation and may be equally applied where the two parties are from the same organisation. The situation may range from an informal agreement up to a legally binding

contract. The standard may be used by a single party through a self-imposed set of processes. Section 6 of the standard is very relevant to the thesis research as it discusses the System Lifecycle Processes. Annex D is of relevance also as it compares and contrasts the ISO/IEC 12207 with ISO/IEC 15288 and their alignment. This process is straightforward as both standards use the same process names and same clause numbers for the individual processes.

> IEEE STD 1233™-1998, IEEE Guide for Developing System Requirements contains specifications which provide a guide for the development of a set of requirements that, when realised, will satisfy an expressed need. In the guide the set of requirements are called the System Requirement Specification (SyRS). Developing a SyRS includes the identification, organisation, presentation, and modification of the requirements. The guide addresses conditions for incorporating operational concepts, design constraints and design configuration requirements into the specification. The guide also addresses the necessary characteristics and qualities of individual requirements. The guide does not specify industry-wide system specification standards or state a mandatory System Requirement Specification. The guide is written under the premise that the current state of the art of system development does not warrant or support a formal standards document (IEEE STD 1233™, 1998, p. 25).

The main areas of interest to the thesis in this document are in Section 4 which discusses a System Requirement Specification (SyRS) that has traditionally been viewed as a document that communicates the requirements of the customer to the technical community, who will specify and build the system. The collection of requirements that constitutes the specification and its representation acts as the bridge between the two groups and must be understood by both the customer and the technical community.

> One of the most difficult tasks in the creation of a system is that of communicating to all of the subgroups within the customer and technical groups, especially in one

document. This type of communication generally requires different formalisms and languages. The standards future discusses in detail the well-formed requirements as part of section 6 including the definition of a well-formed requirement, properties of a requirement, categorisation, and finally pitfalls of requirements. As part of section 7 of the standard an explanation of SyRS development is examined, including identifying requirements techniques, and building well-formed requirements. Organising the requirements and presenting the requirements (IEEE STD 1233™, 1998, p. 40).

The European Space Agency (ESA) Board for Software Standardisation and Control (BSSC) document describes the software engineering standards to be applied for all deliverable software implemented for the ESA, either in house or by industry.

Software is defined in these standards as the programs, procedures, rules and all associated documentation, pertaining to the operation of a computerised system. These standards are concerned with all software aspects of a system, including its interfaces with the computer hardware and with other components of the system. Software may be either a subsystem of a more complex system or it may be an independent system. The software projects vary widely in purpose, size, complexity and availability of resources discussed in this standard. Software project management should define how the standards are to be applied in the planning documents. The main factors in applying standards are the project cost, both in development and operation, number of people required to develop, operate and maintain the software; number of potential users of the software; amount of software that has to be produced; criticality of the software, as measured by the consequences of its failure; complexity of the software, as measured by the number of interfaces or a similar metric; completeness and stability of the user requirements; and risk values included with the

user requirements (ESA Board for Software Standardisation and Control (BSSC), 1991, p. 156).

The standards are broken down into three parts,

1. Discussing the project standards
2. Procedure standards
3. Appendices.

The first part of the standard product standards is of relevance to this thesis research, as it discusses the software lifecycle including the phase‘s activities and milestones, lifecycle approaches, prototyping and handling requirement change. The second chapter discusses the user requirement definition phase including the inputs to the phase, activities and outputs from the phase. Chapter 3 is the major chapter of relevance, as it discusses software-requirement definition phase including the introduction, input to the phase, activities and outputs from the phase.

2.2.2 IEEE Minor standards

IEEE has several minor standards related to system lifecycle processes:

1 IEEE STD 1220™-2005, IEEE Standard for Application and Management of the Systems Engineering Process.

2 IEEE STD 1228™-1994, IEEE Standard for Software Safety Plans.

3 IEEE STD 1362™-1998, IEEE Guide for Information Technology—System Definition—Concept of Operations (ConOps) Document.

4 IEEE STD 1471™-2000, IEEE Recommended Practice for Architectural Description for Software-Intensive Systems.

5 IEEE STD 1074™-2006, IEEE Standard for Developing a Software Project Lifecycle Process Relationship

6 IEEE STD 1517™-1999, IEEE Standards for Information Technology – Software Lifecycle Processes – Reuse Processes.

7 IEEE STD 830™-1998, IEEE Recommended Practice for Software Requirement Specifications.

8 IEEE STD 1540™-2001, IEEE Standards for Software Lifecycle Processes – Risk Management.

9 IEEE STD 1012™-1998, IEEE Standard for Software Verification and Validation

10 IEEE STD 1016™-2009, IEEE Standard for Information Technology— Systems Design— Software Design Descriptions

The IEEE STD 1220™-2005, IEEE Standard for Application and Management of the Systems Engineering Process**,** has maintained its own standard for the "Systems Engineering Process" (SEP), a phrase not used in ISO/IEC 15288. IEEE STD 1220-2005, Standard for the Application and Management of the Systems Engineering Process, has the following abstract:

> The interdisciplinary tasks which are required throughout a system's lifecycle to transform customer needs, requirements, and constraints into a system solution are defined. In addition, the requirements for the systems engineering process and its application throughout the product lifecycle are specified. The focus of this standard is on engineering activities necessary to guide product development while ensuring that the product is properly designed to make it affordable to produce, own, operate, maintain, and eventually to dispose of, without undue risk to health or the environment (IEEE STD 1220™, 2005,p.5).

Explaining the relationship between the thesis and IEEE STD 1220 required considering both the lifecycle processes and lifecycle stages provided. The IEEE STD 1220 focuses on the development of a system, including making plans and providing processes to

deal with the remainder of the system's life. IEEE STD 1220 provides requirements for an integrated technical approach to defining and developing system products.

> The Software Safety Plan exists within a more general system-wide safety program. In particular, the Software Safety Plan provides for safety analyses preparation when the system is designed. In describing the software safety plan, the standard places implicit requirements on the activities of the software development. Other IEEE standards are cited where appropriate (IEEE STD 1228™, 1994, p.3).
>
> The IEEE STD 1228 has little relevance to the research but is worth noting.
>
> IEEE has a standard that may be useful in achieving these outcomes: IEEE STD 1362-1998, IEEE Guide for Information Technology—System Definition—Concept of Operations (ConOps) Document. Its abstract states: that the standard ~~p~~provides a guide to the content of a Concept of Operations document as well as guidance in developing the document and using requirement specification documentation techniques." IEEE STD 1362 works from the assumption that a new system is replacing an existing one of some sort. So, the ConOps document is intended to describe an existing system, its changes, and the new system from the point of view of the user. ~~I~~it provides a place to describe user needs without being overly technical or overly quantitative, so that end-users can participate in the approval of the concept." (IEEE STD 1362™, 1998, p.3). This standard is only of minor interest in relation to the thesis topic.

IEEE STD 1471™-2000, IEEE Recommended Practice for Architectural Description for Software-Intensive Systems has a ~~r~~recommended practice for the characteristics of an architectural description which in turn affects the requirement specification gathering and generation, and requests the requirement specifications to be considered during the architectural description and decision making."A central idea of the standard is that the

description of architecture should be expressed by describing multiple views, each governed by a defined viewpoint to deal with various concerns of stakeholders which are documented by the requirement specifications. The standard does not provide the viewpoints; they should be selected based on the needs of the system.

IEEE STD 1074™-2006, IEEE Standard for Developing a Software Project Life provides a process for creating a software project lifecycle process (SPLCP). It is primarily directed at the process architect for a given software project. It is the function of the process architect to develop the SPLCP, as requirement specification is generated during the SPLCP phase of the project lifecycle. It is important for personnel to fully understand the SPLCP and the effects that requirement specification has on the overall SPLCP.

> This methodology begins with the selection of an appropriate software project lifecycle model (SPLCM) for use on the specific project. It continues through the definition of the software project lifecycle (SPLC), using the selected SPLCM, and the portion of the software lifecycle that is relevant to the project. The methodology concludes with the augmentation of the software lifecycle with organisational process assets (OPAs) to create the SPLCP. This standard does not address non-software activities, such as contracting, purchasing, or hardware development (IEEE STD 1074™, 2006, p. 13).

The procedure for gathering and analysing the requirement specification are examined and examples given as part of Annex 1 to this standard. IEEE STD 1517™-1999, Standards for Information Technology – Software Lifecycle Processes – Reuse Processes provides a common framework for extending the software lifecycle processes to include the systematic practice of software reuse and is of relevance to the thesis research. It specifies the processes, activities, and tasks to be applied during each phase of a software lifecycle to enable a software product to be constructed from assets. This standard also specifies the processes,

activities, and tasks to enable the identification, construction, maintenance, and management of assets.

The main area of this standard that is applicable to the thesis research is examined in Section 5 of the standard integration of reuse into the primary lifecycle process, and the implementation of requirement specification documenting at this stage of the project.

IEEE STD 830™-1998, IEEE Recommended Practice for Software Requirement, describes recommended approaches for the specification of software requirements. It is divided into five clauses.

1 Clause 1 explains the scope of this recommended practice.

2 Clause 2 lists the references made to other standards.

3 Clause 3 provides definitions of specific terms used.

4 Clause 4 provides background information for writing a good Software Requirement Specification (SRS).

5 Clause 5 discusses each of the essential parts of an SRS.

This recommended practice also has two annexes, one which provides alternate format templates, and one which provides guidelines for compliance with IEEE/EIA 12207.1-1997. This is a recommended practice for writing software requirement specifications. It describes the content and qualities of a good software requirement specification (SRS) and presents several sample SRS outlines. —This recommended practice is aimed at specifying requirements of software to be developed but also can be applied to assist in the selection of in-house and commercial software products. This recommended practice describes the process of creating a product and the content of the product. The product is an SRS" (IEEE STD 830™, 1998, p. 2).

This recommended practice can be used to create such an SRS directly, or can be used as a model for a more specific standard and is of relevance to thesis topic and general methodology in relation to requirement specifications.

IEEE STD 1540™-2001, IEEE Standards for Software Lifecycle Processes – Risk Management prescribes a continuous process for software risk management. Clause 1 provides an overview and describes the purpose, scope, and field of application, as well as prescribing the conformance criteria. Clause 2 lists the normative references; informative references are provided in Annex E. Clause 3 provides definitions. Clause 4 describes how risk management may be applied to the software lifecycle. Clause 5 prescribes the requirements for a risk management process. There are several informative annexes. Annex A, Annex B, and Annex C recommends three documents: Risk Management Plan, Risk Action Request, and Risk Treatment Plan. Annex D summarises where risk management is mentioned in the IEEE/EIA 12207 series of software lifecycle process standards.

> This standard describes a process for the management of risk during software acquisition, supply, development, operations, and maintenance. It is intended that both technical and managerial personnel throughout an organisation apply this standard. The purpose of this standard is to provide software suppliers, acquirers, developers, and managers with a single set of process requirements suitable for the management of a broad variety of risks. This standard does not provide detailed risk management techniques, but instead focuses on defining a process for risk management in which any of several techniques may be applied, and is therefore relevant to requirement specification and the introduction of risk management into this section of the project lifecycle as this is the manner in which future projects are to be performed (IEEE STD 1540™, 2001, p. 7).

IEEE STD 1012™-1998, IEEE Standard for Software Verification and Validation provides the Verification and Validation (V&V) standard and is a process standard that addresses all software lifecycle processes including acquisition, supply, development, operation, and maintenance. The standard is compatible with all lifecycle models; however, not all lifecycle models use all of the lifecycle processes listed in this standard. Software V&V processes determines whether the development product of a given activity conforms to the requirements of that activity and whether the software satisfies its intended use and user needs. The determination may include analysis, evaluation, review, inspection, assessment, and testing of software products and processes. Software V&V processes consists of the verification process and the validation process. The verification process provides objective evidence as to whether the software and its associated products and processes

1 Conform to requirements for all lifecycle activities during each lifecycle process.

2 Satisfy standards, practices, and conventions during lifecycle processes

3 Successfully complete each lifecycle activity and satisfy all the criteria for initiating succeeding lifecycle activities (IEEE STD 1012™, 1998).

The validation and verification occur at a later date in the project lifecycle but the requirement specifications that have been documented contribute largely to this section of the project and in turn if the requirment specification contain errors, the validation and verification section will incur errors. This is relevant to the thesis research as it demonstrates the importance of requirements when trying to eliminate errors.

IEEE STD 1016™-2009, IEEE Standard for Information Technology—Systems Design— Software Design Descriptions, describes software designs and establishes the information content and organisation of a software design description (SDD). An SDD is a representation of a software design to be used for recording design

information and communicating that design information to key design stakeholders. The standard is intended for use in design situations in which an explicit SDD is to be prepared. The standard can be applied to commercial, scientific, or military software that runs on digital computers. Applicability is not restricted by the size, complexity, or criticality of the software. This standard can be applied to the description of high-level and detailed designs. This standard is applicable to the thesis research as in Section 3 of the standard it discusses a conceptual model for software design descriptions (IEEE Std 1016™, 2009, p. 13).

To conclude on standards, 15288 are applicable to this stage of the requirement specification as it discusses the process lifecycle for hardware, process and data. The IEEE 12207 is applicable to the process lifecycle for software and discusses in detail the software lifecycle and how to implement the standard requirements during the requirement specification stage of the project lifecycle. The most relevant standard is IEEE 1233, as this guides the development of a set of requirements for the system. This guide includes the identification, organisation, presentation and modifications of requirements. The guide also addresses conditions for incorporating operational concepts, design concerns, and design configuration requirements for the specifications. This standard is also used in the collection of requirements which constitute the specification and represents the bridge between the vendor and the customer. The ESA standard is relevant to this thesis research as it discusses the software lifecycle and defines the user requirement phase and software requirement phase. In relation to the software requirement definition phase, the standard introduces the phase and discusses the activities related to this phase and the outputs formed from the phase including the software requirement document, system test plan, project management plan, configuration management plan, verification and validation and finally the quality assurance plan.

In conclusion, the International Standard on System Lifecycle Processes, ISO/IEC 15288, defines a top-level cradle-to-grave lifecycle framework for managing modern systems configured with hardware, computers, software and humans.

ISO/IEC /SC7 WG7 are responsible for developing ISO/IEC 15288. As summarised by the Convenor of Working Group (WG) 7, ~~t~~he Standard:

a. Can be applied to the acquisition, supply, development, operation and maintenance of systems.

b. Supports the above through configuration management and quality assurance.

c. Can be used as an internal framework by an enterprise.

d. Can be used in developing an agreement between two parties, or as a reference standard for lifecycle processes for further standardisation, guidance and tools development" (Harauz & Poon, 1999, p. 1).

The IEEE Computer Society and other standard organisations are doing much to promote a disciplined approach to software engineering practice and how aerospace systems engineering organisations can apply these standards to their projects. ~~—~~The presentation also highlights the software standardisation activities of national and international standards organisations (American National Standards Institute (ANSI), ISO, International Electro – technical commission (IEC), government organisations (National Bureau of standards (NBS), DoD), professional societies (IEEE, ASQC, Association of Computing Machinery (ACM)) and trade associations (Aerospace Industries Association (AIA), Electronic Industries Association (EIA), and National Standards of Authority Ireland (NSIA))." (Marriott & Siefert, 1992, p. 1)

2.2.3 Industrial standards

After discussing ISO /ANSI standards, another industry standard of interest is DO-178B which provides guidelines for software certification. This standard addresses the use

of emerging software as an aid to reducing the lifecycle costs. However, to re-use requires re-certification or certification of software that was not developed according to DO-178B.

The main areas of interest discussed by Hesslink (1995) are that the investigation assists in understanding the rationale behind several standards that can be used for certification according to DO-178B of software which was developed using another standard and a comparison of each of the examined topics has been made. Also in this paper the software lifecycles, documentation, and certification matters were compared" (Hesslink1995, p. 3).

The main problem areas with emerging standards and guidelines are that they need to be timely, and reflect the requirements of the industrial sector that they are designed to support. Therefore, the standards process fails if this delay results in out-of-date standards, or standards that are not useful (e.g. if the production of standards cannot keep up with the rate of technological advance.)

One answer to this problem is *de jure* standards which are developed by formal standard development organisations that are recognised by the International Organisation for Standards (ISO). These standards have been produced in a way that is open to input from the public, though to make such input a person is typically required to attend meetings around the world at the person‘s own expense. Another attribute of *de jure* standards is that they must be maintained. In the case of ISO, maintenance has typically meant that the standard is updated exactly once every 5 years.

> Typically, for *de jure* standards the standards development organisation claims copyright of the standard and sells the standard. Governments realise the importance of standards but sometimes want standards that are developed to be more flexible. Accordingly, an agency of the government will fund organisations to develop certain standards. For instance, the United States Department of Defence funded the

Carnegie-Mellon University Software Engineering Institute to develop and maintain the Capability Maturity Model of software. Such standard development efforts typically attempt to achieve consensus but may determine their own rules of participation in the consensus process. The resultant standards may be distributed for free because the government agency wants to maximise access to the standard" (Rada, 2001, p10).

Standards should be based on scientific results and best industrial practice. They should be subject to evaluation to ensure they really work in the environment for which they are intended. All this is difficult because standards tend to be produced in a highly politicised environment in which corporate economic needs and cultures can take precedence over usefulness. ~~I~~It can be seen that all of this adds up to one thing, we need change. Both the process and nature of the software standards demand objective review." (Glass, 2009, p. 2).

However, within the last 20 years the software industry has been looking for ways to reduce the cost of developing software while at the same time improving its quality and reliability. ~~O~~One approach to answering industry's needs has been through the development and use of the IEEE standards that establish the norms of professional practices in the field of software engineering. Most if not all of the variable software engineering standards in the world today are developed through the Institute of Electrical and Electronics Engineers (IEEE) Computer Society" (Marriott & Siefert, 1992, p. 1).

The harmonisation of professional standards usually means an attempt to unify the standards among different nations and states. The main obstacle to this development is that the ~~s~~software engineering profession has not reached a mature stage of development" (Tse, 2000, p. 347).

To emphasise the importance of requirement specification and standards in combination, and the improvement that can be demonstrated, the need for a standard is assessed in the next section of the literature review chapter.

2.3 Introduction to requirement specifications

A requirement is defined as ―a condition or capability that must be met or possessed by a system, product, service, result or component to satisfy a contract, standard, specification or other formally imposed documents.” (Jackson, 1995)

Requirements include the quantified and documented needs, wants and expectations of the sponsor, customer and other stakeholders. A requirement specification is a defined goal that details the desired end result of a project. This helps project managers to ensure that they are delivering exactly what their customers want, because it is specified and communicated clearly to those involved in the project.

A requirement is something the product must do or quality it must have. A requirement exists either because the type of product demands certain functions / qualities or because the client wants the requirements to be part of the delivered product. A functional requirement is an action that the product must take if it is to be useful to its users.

> Functional requirements arise from the work that the stakeholders need to do. Almost any action, for example calculate, inspect, publish or most other active verbs, can be a functional requirement. This requirement is something that the product must do if it is to be useful within the context of the customer‘s business. Non-functional requirements are properties or qualities that the product must have. In some cases, non-functional requirements describe such properties as look and feel, usability, security and legal requirements and are critical to the product‘s success (Robertson & Robertson, 2008).

Since software requirements are textual descriptions of customer demands they can be created in different ways. The main problem connected with software requirement descriptions is the content of the description, and not the form of the description. The discovery of software system‘s requirements is a long and complicated process that must be considered extremely important for developing successful software solutions. However, formal definition of a template for software requirement description is an important precondition for efficient requirement management, because software requirements must be described in a form that is easy to understand and use. Software requirements should be used in all phases of software development in order to provide guidelines for development activities.

2.3.1 Techniques for requirement specification generation

The main techniques used for requirement gathering are use case, benchmarking and models / diagrams. A business use case is used when the functionality of the work needs to be examined to aid with a business event. A requirements analyst is assigned to each of the business use cases for further detailed study. The analysts use techniques such as benchmarking, models and use cases workshop among many others, to discover the true nature of the work. Once they understand the requirements, analysts work with the stakeholders to decide the best product to help with this work. That is, they determine how much of the work to automate or change and what effect these decisions will have on the work. Once they know the extent of the product the requirements analyst write their requirements. At the beginning of a project the requirements analyst is concerned with understanding the business the product is intended to support. At this stage, the analyst is working with scenarios and other models to help him, and the stakeholders come to an agreement on what the work is to be.

As the understanding of the work progresses, the stakeholders decide on the optimal product to help with the work. Now the requirement analysts start to determine the detailed functionality for the product and write its requirements. The non-functional requirements are derived around the same time and written along with the constraints. At this point, the requirements are written in a technologically neutral manner and they specify what the product is to do for the work, not what terminology is used to do it (Jackson, 1995).

Further to this, an abstract requirement specification is developed that states system requirements precisely without describing a real or a paradigm implementation. Although such specifications have important advantages, they are difficult to produce for complex systems and hence are seldom seen in the "real" programming world.

In summary, a requirements document for a system defines the set of acceptable implementations of that system, including any implementation constraints. Such a document should be clear, precise, easy to modify, and easy to check for completeness and consistency. When properly written, a requirements document has many uses:

1. As a contract between buyer and seller that specifies what is to be built.
2. As a metric that management can use to measure progress.
3. As a standard for determining the correctness of an implementation.
4. As a guide that developers can use to formulate and evaluate various design and implementation alternatives.

The standard bodies of modern day now understand the requirements document and the how users use the document.

2.3.2 Managing requirement specification generation

Requirements management involves working with a defined set of product requirements throughout the product‘s development process and its operational life. It also

includes managing changes to that set of requirements throughout the project lifecycle. In practice, requirements management includes selecting changes to be incorporated within a particular release and ensuring effective implementation of changes with no adverse impact on schedule, scope or quality. An effective requirements definition and management solution creates an accurate and complete system requirements, while helping organisations improve communications in an effort to better align IT with business needs and objectives. It includes a set of industry best practices for each category, as well as tools to enable and accelerate requirements activities. Therefore, this is a document that needs to be clear and precise. The poor quality of most real-world requirements documents seriously reduces their usefulness.

Other authors have emphasised the ambiguity, imprecision, and inconsistency that one often finds in such documents.

> The paper focuses on an additional but more fundamental problem: many requirements documents say too little about what to do and too much about how to do it. Their bulk consists of decisions that should have been postponed until design time, instead of discussing precise information about the system's external behaviour. As a result, designers are constrained by poor design decisions that have been built into the requirements or forced to collect essential system features from a mass of extraneous detail (Heitmeyer and McLean 1983, p. 1).

The number of requirements derived at any step is not important although experience shows that it is usually fewer than six. If only one requirement is uncovered from each step, it suggests either the level of detail in the scenario is not granular or the functional requirements are too coarse. If more than six requirements per step are achieved, either the requirements are too granular or have a very complex use case. The objective is to discover enough functional requirements for the developers to build the precise product that the client is expecting and the actor needs to do the work.

When examining functional requirements, it is advised to group them by use case. The advantage achieved by doing so is that it becomes easy to discover related groups of requirements and to test the completeness of the functionality. Nevertheless, sometimes other groupings may prove more useful. Non-functional requirements do not alter the product‘s essential functionality. That is, the functional requirements remain the same no matter which properties you attach to them. To confuse matters even more, the non-functional requirements might add functionality to the product. A suggestion is to think of the functional requirements as those that cause the product to do the work, and the non-functional requirements as those that cause the product to give character to the work.

> If the requirements are traceable, then when changes happen it is far easier to find the parts of the product affected by the changes and to assess the impact of the change on the rest of the product. In keeping the requirements traceable it means that they can be designed in a more effective way to allow change (Robertson & Robertson, 2008).

Most advocates of formal methods agree that such methods should be applied to systems where the issue of correctness is a priority. While safety-and security-critical systems fall into this category, there are a number of other systems which are not commonly classified in these terms, and could equally benefit from the application of formal methods. While formal methods are being applied to hardware in industry, the results of formal methods research for software have only rarely reached beyond the research lab and are used in industrial practice for day-to-day software development. ~~–O~~ne of the key factors associated with this widespread lack of adoption is that formal methods are seen as being difficult to comprehend” (Carew, Exton & Buckley, 2005, p. 1).

In particular, because many errors in the source code can be traced to the errors in the requirement specification, it is especially important to have effective verification techniques for managing requirement specification.

> The paper currently assumes that the requirements statements are unambiguous. Ambiguities in the requirements statements may lead to different interpretations of the software system, thus making the verification more difficult. The research is needed to deal with the verification of requirements statements containing ambiguities. In addition, by extending the information tree with hierarchical structure, the approach can be applied to the requirements verification of large-scale software for distributed computing systems (Yau, and et al, 1994, p. 7).

Within the dependable systems community there has been considerable interest and effort looking at improving the development process of managing requirements, especially the Requirements Engineering process according to Emmet and Bloomfield (1997). In particular, the importance of the human factors of development processes is increasingly recognised as being an important source of process weaknesses and improvement opportunities.

The standards process itself can be considered a requirement engineering process in which user requirements are captured, negotiated, analysed and defined. This development process hopefully results in a document that expresses the requirements of the industry and standards process participants.

> Like many requirement engineering processes it is an inherently socio-technical process, constructed from a number of integrated social and technical activities, including various document production, distribution and reviewing activities which are co-ordinated at the subgroup and individual levels; in short, a good candidate for a requirement engineering analysis (Emmet & Bloomfield, 1997, p. 208).

The work performed by Pozagi, Sertie and Boban (2003) is focused on presenting an effective means of requirement specification and methodologies that can be used in order to support and improve requirement management techniques on software development

projects". Therefore, with this work they propose methodologies for dealing with software requirements that should be used in order to develop successful software solutions. Since the conditions on today's global software market are rapidly changing, change of requirements during software development is a common occurrence.

> Consequently, they have also proposed methodology for dealing with requirement change that enables fruitful project conclusion even if major requirement change occurs. Presented methodologies are dedicated to provide effective requirement management practice and to support descriptions of customer demands on software system solution. These demands are usually captured as text statements about capabilities of software system (Pozagj, Sertie & Boan, 2003, p. 670).

Methodology activities are intended for assessing system scope in order to examine target system problem areas and to define global project scope in terms of used technology. After that, a competence-based project infrastructure should be established. —This means that developer's competencies and knowledge should be assessed because each member of the software development team should be responsible for tasks that are closely connected with his competencies" (Pozagj, and et al, 2003, p. 671).

Information about requirement dependency is important for selection of requirements to be realised. By capturing dependency information, project teams can easily choose requirements for realisation. When the whole set of requirements is identified, described and analysed in terms of dependency, requirements should be assigned to particular project members according to their competencies and become their responsibility. Therefore, presented methodology divides requirements according to knowledge areas on software development projects in order to improve the requirement discovery and definition process because project team members will better understand and capture requirements.

2.3.3 Risk Management in relation to requirement specification

At this point, it should be remembered that requirements are a means, not an end. Requirements that deliver value are defined here as the benefit of reducing development risk (developing a system that doesn‘t satisfy stakeholders‘ desires and needs) minus the cost of specifying the requirements. Consequently, replace the rule „You shall quantify all quality requirements‘ with „A quality requirement should be represented such that it delivers optimum value‘. This will demonstrate that risk assessment is the key means to determining how a given quality requirement should be represented.

By choosing the representation on the basis of a risk assessment, it can emphasise the fact that projects can fail if quality requirements aren‘t considered and treated adequately. Basically, they must assess how every quality requirement should be represented so that it delivers the most value. This means assessing the risk of developing a system that doesn‘t satisfy the stakeholders‘ desires and needs with respect to a given quality requirement, and how they can mitigate this risk at the lowest possible cost. A risk-based, value-oriented strategy for specifying quality requirements needs a broad range of representation forms

> Whenever there‘s a high risk that the deployed system won‘t meet a quality requirement to the satisfaction of a critical stakeholder (and there‘s no way to weaken the requirement, lower the stakeholder‘s expectations, or shift the risk onto somebody else‘s shoulders), the best way to mitigate this risk is still a classic, comprehensive quantification of the requirement (Glinz, 2008, p. 37).

2.3.4 Stage-based lifecycle models

A requirement specification represents both a model of what is needed and a statement of the problem under consideration. This type of specification is derived through an iterative approach which involves the two major activities of conceptual modelling and analysis of modelled reality. „These activities involve much informality and uncertainty.

Consequently, some authors attribute the problems in requirement specification to the nature of the task" (Loucopoulos & Champion, 1990, p. 116).

Software requirements must be used to guide such approach to software development, because iteration is based on selection of software requirements that will be implemented. This approach to software development demands useful definitions of software requirements because software requirements are used to plan software development. However, there are many problems connected with effective use of software requirements for guiding software development such as bad requirement definition, wrong requirement selection and change of requirements during software development. Besides these problems, software requirements are often badly organised and difficult to understand.

> To solve these problems with stage-based lifecycle models, it proposed methodologies for efficient requirement management. The first proposed methodology focuses on the process of requirement discovery and definition. The process of requirement discovery is highly important because all sets of requirements must be discovered in order to build software solutions that satisfy customer demands. Therefore, to avoid problems connected with late requirement discovery such as system architecture change. –System architecture can be defined as software system organisation or structure of significant components interacting through interfaces and development of wrong software systems, a specific approach to requirement discovery proposed with this methodology should be taken" (Pozagj and et al, 2003, p. 671).

2.3.5 Requirement specifications stages frameworks

Requirements prototyping are simulation models designed to help you learn more about the stakeholder's requirements. The aim of a prototype is to make it easier for people to imagine what it might be like to use the real product to do work. Ideally, working with this

model will stimulate them into remembering requirements they have forgotten, or thinking of ideas that might not otherwise occur to them until they began using the real product.

Low-fidelity prototypes offer a quick way to put together a mock-up of a product using familiar technology such as pencil and paper, whiteboards, flipcharts and so on. These protocols encourage stakeholders to focus on what the product does. They help to discover missing functionality and to test the scope of the product. High-fidelity prototypes use software tools and give the appearance of reality, and their advantage is obvious: It takes little imagination to see the prototype as a working system.

> The authors suggest that you adopt prototypes of both kinds as a regular part of the requirements process. The review process follows an iterative cycle until all problems have been resolved. That is, when errors are discovered, their corrections are reviewed and, if necessary, the specification looked at again to ensure none of the corrections introduced new problems. This iteration continues until you stop finding errors. The authors recommend keeping a record of discarded requirements to prevent their accidental reintroduction and to monitor which kinds of requirements are being rejected. This kind of documentation might prevent the reappearance of the unwanted requirements in future projects (Robertson & Robertson, 2008).

The review gives an ideal opportunity to reassess the easier decision on whether to go ahead with the project. A seriously-flawed specification or indication that the costs and the risks outweigh the benefits is almost always an indication that there may be a need to consider project euthanasia. The requirement process is not applicable just to new products that are being developed from the ground up.

> Most product development that is done today is aimed at maintaining or enhancing an existing product or at making a major overhaul to an existing product or suite of products. A lot of today‘s development involves Commercial off-the-Shelf (COTS)

products, open source products or other types of component ware. Whatever the development method, understanding the requirements for the final outcome is still necessary (Robertson & Robertson, 2008).

The section continues to focus on use cases as the requirement specifications and proposes a technique to check whether the given use cases are implementable with the framework.

To check the implementability, consistency of branch conditions of the frameworks and the requirement specification have to be checked as well as equivalence of action sequences between the frameworks and the requirement specification. To this end, a novel approach based on a satisfyability problem for deriving the consistent truth assignments of the branch conditions is introduced. The approach can be incorporated in bi-simulation checking for assuring the equivalence of the action sequences, and therefore, the implementability can be checked. Furthermore, this paper shows a feasibility of the proposed technique by using Compositional Reachability Analysis as a mean of bi-simulation checking (Zenmyot, Kobayashi & Scakit, 2008, p. 1).

2.4 Standard and requirement stage in combination

Good requirements practices can accelerate software development. The process of defining business requirements aligns the stakeholders with shared vision, goals and expectations. Substantial user involvement in establishing and managing changes increases the accuracy of requirements and ensures that the functionality will enable users to perform essential business tasks. Software requirements engineering encompasses the two major sub-domains of requirements definition and requirements management. A variety of practices can help software teams bridge communication gaps and do a better job of understanding, documenting and communicating customer needs. By incorporating relevant standards into this procedure, a more robust and precise requirement specifications can be built.

A set of requirements is needed for any project, especially computer system projects, to be successful. This is where many projects fail, in that they do not specify correctly what the system should do. In fact, many systems have just been given a deadline for delivery, a budget to spend, and a vague notion of what it should do.

The root of this problem is:

1. Computer systems developers rarely have as good an idea on how a business runs or should run, compared to a business user,

2. Business users have little idea of what a computer system could achieve for them.

As a result paralysis sets in and business management time is concentrated on meeting timescales and budgets, rather than what are going to be delivered and the quality of the system. The advantage of a good set of requirements need not just be a reduction in costs. In fact, many systems justified on a reduction in operating costs fail to deliver as low skilled but relatively cheap staff has to be replaced by high skilled and more expensive staff. The advantage can be a reduction in time to process something, which will lead to a reduction in costs, or being better able to use the unique knowledge base belonging to a business.

In addition to the general constraints you may include some development constraints. These are mainly in project management, but are still a restriction on the types of solution that can be offered. There are three general types of development constraints:

a. Time - When a system should be delivered is the obvious time constraint.

b. Resource - How much money is available to develop the system is obvious, but a key resource would be the amount of time business staff could spend in briefing system development staff.

c. Quality - Any standards which are used to develop the system, including project management, development methods, etc.

The goal is to deliver high quality and well-structured requirement specification documents. For example, the system to be specified was a set of embedded systems responsible for driver and passenger comfort. A feature requirement specification template was defined to capture and document the results of the specification activities. For each feature, the template was used and consisted of the following sections:

a. Basic feature description
b. Context diagram
c. Scenarios
d. Detailed requirements
e. Interface and data descriptions

—The activities resulted in about 50 feature requirement specification documents. On average, each specification document covered about 40 pages. 48% of the specifications delivered are expected to be highly stable, 24% are still ranked as quite stable" (Robertson and Robertson, 2008). At the end of the specification period all documents were required to be inspected. The goal of this step was to improve the quality of the requirement specifications, to enhance the common understanding of the content of the documents and to eliminate open points, mistakes and ambiguities.

> Due to the large volume of documents a parsimonious yet effective inspection approach was necessary. Hence, the defect detection as well as the meeting-based collection activity was modified to fulfil these requirements as well as to address the inspection issues outlined above. This resulted in the non-traditional inspection implementation (Laitenberger, Beil & Schwinn, 2002, p. 3).

Rework typically accounts for up to 40% of a development organisation's total budget, and most of this rework focuses on correcting software requirements defects.

Software requirements is a collaborative process for simulating, iteratively improving and validating a set of requirements to which all key project stakeholders agree, enabling:

1. Save time and money: Accurate upfront software requirements definition helps ensure your team works on the business problems that matter most

2. Reduce rework: Early validation and agreement by stakeholders means development and quality teams spend less time on rework and deliver projects faster

3. Improve requirement accuracy: The collaborative creation of working simulations improves accuracy by promoting understanding and eliciting relevant feedback—before development begins

The other areas that affect the project lifecycle, including budget, cost and quality are requirement creep and requirement leakage. Requirement creep refers to new requirements entering the specification after the requirements are considered complete. Any requirement appearing after this point is considered to be requirements creep. Requirements creep has been tagged with a bad name, usually because of the disruption to the schedule and the bloated costs of product delivery.

> Without wanting to defend requirements creep, the authors think it prudent to look at some of the causes of creep and to discuss how they can approach that problem. Firstly, most creep comes about because the requirements were never gathered properly in the first place. If the requirements are incomplete, then as the product develops, more and more omissions must of necessity be asked for. The authors suggest they were requirements that really were part of the product all along. They were just not, until now, part of the requirement specification (Robertson & Robertson, 2008).

By implementing standards early, requirement creep can be minimised. The main areas where requirement creep occurs is if the users and the clients are not given the

opportunity to participate fully in the requirements process, then specification will undoubtedly be incomplete. Almost certainly the requirements will creep as delivery approaches and the users begin asking for functionality they know they need. Creep is also observed because the original budget, for corporate policy reasons, is set unrealistically low. When noticeable creep sets in it is mainly not a matter of the requirements creeping, but of the product itself not being up to the correct functionality and also requirements change. Quite often they change for very good reasons in that the business has changed, or new technological advances have made change desirable. These kinds of changes are often seen as requirements creep. In truth, if changes that cause new requirements happen after the official end of the requirements process, and they could not have been anticipated, then this type of requirement creep could not have been avoided. Whatever the reason, whether good or bad, reasons for requirement creep must be identified and must be able to respond appropriately

In summary, the best way to minimise requirements creep is to engage in a good requirements process, with the active and enthusiastic participation of the stakeholders and to start with a reasonably sized project guided by relevant standards. Anything less and there can be requirement creep which must be expected.

Requirement leakage refers to requirements that somehow ~~l~~eak" into the specification. For example, think of this as the way water can leak into a rowboat as you cross a lake. Little water may not harm you, but too much of it and your chances of getting safely to the other side are seriously diminished. You can also think about requirements leakage as unrecognised. The main problems with requirement leakage are that nobody knows where the requirement leakages come from and who is responsible for them. Therefore, nobody wants to own them, and yet leaking requirements affect the budget. Either they are rejected or the project plan is adjusted to reflect the current reality.

Jackson (1995) reports –for the average project, about 33% of the requirement appear after the requirements process is deemed to have ended. That is about one third of all the requirements that have crept or leaked into the specification." The graph in Figure 1 depicts the cost of delivering functionality.

Look at what happens when the size of the product creeps up by 35%. The effort needed expands by a little more than that, yet this is the part of the product that somebody expects to get for free. When the requirements grow beyond what was organically anticipated, the budget must grow proportionally. Each requirement has a cost attached" (Robertson & Robertson, 2008).

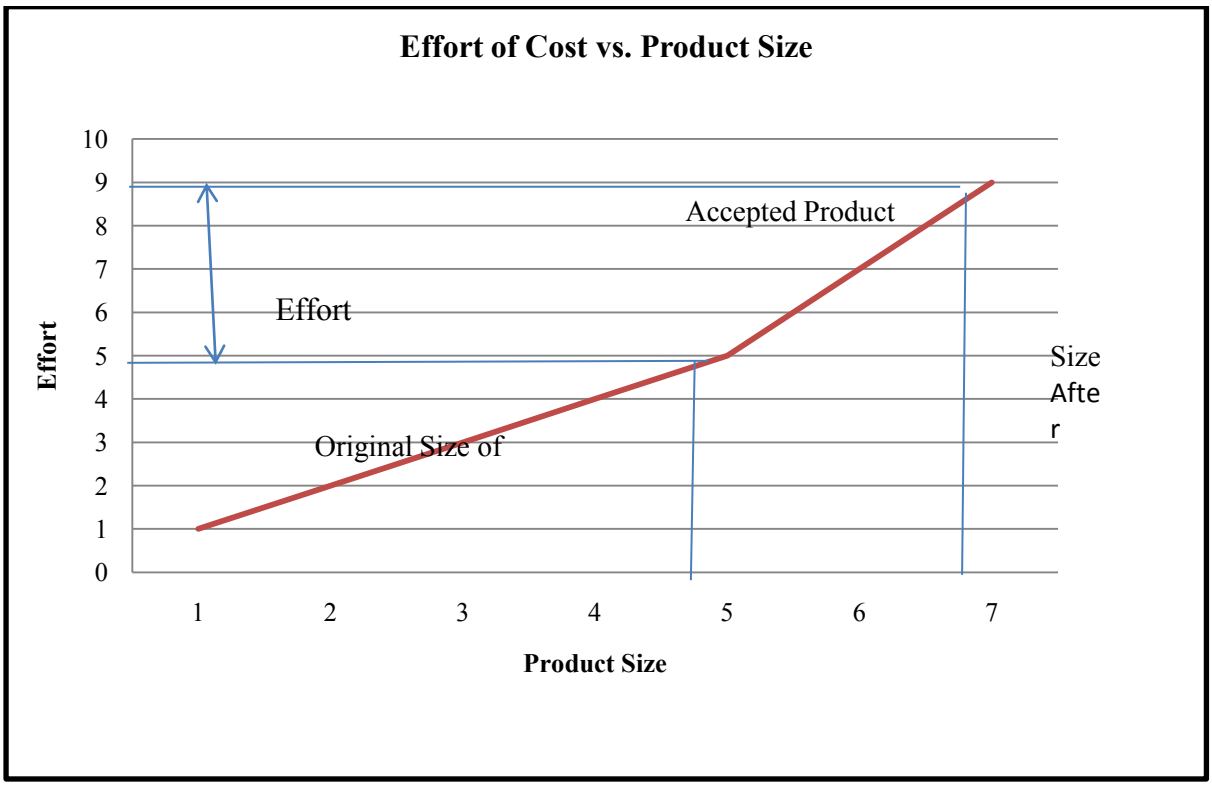

Figure 1. Effect of Cost vs. Product Size. From Robertson & Robertson (2008).

2.5 Conclusion

In conclusion, the four standards (IEEE Std 15288, IEEE Std 12207, IEEE Std and ESA Board for Software Standardisation and Control) are relevant to the requirement specification of the project lifecycle and therefore in turn to this thesis research. By applying

these standards a more productive and responsive requirement specification document can be produced.

The requirements definition is the most crucial part of the project. Incorrect, inaccurate, or excessive definition of requirements must necessarily result in schedule delays, wasted resources, or customer dissatisfaction. The requirements analysis should begin with business or organisational requirements and translate those into project requirements. If meeting stated requirements will be unreasonably costly, or take too long, the project requirements may have to be negotiated down, down-scoped or down-sized, in discussions with customers or sponsors. Any discussion of requirements analysis methods will quickly become specific to the type of project effort. Many industrial areas have specific, proven techniques for obtaining thorough and accurate definition of requirements.

There are three types of major problems with requirements definitions written in natural language;

1. Lack of clarity,
2. Requirements confusion from requirements not being fully traceable.
3. Requirement amalgamation.

Good requirements practices can accelerate software development. The process of defining business requirements aligns the stakeholders with shared vision, goals and expectations. Substantial user involvement in establishing and managing changes to agree upon requirements increases the accuracy of requirements, ensuring that the functionality built will enable users to perform essential business tasks. Software requirement engineering encompasses the two major sub-domains of requirements definition and requirements management.

Chapter 3 – Methodology

3.1 Research Method

This study is intended to examine and describe a requirement specification stage of the project lifecycle incorporating international and industrial standards. The description was developed based on international and industrial recommendations including risk management, stage-based lifecycle models, and frameworks. The research literature indicates that requirement specifications are an important consideration in the project lifecycle. A case study was conducted as an investigation, and further examined using interview techniques. This chapter summarises the studies, and discusses the methodology of the current thesis.

The main area of interest in the methodology of this thesis was a case study with statistical analysis. A case study is described by Leedy (2005) ~~–as~~ a particular individual, program or event which is studied in depth for defined periods of time". For this thesis the case study looks at the natural course and treatment of requirement specification gathering using relevant standards for guidance. The case study covers a range of areas in this topic including the following

1. Different approaches taken in industries.
2. Requirement specifications purpose and usage.
3. The standards relevant to requirement specification and industries.
4. Tools and equipment used during requirement specification gathering.
5. Standards and requirement specification in combination.

The case study proved useful for investigating how individual requirements and projects change over time, due to certain circumstances or interventions. In either circumstance, the case study was useful for generating or providing preliminary support for

hypotheses. It has been decided to perform the case study using 25 individuals from 4 different industrial sectors, therefore giving a wide range of information.

3.2 Application of thesis in the research subject

The main aim of the case study for this thesis is to learn more about the requirement specification stage of the project lifecycle, and also how companies change over to gathering requirement specification in detail and implementing standards in relation to requirement specification.

The method used for the case study was a collection of extensive data from the individuals on which the investigation was focused. These data included observations, interviews, documents (questionnaire, past articles, journals, and books), and past records (past case studies). I also recorded details about the context surrounding the case, including information about the physical environment, and the economic and social factors that affect the situation.

3.3 Justification

The justification is performed to examine the correct qualitative and quantitative research to use for the thesis study. The main research techniques found to be suitable for the thesis research was a case study with statistical analysis.

3.3.1 Qualitative Research

The main research techniques that were examined for the thesis research and deemed non applicable were ethnography, phenomenological studies, grounded theory study and content analysis. The main reasons for not using these research techniques are outlined below.

In ethnography, the research looks at an entire group, more specifically a group that shares a common culture in depth. The research studies the group in its natural setting for lengthy time periods (several months or years) and the main focus of investigation is on the

everyday behaviour of the people in the group. The subject of thesis requirement specification and standards is not an everyday behaviour, and the time line required for the ethnography was not available. Also this technique requires experience and is not suitable for a novice like myself.

Phenomenological studies attempt to understand people‘s perceptions, perspectives and understanding of a particular situation. –In order words, a phenomenological study tries to answer the question of what it is like to experience such and such. In some cases, the researcher has had a personal experience related to the phenomena in question, and wants to gain a better understanding of the experiences of others” (Leedy, 2005). Although phenomenological research depending on exclusively lengthy interviews with carefully-selected samples of participants (5 to 25) it could have been performed for this thesis, instead I decided to perform a case study and interview 5 people in relation to the case study.

The next qualitative research is grounded theory study, which is one of the researches that are least likely to be used for a particular theoretical framework. The main purpose of the grounded theory is to begin with the data, and use them to develop a theory. More specifically, a grounded theory study uses a prescribed set of procedures for analysing data, and constructing a theoretical model from them. The term grounded refers to the idea that the theory that emerges from the study is derived from the –ground” in data that has been collected in the field, rather than taken from the research literature. Grounded theory studies are especially helpful when current theories about phenomena are either inadequate or nonexistent. Typically, a grounded theory study focuses on a process (including people‘s actions and interactions) related to a particular topic, with the ultimate goal of developing a theory about that process. This type of qualitative research is mainly used in research techniques outside of information system and computing and, therefore was not deemed suitable for this thesis topic.

The final type of qualitative research is content analysis. Content analysis is a detailed and systematic examination of the contents of a particular body of material for the purpose of identifying patterns, themes or biases. Content analyses are typically performed on forms of human communication, including books, newspapers, films, television, art, music, videotapes of human interactions, transcripts of conversional and internet blog and bulletin board entries. The content analysis involves the greatest amount of planning at the front end of the project. The researcher typically defines a specific research problem or question at the very beginning. The research also identifies the sample to be studied and the method of analysis early in the process. Content analysis is not necessarily a stand-alone design. For instance, content analysis might be incorporated into a cross-sectional study to discover development trends in children‘s conceptions. The main disadvantage of this qualitative search is that it is a challenging qualitative study due to its flexible nature, and therefore was not deemed appropriate for this thesis topic.

3.3.2 Quantitative Research

The other research methods that can be used are the following:

1. Pre-experimental designs.
2. True experimental designs.
3. Quasi experimental designs.
4. Ex post facto designs.

In pre-experimental design it is not possible to show cause and effect relationships, because either (a) the independent ―variable" doesn‘t vary, or (b) experimental and control groups are not comprised of equivalent or randomly-selected individuals. Such designs are helpful only in forming tentative hypotheses that should be followed up by more controlled studies. The main types are:

1. One-shot experimental case study

A single group is studied at a single point in time after some treatment that is presumed to have caused change. The carefully studied single instance is compared to general expectations of what the case would have looked like had the treatment not occurred and to other events casually observed. No control or comparison group is employed.

2. One group pre-test –post-test design

A single case is observed at two time points, one before the treatment and one after the treatment. Changes in the outcome of interest are presumed to be the result of the intervention or treatment. No control or comparison group is employed.

3. Static group comparison

A group that has experienced some treatment is compared with one that has not. Observed differences between the two groups are assumed to be a result of the treatment

One of the important drawbacks of pre-experimental designs is that they are subject to numerous threats to their validity. Consequently, it is often difficult or impossible to dismiss rival hypotheses or explanations. Therefore, researchers must exercise extreme caution in interpreting and generalising the results from pre-experimental studies (Child Care and Early Education Research Connections. (n.d.))

The main advantages in pre-experiments, can be a cost-effective way to discern whether a potential explanation is worthy of further investigation. However, the main disadvantage is that it offers few benefits since it is often difficult or impossible to rule out alternative explanations. The nearly insurmountable threats to their validity are clearly the most important disadvantage of pre-experimental research designs, and based on these disadvantages it was decided not to perform pre-experimental design for this thesis.

The first three designs of true experimental designs are

1. Pre-test – Post-test Control Group Design,
2. Solomon Four Group Design
3. Post-test-Only Control Group Design

True experimental design is regarded as the most accurate form of experimental research, in that it tries to prove or disprove a hypothesis mathematically, with statistical analysis. The main advantage of this design is that the results of a true experimental design can be statistically analysed, and so there can be little argument about the results. It is also much easier for other researchers to replicate the experiment and validate the results.

The main disadvantage of true experimental design is that while perfect in principle, there are a number of problems with this type of design. Firstly, they can be almost too perfect, with the conditions being under complete control and not being representative of real-world conditions. True experiments can be too accurate, and it is very difficult to obtain a complete rejection or acceptance of a hypothesis, because the standards of proof required are so difficult to reach. True experiments are also difficult and expensive to set up. They can also be very impractical. Based on these drawbacks it was decided not to proceed with true experimental design for this thesis research.

A quasi-experimental design is one that looks a bit like an experimental design but lacks the key ingredient of random assignment. With respect to internal validity, they often appear to be inferior to randomised experiments. However, there is something compelling about these designs; taken as a group, they are more easily and frequently implemented than their randomised cousins. The most commonly used quasi-experimental design is the non-equivalent groups design. In its simplest form it requires a pre-test and post-test for a treated and comparison group. This was not a workable method for the thesis study. The second design is the regression-discontinuity design. At first glance, the regression discontinuity design strikes most people as biased because of regression to the mean. After all, we're

assigning low scorers to one group and high scorers to the other. This was deemed not to be suitable for this thesis research.

The ex post facto design is a variation of the "after-only with control group" experimental design. The chief difference is that both the experimental and control groups are selected after the experimental variable is introduced rather than before. This approach eliminates the possibility that participants will be influenced by awareness that they are being tested. Ex post facto designs provide an alternative means by which a researcher can investigate the extent to which specific independent variables may possibly affect the dependent variable of interest. Although experimentation is not feasible, the researcher can identify events that have already occurred or conditions that are already present and then collects data to investigate a possible relationship between these factors and subsequent characteristics or behaviours. After observing that different circumstances have prevailed among two or more groups, the researcher attempts to determine whether these different circumstances preceded an observed difference on some dependent variable. Ex post facto designs are often confused with correlation or experimental designs because they share certain characteristics with each of these other design types. Like correlation research, ex post facto research involved looking at existing circumstances. However, like experimental research it has clearly identifiable independent and dependent variables. Although ex post facto studies lack the control element and so do not allow us to draw definite conclusion about cause and effect, it is nevertheless a legitimate research method that pursues truth and seeks out the solution of a problem through the analysis of data. This was deemed unsuitable as it requires a comparison between a control and experimental group which was not applicable to this thesis topic. .

3.4 Key Processes and Procedures

The method used for the case study includes collecting extensive data on the individual(s), methods used for requirement generation and event(s) on which the investigation is focused. These data included observations, interviews and documents. I also recorded details about the context surrounding the case, including information about the physical environment, and economical factors that have a bearing on the research topic.

3.4.1 Participating Groups

Data gained in the varied industrial settings of the participants assisted in understanding the patterns of requirement specifications, and the meanings and discussions of the participants regarding their participation in requirement specification gathering in accordance with standards. There are four sub-groups which were solicited for participation. The first group represented the medical industry with exposure to requirement specification and standards for that industry. The majority of the people had only worked in the medical industry and experienced standards relevant to that industry and had documented requirement specifications for that industry also. The second group represented the pharmaceutical industry, with exposure to requirement specifications standards for that industry. The majority of the people had only worked in the pharmaceutical industry with one working in computer systems previous to the pharmaceutical industry. The third group represented the software industry with exposure to requirement specifications and standards for that industry. The majority of people had only worked in that industry and were younger and less experienced than any other industry. The final group represented the electronic industry, from a variety of companies with exposure to requirement specification documentation and standards for the electronic industry. The majority of people had only worked in the electronic industry

3.4.2 Sampling

A minimum of 25 people were solicited for the research. The requirement specification and gathering techniques discussed previously were investigated using a sample of professionals, who are currently working with computer systems or automation control systems in the pharmaceutical, medical device, electronic and computer software industries. A sample of professionals is appropriate to this study for the following reasons:

1. The sample questioned and interviewed all work within the relevant industries to the standards examined.

2. The professionals all gather, document and review requirement specifications on a weekly basis and therefore have considerable knowledge in the area.

3. The professionals use and examine standards and operating procedures during their work life.

3.4.3 Data Collection

Twenty- five questionnaires were passed out to participants ranging from a number of industries, including the pharmaceutical, medical device, software and electronics industries throughout the Republic of Ireland. Participants filled out the questionnaire (Appendix B). Each respondent received a document consisting of:

1. A cover sheet explaining the purpose of the study, the participant‘s rights and the name of the contact person and telephone number of those who might have questions after the questionnaire was complete.

2. The questions

3. The instruments described in this section.

The purpose, task demands, and rights were explained in print when the questionnaire was being distributed. Respondents were told that the questionnaire would include questions concerning procedures, techniques, and experiences they may have had. They were

guaranteed anonymity and confidentiality for their responses, and they were told that the session would take 45 minutes or slightly more. In actuality, the time it took for the participants to finish was between 30 minutes and 1 hour. All participants were asked to sign a written consent form before completing their questionnaire. Participants were also given instructions on how to properly fill out the questionnaire before they started. The initial sample consisted of 25 respondents of which 23 chose to complete the questionnaire. Of these, 2 questionnaires were omitted because they were only partially completed. Finally, of the 21 remaining questionnaires, 20 were selected for this study because they met the criteria of having no missing data for any specific questions, and were working with requirement specifications in the related industries

3.4.4 Observations

Over the year, I conducted observations of five participants who were gathering requirement specifications in relation to standards within all four industries, in relation to automation and control system projects. These observations totalled 2 sessions, ranging from the shortest at 30 minutes and the longest at 3 hours. The main procedure for the observations was discussing the gathering of requirement specifications and the procedure and frameworks used to gathered the data. The longer session consisted of witnessing the documenting of the gathered requirement specifications, and the use of the standard in correlation.

3.4.5 Interviews

I had interviews with the five participants, which consisted of general discussion on requirement specification and their importance. The interviews also focused on techniques and procedures used to aid in the gathering of requirement specifications.

The interview was performed and yielded a great deal of useful information. The main questions that were asked to the interviewees were on the relevant standards that are

applicable to their industry, and any in-house procedures that were used during their requirement specification gathering techniques. The interview then proceeded to facts about the methods used for the generation of requirement specification including tools and standards. The interview continued by discussing projects that requirement specifications were implemented on and the advantages and disadvantages of performing a detailed requirement specification generation on projects. The interview also discussed the motives and justification behind implementing requirement specifications on projects. In conclusion, the past behaviours of the past projects where requirement specification were not focused on were discussed, and compared with projects where focus was paid to this specific phase of the project lifecycle, including requirement creep and leakage

The semi-structured interviews include the following types of questions:

1. What sector is your company working in?
2. Have you worked on any projects where requirement specification gathering was performed and how did that project proceed?
3. Do you produce requirement specification document on a daily, weekly or monthly basis?
4. What is the main reason for producing these documents?
5. Do you reference any standards when documenting requirement specification?
6. Do you reference any in-house standards and standard operating procedures when documenting the specifications?
7. What tools or techniques do you use during requirement specification gathering and which do you find the most useful?
8. Do you discuss and review the document with the stakeholders before finalising?
9. Do you use frameworks or criteria to sector the requirements gathered.

10. Do you document the requirements in a testable manner and if so can you please give examples?

11. Do you document the specifications mainly in technical language or business language and which do you find the most effective?

12. What are the main conflicts and obstacles that you find when performing requirement specification gathering?

13. Have you worked on any projects where requirement specification gathering was not performed and how did that project proceed?

14. Have you worked on any projects where requirement specification gathering was performed and how did that project proceed?

15. What do you think are the main factors that affect a project, i.e. budget, quality and schedule?

During the interviews I took heavily-documented notes and narrowed this area to a focus area on requirement specification generation, standards and the combination of both together. The main challenge was trying to discuss and examine the main pitfalls of projects from people, and the main areas of correction on projects in relation to requirement specification documents. As I collected and analysed the data from the questionnaire, I found issues to explore which were mainly in relation to the different industrial standards, different techniques used for requirement specification gathering including frameworks, and the major difference was the conflicts experienced by people and how the conflicts affected the overall project. These questions arose and created a need for further observation and contributed to a second round of interviews with people. I collected data from the interviews looking for emerging themes and recurrent events, categorised them and re-evaluated my themes and categorising. As I collected more data, I wrote analytical memos about my data and re-evaluated my previous theories as I compared old data with new. The themes of academic

engagement, generated by my study continued to expand in depth and breadth, and they generated more themes that guided the development of my study.

3.4.6 Official Records and Documents

Official records and documents were another source of information. At the start and during my study I went to the college libraries, and researched the topic through journals, papers, and books. I also performed searches on databases, paying particular attention to the standards and past papers in relation to requirement specification gathering. I took notes on my laptop and documented notes by hand.

To further examine and support the case study, I also reviewed official records and documents including user requirement specifications documents and standards. During my studies, I examined the relevant standards to industry in detail before performing the interview, and also as part of the Literature Review and Annotated Bibliography. I took handwritten notes and typed abstracts from the literature for later inclusion in my thesis with relevance to the questionnaire.

For example, by going through old papers on this topic it was found that some companies do not perform requirement specification but instead use the manual to examine the requirements and then test the requirements against the manual. This was found not to work for the majority of projects due to the manuals not being descriptive enough or not incorporating all aspects. It was also observed that companies did not find the process of writing the requirement specifications difficult, because they were not really concentrating on writing a traditional SRS document. Instead, they were writing a user‘s manual, which is a familiar document with a well-known structure. Even more important, it is a user-centred document which promotes the requirements elicitation process. –For instance, they frequently found themselves asking questions like, so, if the user wants to move an object, can he find out how to do it by reading the manual? Or how will the system respond if the

user tries to select a group of objects?" (Berry, 2003, p. 10) Due to this information, during the interview I also asked the interviewee if they use manuals for the requirement specification instead of documenting it themselves.

3.4.7 Coding and Analysis

As I collected and analysed data from preliminary observation, I found further issues to explore, mainly around the difference in projects when requirement specifications were documented, and when they were not documented. These questions arose and created a need for further observing or interviewing.

For example, one project I discussed during the interview process had no requirement specification gathered or documented, and no standard referenced. This project went over budget by 10% and over schedule by 2 months. This supported my theory that if the requirement specification stage of the project lifecycle is neglected, then it has a knock-on effect on the overall project in relation to budget and schedule.

However, another project where I had observed the documenting of the requirement specifications also went over budget by 30% and over schedule by 9 months. On reflection this project had very detailed requirement specifications, which were to some degree too detailed, and therefore caused confusion and over-designing to occur. Therefore a happy medium between these two requirement specification documentation techniques needs to occur and standards will aid in this.

3.4.8 Exploring Researcher Values

During this research I continuously reviewed my expectations and values as a regular reminder of the role that requirement specifications and standards have on the project lifecycle in relation to budget, schedule, and quality. I also performed ongoing self reflections in memos and discussion with peers throughout the course of the study, which helped me identify and account for the interference of assumptions in my study. For

example, sometimes I was tempted to express my opinion on the importance of detailed requirement specifications on a project but this would have caused bias in my research and swayed the interviewee into one particular direction.

3.4.9 Leaving the Field

The process of leaving the field was gradual in relation to this thesis topic. As I work in this field I will not be fully leaving my research. This field constantly changes in relation to standards, standard bodies, changing of the standards, and views on this topic. The one area I did feel I was lacking in was the research and examination of material from official standards, standards bodies and industrial standard operating procedure, which this thesis research clarified for me.

3.5 Resources

The thesis was completed by me (the author) who documented and distributed the questionnaire. I also performed the interviews with five (5) people. The five people ranged from two in the automation sector of the pharmaceutical industry, one in the medical device industry, one in the electronics industry, and finally one in the software development industry. There was no budget provided for this thesis topic and time resources were done in my spare time, outside of normal working hours.

3.6 Summary of methodology chapter

The methodology used for this thesis is in the form of a case study with statistical analysis, as it was decided that none of the other qualitative or quantitative research techniques were suitable for the thesis topic. The case study was based on a questionnaire that was distributed to a number of people from different industrial sectors. The case study was further examined by interviewing five people from different sectors of the industry. The interview was performed with the following questions in mind in relation to facts; people‘s beliefs, motives, present and past behaviours, standards of behaviours and reasons behind

actions taken. The main aim of the interview was to discuss the particular issues around requirement specification and standards. The main problems with interviews are that time is limited and people can feel uncomfortable. The results of the case study were analysed and documented, as part of Chapter 4 of the thesis Results and Analysis.

The details required for the case study were organised, and mainly examined the industry, standards, requirement specifications, and generation of requirements incorporating standards. Specific documents, including the specified standards, and standards operating procedures for requirement specification writing, were also referenced and examined during the case study. Identification of patterns was found via statistical analysis, and finally an overall portrait of the case was constructed and conclusions were drawn from the case study and statistics. A research report was also prepared for the case study as part of Chapter 4 which discussed the rationale for the case study, detailed description of the facts related to the case study, a description of the data collected and a discussion of the patterns found. The main purpose of the case study was to understand one person or situation in greater depth, and focus on one case or a few cases within its natural setting. The main methods of data collection used were observations, and interviews performed with five people from four different industries. The main pre-requisites of the case study was that the personnel being interviewed and completing the questionnaire were informed of the nature of the study, and were willing to participate in it.

Chapter 4 –Project Results and Analysis

The hypothesis of this thesis project to be discussed and answered is:

–The requirement specification stage of the project lifecycle is a neglected stage of the project lifecycle, and if, with the combination of relevant standards, can this stage of the lifecycle be improved".

The hypothesis is that there would be significant differences in schedule, cost and quality of the project, if the requirement specification gathering was introduced in combination with relevant standards. A more detailed review of the literature is presented in Chapter 2 Literature Review and Appendix A Annotated Bibliography.

4.1 Experimental Results

The experimental results are split into four (4) key topics, including the following;

1. Standards and relevance to requirement specification
2. Gathering of requirement specification
3. Documenting of requirement specifications
4. Conflicts and problems experienced with requirement specifications.

4.1.1 Standards and relevance to requirement specification

The results are from four different industries, so that requirement specification gathering and documenting used within the industry can be examined. The number questioned were 20 people from all four areas of the medical device, pharmaceutical, software, and electronics industries, with each industry consisting of 25% of the participants, and therefore an even split. The majority of the people used requirement specification techniques for gathering and documenting requirements in their daily work routine; with 90% stating that they used requirement specifications daily, and 10% saying they used requirement specification weekly. The percentage of participants that use requirement specifications daily

did not differ by industry, c2 (1, N = 20) = 0.90, p = .50. The 10% of weekly users were from the software industry.

The questions then led into regulatory standards, and the number of people in the industries that regularly reference standards for clarity and guidance reasons. The statistics found that only 40% of the participants referenced industrial standards. The main standards that were noted as being referenced by the participants were IEEE 12207 and IEEE 1233. One participant acknowledged a Food and Drug Administration (FDA) standard for computer systems (Code of Federal Regulations (CFR) Part 11 standard), which is only applicable to the pharmaceutical industry. It was therefore not discussed as part of the thesis research. The next question then led into in-house standards, and the number of people in the industries that regularly reference in-house standards within their relevant industries for clarity and guidance. The statistics found that 95% of the participants referenced in-house standards and guidelines. The remaining 5% did not use any in-house standards, but were within the 40% that used regulatory standards.

Finally, in relation to standards, it was found that the two main standards that were discussed were IEEE 12207 and IEEE 1233, with one participant referencing International Organisation of Standardisation (ISO) 9001 for the electronics industry.

4.1.2 Gathering of requirement specification

The research then moved into the area of gathering requirement specifications. The participants were asked what the main purpose of gathering was, and documenting requirement specifications within their daily jobs. Figure 2 demonstrates this:

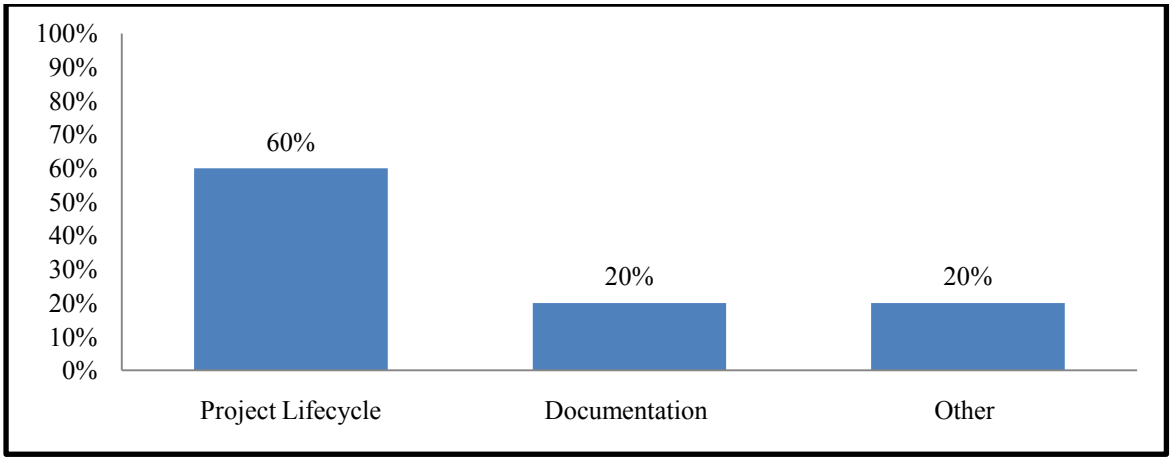

Figure 2. The purpose of requirement specifications.

Figure 2 demonstrates that 60% of the participants use requirement specification to aid, and contribute to the project lifecycle, while 20% use it for documentation purposes, and 20% use requirement specifications for other activities, mainly validation. To discuss this topic further, the types of tools used for gathering requirement specification were discussed, and Figure 3 demonstrates that use case tools are the most popular at 50%, followed by models and figures with 30%, and finally benchmarking at 20%. Benchmarking was found to be the most popular in the medical industry, with no other industry using benchmarking for the survey.

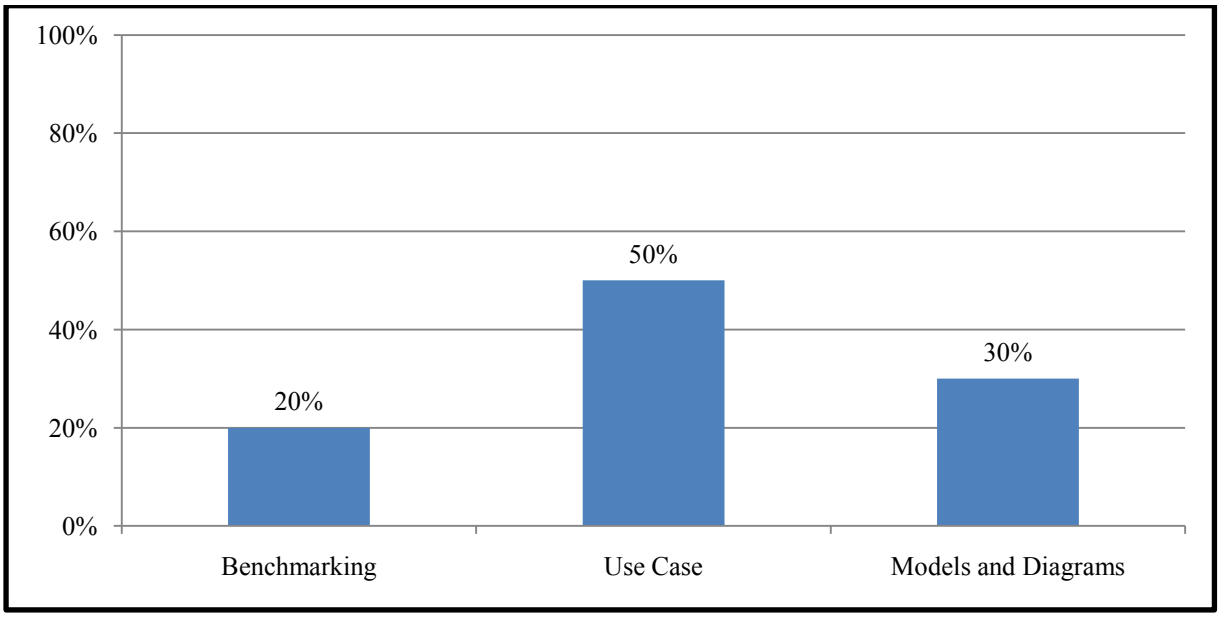

Figure 3. Tools for gathering requirements.

To further discuss and examine the techniques used for gathering and managing requirements, the following techniques were discussed:

a. Stakeholder analysis.

b. Secondary market research.

c. Context of use analysis.

d. Task analysis.

e. Rich pictures.

f. Field study.

g. Diary keeping.

h. Video recording.

It was found that some of the participants used many of these techniques to aid their requirement specification gathering and documenting. Figure 4 demonstrates the trend within this area of techniques.

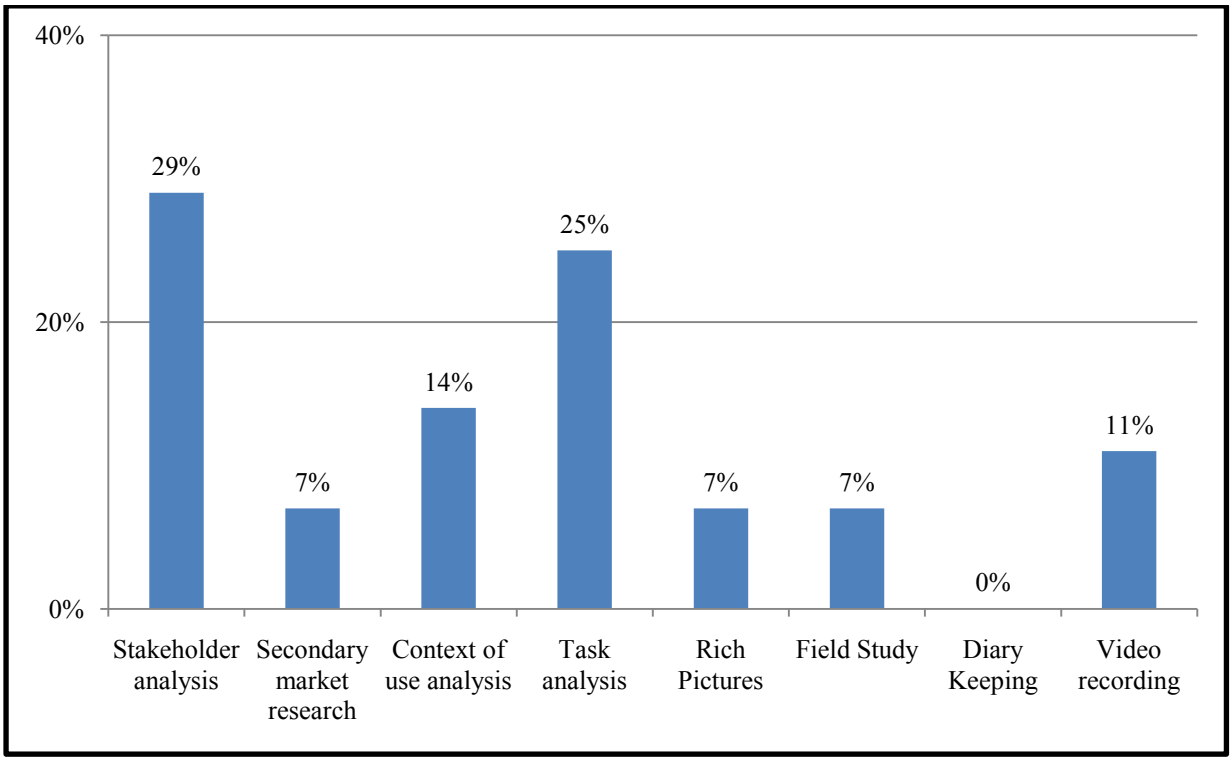

Figure 4. Requirement specification gathering techniques.

The next logical step was to discuss the stakeholders‘ part in relation to requirement specifications, and if the participants communicate and discuss with the stakeholders before formalising the requirements specification. It was drawn from the questionnaire that 90% of the participants discussed the requirements specification with the stakeholders, before formalising the document. Finally, the last part of this section on requirement specification gathering was in relation to when the requirement specifications are gathered, and what techniques are used to identify individual needs of the users, as per Figure 5.

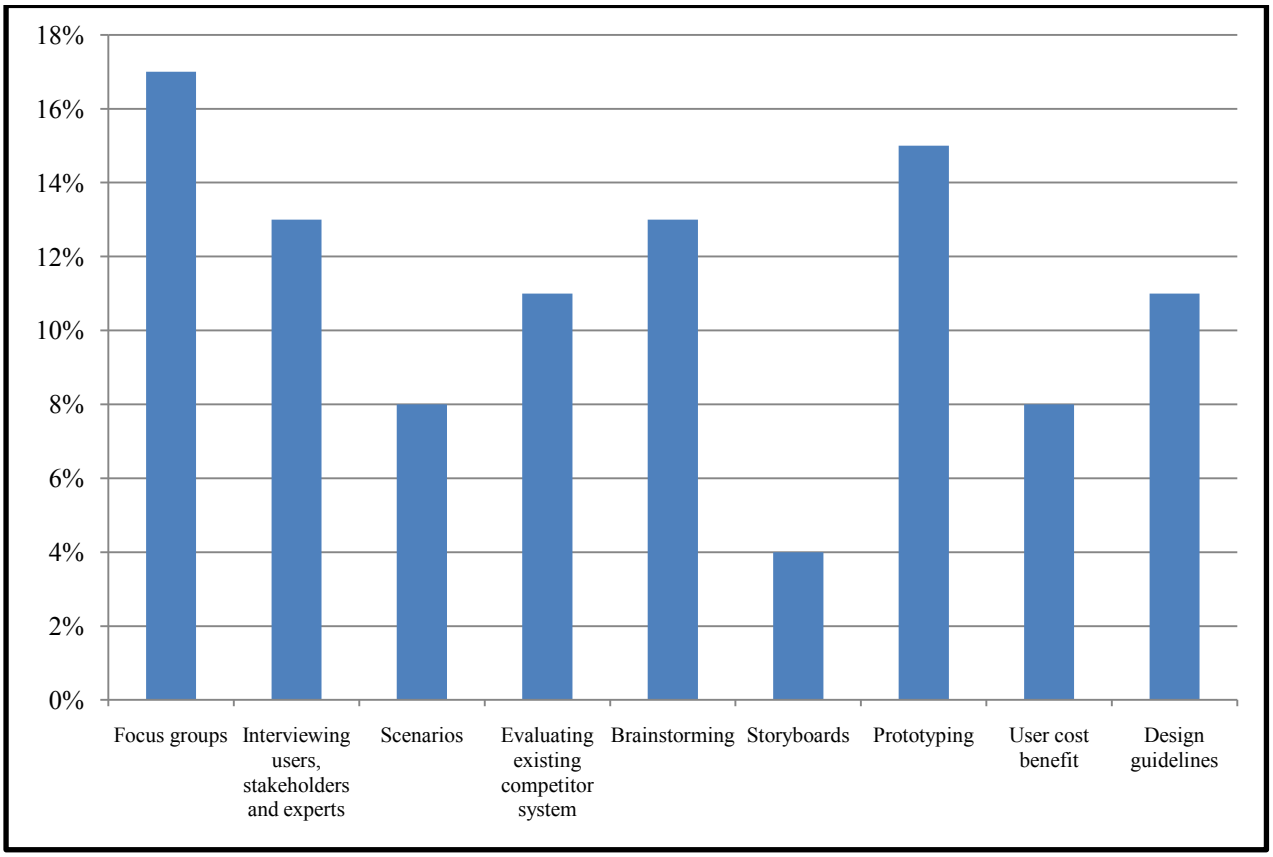

Figure 5. Techniques to identify the needs of users.

4.1.3 Documenting of requirement specifications

The third stage of the results section proceeds to the documenting of requirement specifications. The first section that the participants were asked to explain was when user requirements had been agreed on. Additionally, the required frameworks were examined and, if prioritisation or criteria setting frameworks were implemented. It was found that 50%

of the participants used criteria setting, 20% used prioritisation frameworks and 30% used no frameworks, as Figure 6 demonstrates.

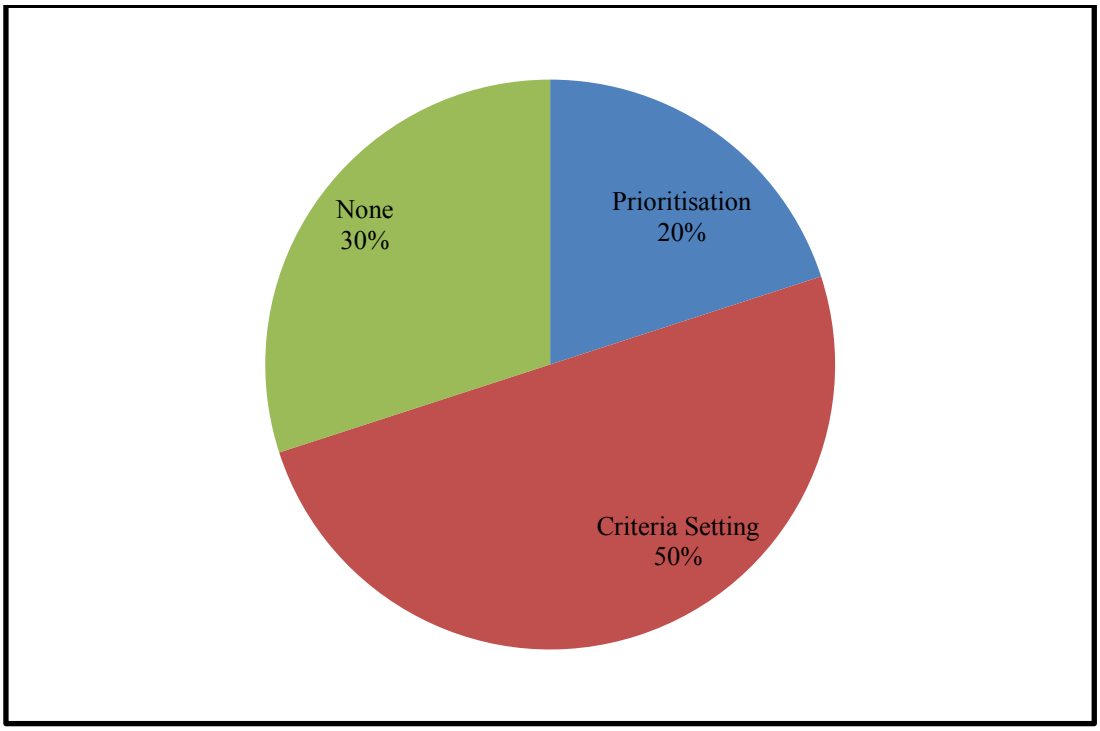

Figure 6. Frameworks in requirement specification documentation.

The next question was to ensure that the stakeholders fully understood and agreed with the written requirements, before they are passed downstream to other departments. It was found that 100% of participants ensured this step occurred before proceeding further with the requirement specification document. The questionnaire then discussed the documenting of requirement in a testable manner, and if this was a priority to the author of the requirement specification document. It was found that 60% of the participants wrote the requirement specification in a testable manner, while the remaining 40% did not deem this a priority of requirement specification documentation. Also in the writing of requirement specification it was asked if most of the requirements were written in a business language, a technical language or both. From the participants‘ feedback it was shown that 70% of the participants used a mixture of both business and technical language, 20% used only business language, and 10% used only technical language (Figure 7).

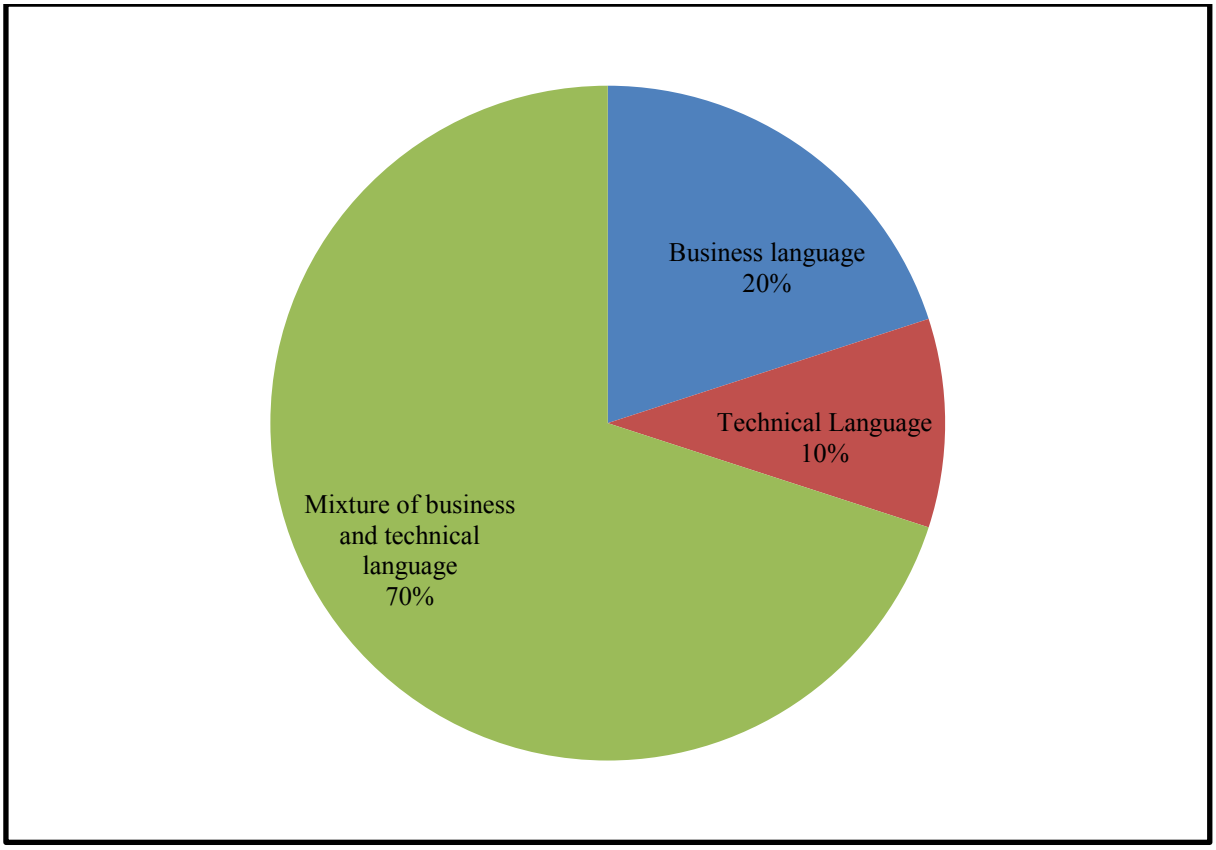

Figure 7. Languages for requirement specification documentation.

To conclude with this section of the results, the final question was whether participants distinguished between formal and informal requirements. It was calculated that 50% of the participants said yes they do distinguish, while 50% said they do not. This was further investigated in the interview process to clarify this area.

4.1.4 Conflicts and problems experienced with requirement specifications

The final section of the questionnaire was to discuss the main conflicts that occurred during the requirement specification stage of the project lifecycle, and how these conflicts were overcome. The main conflict that caused problems during the requirement specification generation was level difference, which is where the level of experience of people or understanding of the project differs significantly. The next conflict was inter-group prejudices, such as conflicts between technical and marketing staff, or quality and manufacturing staff. The final conflict to be discussed was personality clashes, for example

where people cannot work together and cause problems within the project group. Luckily this was only seen by one of the participants and is a limited problem conflict, but hardest to overcome. Figure 8 shows the percentages of the main conflicts.

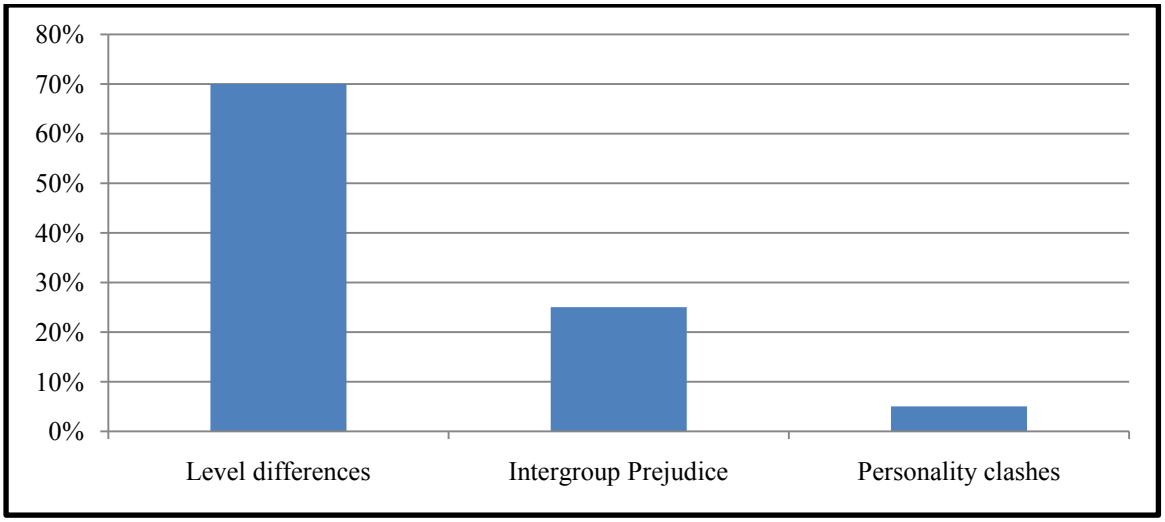

Figure 8. Main Conflicts experienced during requirement specification generation.

4.1.5 Interview

The next stage of the project was the five interviews that were conducted with personnel who use requirement specifications and relevant standards in their daily work. The first area to clarify during the interview was the amount of time a requirement specification document would take to be gathered, documented, and accepted by stakeholders in relation to the overall project size. The project size was measured by the overall project budget. Figure 9 demonstrates the correlation found.

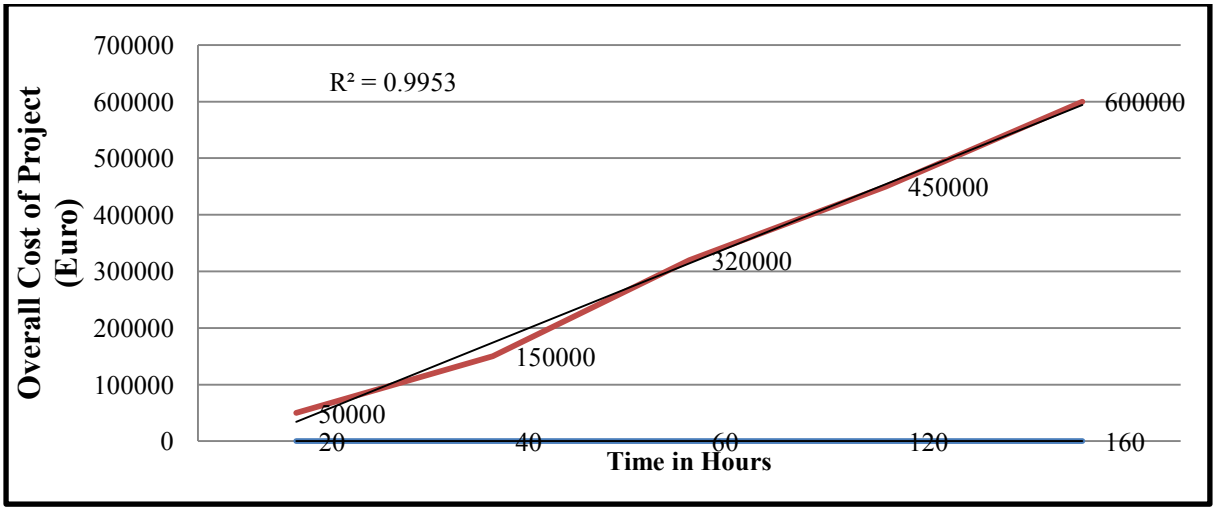

Figure 9. Correlation between hours spent on requirement specification documentation and overall cost of project.

The indicated correlation, found during the interview, was that the number of hours spent on requirement specification documentation and generation is directly related to the overall cost of the project. Therefore bigger projects required an increase in requirement specification but this does not necessarily mean more detail. This also indicated a positive correlation. Table 1 demonstrates the amount of detail required for documentation of a project in relation to cost.

Table: 1. *Requirement Specification as per document and per line*.

Overall Cost of Project (Euro)	**Number of requirement sections in a document**	**Number of requirements per section**	**Number of requirement documents**
80,000	12	1-5	1
120,000	12	5-7	1
300,000	13	6-8	2
1,500,000	19	10-15	3
2,500,000	23	15-25	7

Table 1 shows the relationship between the number of requirements present in a document, and the number of requirements listed per line in a section of the requirement document and number of requirement documents. The table also contains information on the overall budget of the project, and the indicated correlation between this figure and the number of requirements present in the document. This also supports the evidence that some requirement specification documents are heavily written with an extraneous amount of detail, while other requirement specification documents are written in a lighter format. Either of these methods adversely affects quality.

The next section to be examined in the interview process was requirement creep. It was found that requirement creep was a major issue within the medical device and pharmaceutical industries. Figure 10 demonstrates requirement creep within the different industries.

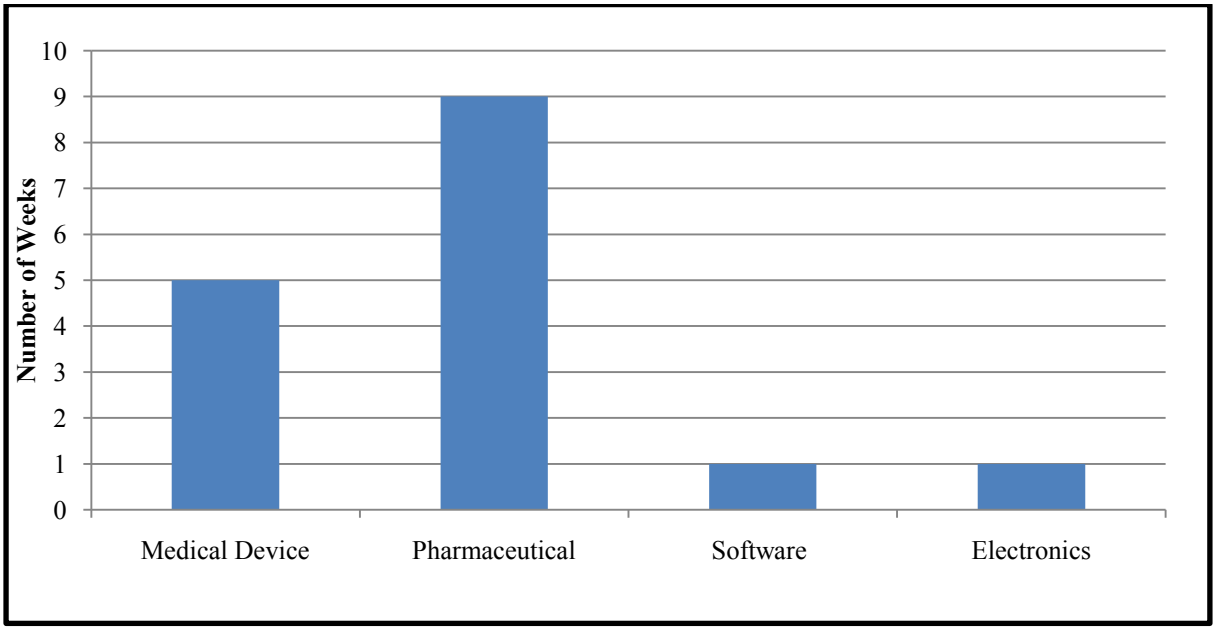

Figure 10. Requirement Creep.

During the interview process it was also evident that both the pharmaceutical and the medical device industries were more highly regulated. But the software and electronics industries were more in compliance with their relevant standards and risk management. After discussing requirement creep in depth during the interview, it was found from the ~~"l~~essons learned" documentation of the industrial projects being discussed by the participants, that between 40-60% of requirement creep was due to inaccurate or insufficient requirement specification documents.

The interview process also examined requirement leakage, and found that most documents within the four industries were revised and updated 2-6 times during the project lifecycle, depending on the size and complexity of the project. It was also noted that approximately 25% of requirements are gathered after the first revision of the requirement

specification document. Due to this issue, project managers deal with this problem by doing a risk assessment and placing a risk of 10% to 15% in their budget to allow for the cost of requirement creep and leakage.

During the interview process, the use of prototypes was examined; it was found that all four industries used prototypes in one form or another. The pharmaceutical and medical device industries used the prototypes for simulation techniques, and off-line testing to ensure that all requirements were met by the system. The electronics and software industries used the prototypes to demonstrate the possible usage of the system, and ensure the users / stakeholders understood the system fully. Also from doing the interview and witnessing process, it was evident that manuals were used as a tool for requirement specification gathering and documenting in the pharmaceutical and medical industries. The manuals were mainly used for the off-the-shelf systems.

During the questionnaire it appeared that one problem area was defining and distinguishing between informal and formal requirements. There was a 50/50 split on the question of whether this occurred or not. It was therefore investigated further in the interview process and found that the participants distinguished between requirements very differently, depending on the industry they work in. Table 2 demonstrates this point.

Table: 2 *Distinguishing criteria depending on industry*

Industry	Medical Devices	Pharmaceutical	Software	Electronics
Distinguishing Criteria	Quality critical Safety critical Product contact	Quality critical Product critical Environment, Health and Safety (EHS) requirements Business requirements	White box Black box	Quality critical Safety critical

4.2 Analysis of results gathered.

To further the research of the thesis, the results found and calculated during the results section are analysed to aid in the understanding of standards and requirement specification and their combination.

4.2.1 Standards and relevance to requirement specification

To ensure that the results were applicable to the thesis, the questionnaire was distributed 25% to the medical device industry, 25% to the pharmaceutical industry, 25% to the software industry and 25% to the electronics industry. The main standards that were identified during the analysis were IEEE 12207 and IEEE 1233. Both of these standards are discussed in Chapter 2, Literature Review.

The IEEE 12207 is an international standard which establishes a common framework for software lifecycle processes with well-defined terminology that can be referenced by the software industry. It contains processes, activities, and tasks that are to be applied during the acquisition of a software product or service, and during the supply, development, operation, maintenance and disposal of software products, which includes the software portion of firmware. The standard applies to the acquisition of systems and software products and services to the supply, development, operation, maintenance, and disposal of software products. It also applies to the software portion of a system, whether performed internally or externally to an organisation. These aspects of system definition are needed to provide the context for software products and services included. The standard also provides a process that can be employed for defining, controlling, and improving software lifecycle processes.

The purpose of the standard is to provide a defined set of processes to facilitate communication among acquirers, suppliers and other stakeholders in the lifecycle of a software product. The standard is written for acquirers of systems, software products and

services for suppliers, developers, operators, maintainers, managers, quality assurance managers, and users of software products.

IEEE STD 1233™-1998, IEEE Guide for Developing System Requirements contains specifications which provide guidance for the development of a set of requirements that, when realised, will satisfy an expressed need. In the guide, the set of requirements are called the System Requirement Specification (SyRS). Developing a SyRS includes the identification, organisation, presentation, and modification of the requirements. The guide addresses conditions for incorporating operational concepts, design constraints, and design configuration requirements into the specification. The guide also addresses the necessary characteristics and qualities of individual requirements and the set of all requirements.

The results also demonstrated the importance of in-house standards, and the number of in-house procedures that are in place for requirement specification generation. This supports the theory that requirement specification generation is becoming an important topic in the related industries, with 95% of the participants stating that there are in-house procedures and standards in place. This allows the requirement specification gathering and documenting to occur in a standard manner, with guidelines available.

In summary, it was found from the results that standards are now implemented during the requirement specification within all industries, either in the form of international standards, or as in-house standards.

4.2.2 Gathering of requirement specification

The study proceeded to discuss the main objectives of the participants when gathering and documenting requirement specifications. It was found that 60% were using requirement specification to aid with project lifecycle. The project lifecycle refers to the logical sequence of activities to accomplish the projects goals or objectives. Regardless of scope or complexity, every project goes through a series of stages during its life. There is first an

initiation, in which the outputs and critical success factors are defined, followed by a planning phase, characterised by breaking down the project into smaller parts/tasks, an execution phase, in which the project plan is executed, and lastly a closure or exit phase that marks the completion of the project.

The requirement specification stage takes effect during the planning stage of the project lifecycle. This second phase includes a detailed identification and assignment of each task until the end of the project. It includes a risk analysis and a definition of criteria for the successful completion of each deliverable. The governance process is defined, stakeholders identified and reporting frequency and channels agreed. The most common tools or methodologies used in the planning stage are business plans and milestone reviews including requirement specification.

Only 20% of participants used the requirement specification for documentation purposes, and the other 20% of participants used requirement specifications for other activities, which from further analysis, proved to be for validation activities, and requirement traceability matrices for projects included in the risk management.

In seeing how companies implement requirement specification, the results demonstrated the tools used for gathering requirements, with the most popular tool being use cases. These demonstrate the actors, system boundaries, system interaction and relationship in a clear and defined manner. Models and figures are also used for requirement specification gathering and documenting, including class figures and models of interactions. The last tool for requirement specification gathering is benchmarking, where companies compare previous similar systems from other projects and use the requirement specification documents to design and build a similar system and also to implement reuse.

To further discuss and examine the techniques for gathering and managing requirements, stakeholder analysis was examined, and found that 29% of the participants use

stakeholder analysis. Stakeholder analysis identifies all the users and stakeholders, who may influence or be impacted by the system. This helps ensure that the needs of all those involved are taken into account. User groups may include end users, supervisors, installers and maintainers. Other stakeholders include recipients of output from the system marketing staff, purchasers and support staff. Stakeholder analysis identifies for each user and stakeholder group their main roles, responsibilities and task goals in relation to the system. One of the main issues is how to trade off the competing needs of different stakeholder groups in the new system.

Secondary market research was used by 7% of the participants, involving researching published sources such as research reports, census data and demographic information that throw light upon the range of possible user markets. Websites representing special groups of users, such as that for the Royal National Institute for the Blind, give information about the nature of the user population they represent.

Context of use analysis was used by 14% of the participants, and is used when a system or product is developed. The quality of a system, including usability, accessibility and social acceptability factors, depends on having very good understanding of the context of use of the system. For example, in an office environment, there are many characteristics that can impinge on the usability of a new software product, e.g. user workload, support available, or interruptions. Capturing contextual information is therefore important in helping to specify user requirements. In order to gather contextual information, stakeholders attend a facilitated meeting, called a context meeting. Here a questionnaire is completed to capture the characteristics of the users, their tasks and operating environment

Task analysis proved popular with 25% of the participants using this technique. Task analysis involves the study of what a user is required to do, in terms of actions and/or cognitive processes to achieve a task. A detailed task analysis can be conducted to understand

the current system, the information flows within it, the problems for people, and opportunities that indicate user needs. There are many variations of task analysis and notations for recording task activities. One of the most widely used is hierarchical task analysis, where high-level tasks are decomposed into more detailed components and sequences. Another method creates a flow chart, showing the sequence of human activities and the associated inputs and outputs.

Rich pictures were only used by 7% and it can help stakeholders map, explore and understand a complex problem space, and thereby help to identify hidden requirements. The technique involves creating a series of sketches to show how people and systems relate to each other in an organisation. They may show peoples‘ roles, power structures, communications and reporting mechanisms. Drawing simple figures of people with thought and speech bubbles linked to them can show particular problem areas in the current environment that may lead to new user requirements.

The field study was used by 7% of the participants. Field study and observational methods involve an investigator viewing users as they work, and taking notes of the activity that takes place. Observation may be either direct, where the investigator is actually present during the task, or indirect, where the task is recorded on videotape by the analysis team and viewed at a later time. The observer tries to be unobtrusive during the session, and only poses questions if clarification is needed. Obtaining the co-operation of users is vital, so the interpersonal skills of the observer are important.

Diary keeping was not used by any of the participants, but could provide a record of user behaviour over a period of time. They require the participant to record activities they are engaged in, throughout a normal day, which may lead to the identification of user requirements for a new system or product. Diaries require careful design and prompting, if they are to be employed properly by participants.

Video recording was used by 11% of the participants, and can be used to capture human processes in a stakeholder‘s workplace or other location. The results can then be revised for the purpose of understanding more about the work and generating relevant questions relevant to user needs. Video can also be a useful supplement to other methods, e.g. to demonstrate new system concepts to users during user/stakeholder discussion groups.

In summary it was found that requirement specification were being implemented within all industries, and that tools and methods were being used to aid with requirement specification generation. As the requirement specification gathering techniques had been fully discussed, the analysis moved to the documentation discussion, in relation to requirement specifications.

4.2.3 Documenting of requirement specifications

The next logical step in the analysis section is the documenting of requirement specifications. The first section to be examined is frameworks, which examines prioritisation and criteria setting. It was demonstrated that 50% used criteria setting, 20% used prioritisation frameworks and 30% used no frameworks. Criteria setting relates to the need for criteria to help decide whether the user requirements have been achieved. This can be done by an inspection team or by user testing. Defining acceptance criteria in advance can be achieved by performing pre-tests on the existing system, or on a competitor system, to specify that the new system must be at least as good as these current systems. Prioritisation of user requirements is important so that development resources can be directed appropriately. This helps control the risks in system development, and allows the customer to redirect future effort to meet the user‘s needs more closely.

The importance of ensuring the stakeholders were fully involved in the requirement specification was analysed and the results found that all participants (100%) ensured key stakeholders were involved. By ensuring that all stakeholders were involved, all

requirements can be assessed fully, and a full understanding of the requirements is achieved by all people involved in the requirement specification generation procedure. To ensure stakeholders can fully understand the document, the following should be stated in the requirement specification document:

a. Identification of the range of relevant users and other stakeholders.
b. A clear statement of design goals.
c. The requirements with an indication of their priority levels.
d. Measurable benchmarks against which the emerging design can be tested.
e. Evidence of acceptance of the requirements by the stakeholders.
f. Acknowledgement of statutory or legislative requirements.

To further examine the documenting of requirement specifications, the manner in which requirements are documented was examined, including writing the requirements in a testable manner. It was found that only 60% of the participants wrote the requirements in a testable manner, with the remaining 40% not deeming this a priority. This was disappointing as requirement specification if written in the correct manner can aid with the testing and implementation phase of the project lifecycle. Therefore this was one section of the requirement specification that was seen to be slightly neglected.

Requirement specification, if written in a business language, technical language or a mixture will again aid with testing further within the project lifecycle. It was found that 70% implement a mixture of both languages, 20% only use business language and the remaining 10% use only technical language. By implementing a mixture in the languages it helps in the understanding between the stakeholders.

Another area to aid in the documentation of requirement specification is the splitting of requirements between formal and informal requirements. This again proved to be a

neglected stage of the requirement specification as it had a split response on the topic. Therefore, during the interview process it was further examined.

In summary, it was found that documenting of requirements was performed, using aids and frameworks, to ensure that requirement specifications were written in a language that was applicable to the industry. This was one area of the research that was lacking, and in some way neglected, as only 60% of the participants ensured that requirements were written in a testable manner.

4.2.4 Conflicts and problems experienced with requirement specifications

This area discussed the main conflicts experienced as part of the team group, and within the project team. It was demonstrated from the results that the main conflict (70%) experienced by participants was level difference. The level difference can be from within the project team structure, for example, the level difference of understanding between the project engineer and the quality assurance personnel. Another example is from outside of the main project team, for example between marketing people of a new-built system and the project builder of the system. The main ways of overcoming this area of conflict is by presenting the requirement specification data in a mixture of technical language and business language, therefore allowing all levels to understand the system more readily. The next conflict experience by the participants was inter-group prejudices, which 25% of the group said they had experienced. These prejudices stem mainly from the main objectives and purpose of different groups, for example, an engineering team person will be concerned with budget, scheduling and building of the system while a quality / validation team person will be mainly concerned with quality and only secondly concerned with budget and schedule. Therefore, this can cause a conflict within the key group, as neither of the peoples‘ key objectives is identical. The main way to overcome this problem is by communication within the wider

group of all key project objectives, i.e. budget, schedule and quality, and to equally share the objectives.

The last conflict to be experienced by participants in the group was personality clashes. This conflict is of major concern to any project, as it is not easily overcome, and is mainly due to personality clashes in organising key people. This requires all personnel to behave in a professional manner within the team group and not allow personal opinions to affect the documentation and execution of a project; luckily this conflict was only experienced by 5% of the participants.

4.2.5 Interview

The next stage of the project was in relation to the interview process. The main objective of this stage of the thesis research was to examine the correlation between hours spent on requirement specification documentation and the overall cost of the project.

The main observation made from this correlation is that, as the budget increases on the project, the requirement specification documentation increases, and in turn, the hours spent generating and documenting the requirements increase.

It was also found that requirement specification detail increased with the size of the project. For example a project worth €80,000 would usually have one requirement specification document, with approximately 12 requirement sections, with 1-5 requirements per section, while a project worth €1.5 million may have three requirement specification documents, with approximately 19 requirement sections, with 10-15 requirements per section. This was also witnessed when documents were examined and it was found that requirement specification documents from the pharmaceutical industry, specifically, had an extraneous amount of detail present, while requirement specification documents from the electronics industry were lighter on detail. The level of detail did not have any effect on the quality of the project, but the higher level of detail increased the cost of the project to ensure

that it was implemented correctly. This again was another key aspect that affected the project lifecycle. Also evident in the industries was the use of manuals when documenting the requirement specification. This was especially used for off-the-shelf systems, which reduced the amount of generation required for the requirement specifications, and aided immensely with documentation of the requirement specification documents.

The next logical section of this thesis to be discussed was requirement creep. Requirement creep refers to a new requirement entering the specification, after the requirements are considered complete. Requirement creep can come about because the requirements were not gathered completely in the first place. If the requirements are incomplete, then as the product develops, more and more omissions must, of necessity, be asked for. It was found during the interview process that requirement creep is a major issue in the pharmaceutical industry, with some projects delayed or off schedule by 9 weeks due to requirement creeps occurring during the life of the software project. The medical device industry experienced 5 weeks delays to software projects due to the same issue of requirement creeps. However, the software and electronics industries experienced only 1 week delays to software projects due to requirement creep.

By implementing standards early, requirement creep can be minimised. The main areas requirement creep occurs in is if the users and the clients are not given the opportunity to participate fully in the requirements process, then the specification will undoubtedly be incomplete. Almost certainly the requirements will creep as delivery approaches and the users begin asking for functionality they know they need. Creep is also observed because the original budget, for corporate policy reasons, is set unrealistically low. When noticeable creep sets in it is not a matter of the requirements creeping but of the product itself not being of the correct functionality. Quite often they change for very good reasons, i.e. the business

has changed, or new technological advances have made change desirable. These kinds of changes are often seen as requirement creep.

In truth, if changes that cause new requirements happen after the official end of the requirements process, and they could not have been anticipated, then this type of requirement creep could not have been avoided. Whatever the reason, whether good or bad, reasons for requirement creep must be identified, and must be responded to appropriately. After discussing this problem further during the interview process, it was found that requirement creep was mostly witnessed on bespoke software equipment within the industry. The fact that both the medical device and pharmaceutical industries were not fully specialised in software, causing the requirement specification documentation not to be as specified as is required for the bespoke software system, and requirement creep and leakage occur. In summary, it was found from the interview process that the best way to minimise requirement creep is to engage in a good requirements process, with the active and enthusiastic participation of the stakeholders, and to start with a reasonably-sized project, including system boundaries guided by relevant standards.

Requirement leakage was another area to be analysed as part of the interview process, and the results that were experienced. Requirement leakage refers to requirements that somehow ~~l~~eak" into the specification. The main problems with requirement leakage are that nobody knows where the requirement leakage comes from, and who is responsible for them. Therefore, nobody wants to own them, and yet leaking requirements have an effect on the budget. Either they are rejected or the project plan is adjusted to reflect the current reality. It was found during the interview process that a requirement specification can be revised from two to six times during the lifecycle of the project, to incorporate requirement specification adjustments, including creep and leakage. On average, about 25% of the requirements appear after the first requirement specification process has been completed. The main issue with

requirement leakage, or adding new requirements, is the cost involved and the adjustments that are required to the budget and the schedule. On investigation during the interview process it was found that the majority of the industry builds in a cost of between 10% to 15% risk of requirement leakage and requirement creep occurring. Risk management was also used throughout requirement specification, in terms of the amount of testing and details required for certain criteria, for example, quality criteria versus business criteria.

Prototyping was another area to be witnessed as part of the interview process; the prototyping was used in different aspects depending on the industry. The pharmaceutical and medical industries used prototypes to aid with requirement specifications, to ensure that all aspects of the system were thought about and in turn that the requirements were generated and documented as part of the requirement specification document. The electronics and software industries used the prototype to allow the stakeholders and users to interact with the system. This ensures that all key processes were in place and, if a process was found to be missing, that the stakeholder / user would highlight this issue, and the requirement specification document could be revised, to include this process within the relevant language.

Finally, during the questionnaire process the uses of informal and formal requirements were discussed, but did not prove beneficial. It was examined further and the main manner in which the requirements were distinguished was by using other forms of criteria. For example, in the medical device industry the requirements are distinguished using quality-critical, safety-critical and product-contact requirements. The formal requirements are examined as part of the quality-critical and product-contact requirements. The informal requirements are distinguished as safety-critical requirements. This manner of distinguishing was also evident in the pharmaceutical industry, with formal requirements being captured in quality-critical and product-critical requirements, and informal requirements being captured as part of EHS requirement and business requirements. Within the software industry, the

formal requirements and informal requirements are mixed between both the white box testing and black box testing, and therefore this is the manner in which this industry defines the requirements. Finally, the electronic industry uses quality-critical for the formal requirements and safety-critical for informal requirements. Requirements that may not be captured as informal or formal requirements may however be captured under a different name. However, in general terms, they can still be traced back to either formal or informal requirements.

The one main finding from this research is that all changes, revisions, updates and errors found during the project lifecycle affect the schedule and budget of the project. However, the quality of the system always remains at 100% and is never allowed to waver. Either the budget or schedule must be revised to ensure quality stays high.

4.3 Conclusion

The main findings from the results and analysis section is that the requirement specifications are not necessarily a neglected phase of the project lifecycle, but evidence shows that personnel perform requirement specifications in combination with standards, but not necessarily in the correct manner. From the research and analysis that was examined, it was found that some industries pay high attention to requirement specification, because they are regulated to do so, while other less-regulated companies still implement standards within the requirement specification, but in a more controlled manner.

On reflection, relevant standards are used to aid with the generation and documentation of requirement specification, and therefore it is not a neglected stage of the project lifecycle. However, it is a stage of the project lifecycle that is misunderstood and is not always completely reflective of the project itself.

Chapter 5 – Project History

5.1 The beginning of the thesis

The thesis began due to my work area being in the installation and commissioning of computer systems. My work was mainly based around gathering and documenting requirement specifications in alignment with current and applicable standards. My work consists of requirement specification gathering, generation of documents, and consistently ensuring the requirement specification documents are updated and reflect the live system. As this area was related to my job, this added extra emphasis to the topic and an easy way of introducing the thesis into my professional life, as one of the main objective of the thesis research was to ensure that the project was suitable to my profession and related to my daily work environment.

By ensuring that my chosen topic was a relevant and current one, I reviewed topics from other universities in relation to research opportunities. I found, from researching relevant papers and university postgraduate opportunities, that the requirement specification topic had been researched in detail in the early 1990‘s and that there were many relevant papers. The topic then re-appears in the late 2000s (2006 to 2009) and there were a number of relevant papers and research opportunities.

The main papers of relevance from the previous 20 years were the following:

1. Loucopoulos, P & Champion, R. (1990). Concept acquisition and analysis for requirement specification.

2. Jang, H. (1994). A knowledge based analyser for requirement specification analysis.

3. Yau, S. & Liu, C. (1988). An Approach to software requirement specification

4. Yau, S. & Bae, D. (1994). An approach to object oriented requirement verification in software development for distributed computing systems

5. Hunt, L. (1997). Getting the Requirement Right – A Professional approach.

6. Hesslink, H. (1995). Comparison of standards for software engineering based on DO-178B for certification of avionics systems.

The main current papers to aid in this decision to progress this research topic were:

1. Glinz, M. (2008). A Risk Based, Value Oriented Approach to Quality Requirements.

2. Alipourt, H. & Isazadeh, A. (2008). Software Reliability Assessment Based on a Formal Requirement specification.

3. Glass, R. (2009). Doubt and Software Standards

After performing a month of research in June 2009, I decided to follow this line of research into my thesis topic, and therefore continued my chosen topic through to the idea paper. The idea paper was submitted and named –Requirement specification stage of the project lifecycle and the standards that can be implemented at this stage".

Then I compiled my thesis idea paper, and I identified three suitable thesis advisors, with knowledge and interests in this general area of software engineering. The thesis advisor willing to accept the thesis was Mike Prasad, with whom I corresponded for approval of my thesis idea paper. I then formally submitted the idea paper.

5.2 Managing the thesis

From here the thesis was managed in two ways:

1. At the beginning it was managed according to the dates received from National University of Ireland (NUI) Galway.

2. After the first thesis module the thesis has been managed according to a Project Plan that was reviewed by me and my thesis advisor (Mike Prasad).

This project plan has been made a live document and is updated as required to include all adjustments and minor changes to the timeline of the thesis. The NUI Galway dates have

been incorporated in the project plan, to ensure that the project plan represents the full life of the project. Figure 11 shows the current project plan. The main changes to the original plan were in relation to the amount of work that was possible to commence on the thesis during the weekly modules of MScSIS course. The work load increased in some modules, which made it harder, if not impossible, to continue with work on the thesis project. Also my professional work life increased in complexity and workload, which in turn affected the thesis project progression.

ID	Task Name	Duration	Start	Finish
1	Idea Paper Research	21 days	Mon 01/06/09	Mon 29/06/09
2	Thesis Advisor Search	20 days	Mon 29/06/09	Sun 26/07/09
3	Thesis Advisor Found and Discussion	15 days	Mon 27/07/09	Sat 15/08/09
4	Thesis Idea Paper preparation	33 days	Wed 01/07/09	Fri 14/08/09
5	Thesis Idea Paper Submittal	1 day	Mon 17/08/09	Mon 17/08/09
6	Thesis Idea Paper Submittal Deadline	0 days	Fri 18/09/09	Fri 18/09/09
7	Thesis Module MCT-624 Part 1	40 days	Mon 19/10/09	Fri 11/12/09
8	Thesis Statement Preparation	20 days	Fri 30/10/09	Thu 26/11/09
9	Thesis Statement Submitted to Thesis Advisor	1 day	Mon 30/11/09	Mon 30/11/09
10	Re-Submittal of thesis statement following feed back	1 day	Mon 14/12/09	Mon 14/12/09
11	Completion of thesis module MCT-624 Part 1	0 days	Mon 14/12/09	Mon 14/12/09
12	Annotated Bibliography Research	16 days	Mon 21/12/09	Mon 11/01/10
13	Annotated Bibliography Drafting	20 days	Mon 11/01/10	Sat 06/02/10
14	Annotated bibliography Review	7 days	Sun 07/02/10	Sat 13/02/10
15	Annotated Bibliography Submittal	0 days	Sun 14/02/10	Sun 14/02/10
16	Questionnaire drafting	15 days?	Mon 15/02/10	Sun 07/03/10
17	Thesis Module MCT-624 Part 2	40 days	Mon 08/03/10	Fri 30/04/10
18	Completion of thesis module MCT-624 Part 2	0 days	Sun 02/05/10	Sun 02/05/10
19	Chapter 2 Literature Review	25 days	Fri 26/03/10	Thu 29/04/10
20	Chapter 2 Literature Review Submittal to thesis advisor	0 days	Thu 29/04/10	Thu 29/04/10
21	Questionnaire Distribution	20 days?	Mon 03/05/10	Sun 30/05/10
22	Questionnaire Review	5 days?	Mon 31/05/10	Sun 06/06/10
23	Interview Preparation and Organisation	5 days?	Mon 10/05/10	Sun 16/05/10
24	Interview Process and review	26 days?	Mon 17/05/10	Mon 21/06/10
25	Chapter 3 Methodology	29 days	Fri 21/05/10	Wed 30/06/10
26	Chapter 3 Methodology Review	4 days	Thu 01/07/10	Tue 06/07/10
27	Chapter 3 Methodology Submittal to thesis advisor	0 days	Wed 07/07/10	Wed 07/07/10
28	Chapter 4 Results Section	5 days	Mon 28/06/10	Sun 04/07/10
29	Questionnaire and Interview Trending and Statistics	3 days?	Wed 30/06/10	Sun 04/07/10
30	Chapter 4 Results Section Review	0 days	Sun 04/07/10	Sun 04/07/10
31	Chapter 4 Analysis Section	5 days	Mon 05/07/10	Sun 11/07/10
32	Chapter 4 Analysis Section Review	7 days	Sun 11/07/10	Mon 19/07/10
33	Chapter 4 Results and Analysis Submittal to thesis advisor	0 days	Mon 19/07/10	Mon 19/07/10
34	Chapter 5 Project History	4 days	Wed 14/07/10	Sun 18/07/10
35	Chapter 5 Project History Review	1 day	Mon 19/07/10	Mon 19/07/10
36	Chapter 5 Project History Submittal to thesis advisor	0 days	Mon 19/07/10	Mon 19/07/10
37	Chapter 6 Conclusion	4 days	Tue 20/07/10	Sun 25/07/10
38	Chapter 6 Conclusion Review	1 day	Mon 26/07/10	Mon 26/07/10
39	Chapter 6 Conclusion Submittal to thesis advisor	0 days	Mon 26/07/10	Mon 26/07/10
40	Chapter 1 Introduction	7 days	Thu 29/07/10	Fri 06/08/10
41	Chapter 1 Introduction Review	1 day	Fri 06/08/10	Fri 06/08/10
42	Chapter 1 Introduction Submittal to thesis advisor	0 days	Sun 08/08/10	Sun 08/08/10
43	Compile Appendices	8 days	Thu 29/07/10	Sun 08/08/10
44	Format and Tidy appendices	10 days	Mon 09/08/10	Sun 22/08/10
45	Final Review and re-formatting	10 days	Mon 09/08/10	Sun 22/08/10
46	Preparation for Submittal of final thesis	3 days	Mon 23/08/10	Wed 25/08/10
47	Submit Thesis	0 days	Fri 27/08/10	Fri 27/08/10

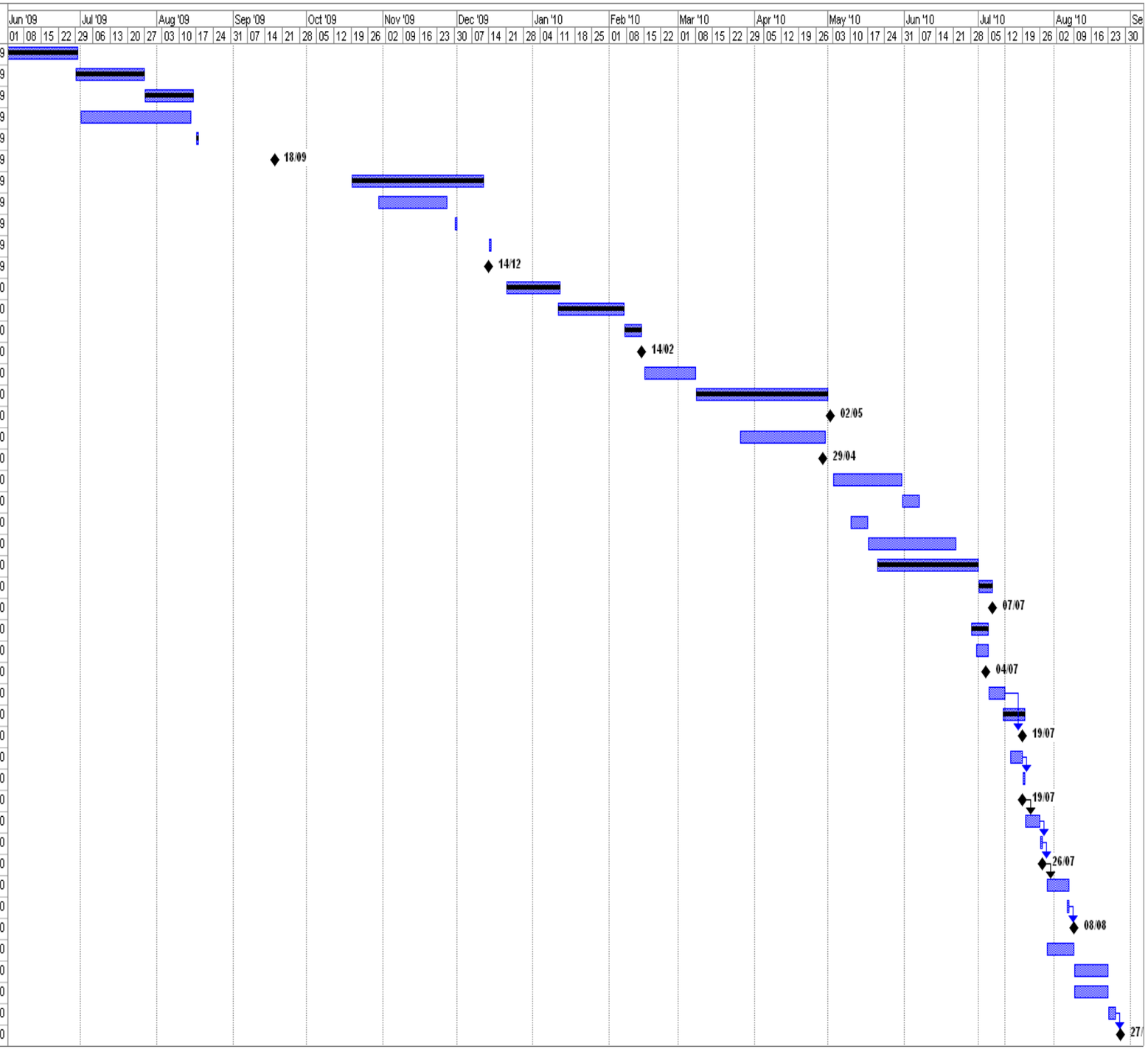

Figure 11 Project Plan

5.3 Evaluation of whether or not the project meets project goals

The main aim of the thesis was to prove that requirement specification documenting was a neglected phase of the project lifecycle. The main conclusion found from the thesis was that requirement specifications are being implemented in projects within industries, but that this stage of the project lifecycle is still misunderstood, and therefore not implemented in the correct manner. This, in turn, does not provide all the benefits that requirement specifications can produce if done correctly.

5.4 Lesson Learned for project management

As part of the lesson learned activity, an evaluation of whether or not the project met project goals was assessed. What went right, and what went wrong, were also assessed.

5.4.1 Lesson Learned Introduction

The purpose of this event was to document how the thesis project ran, and to establish any improvements that could be obtained for any future thesis projects. This exercise was carried out to ensure that any key lessons learned during this project are captured and can act as a benchmark for other thesis projects.

5.4.2 What Went Well?

Table: 3 *What went well during the thesis project*

Number	Description	Comment/Lessons Learned
1	All interviews occurred, and were a productive experience	The same process of interviewing would be used in future projects
2	Training and relevance to my profession was beneficial	Future projects should be in an area of benefit to me, or any other authors.
3	The introduction of a requirement specification gathering system as discussed in Chapter 4, was not in the scope of the original thesis project	This was a major risk in the project due to time lines, but proved relevant in explaining and categorising requirement specifications.
4	All examination of requirement specification was performed in cross-functional industries, and examined the experience of personnel from a number of industries, in relation to requirement specification and standards.	This allowed a cross-functional group to be examined, in relation to the relevant topics.
5	Schedule was met.	The project was delivered on time

Number	Description	Comment/Lesson Learned
6	Any relevant standards were researched and examined which aided in the thesis project, and also general knowledge of the area of requirement specification.	Therefore providing a solid base work.
7	Performing the literature review early aided in progressing the following chapters	Perform literature review early on the thesis project, set the ideas in places for the remaining chapters.

5.4.3 What Didn't Go Well?

Table: 4 *What didn't go well during the thesis project*

No	Description	Comment/ Lesson Learned
1	Researching of standards and requesting of standard documents took longer than expected.	Perform standard research earlier in the thesis project.
2	Idea Paper required much improvement, when referenced to complete a thesis statement.	More research and preparation required for the idea paper, to aid with the thesis statement preparation and approval.

No	Description	Comment/ Lesson Learned
3	Questionnaire was re-written after first review by two independent personnel, to aid with clarity of question and expected response.	Clarity of question was still required during the questionnaire distribution process and five questionnaires were omitted from the results and analysis section due to incompleteness or incorrectness.
4	Finding clarity of required dates from the university.	There was a medium risk of missing the deadline of the 14th of February for annotated bibliography submittal, due to a missed communication in December about the revised date. Continuous checking of information on university system required for future projects.

The project ended with a thesis project that was delivered on time and in a reviewed American Psychological Association (APA) formatted manner, consisting of six chapters and four appendices. In all, on review the thesis project did meet the project goals that were explained and set out in the idea paper, thesis statement and scope of thesis project.

5.5 Project variables

The main project variables are the following:

1. Timelines

The timelines were assessed at the beginning of the thesis (August 2009). These timelines aimed to have the thesis project completed by June 2010. This was moved to mid-

August, due to higher commitments to second year‘s modules than anticipated. The timeline was kept active and up to date, via the project plan as per Figure 11 and Appendix C.

2. Case Study Questionnaire

The case study questionnaire was performed from January 2010 to May 2010. It was submitted to 25 people in February, and all questionnaires were received and reviewed by April 2010. Interviews were performed with additional people that worked in a relevant industry, with user requirement specification documents on a weekly / monthly basis. The main variables in this exercise were people‘s perception of the questions being asked, how to overcome this problem, and how to avoid errors were given clarity on the questionnaire.

3. Case Study Personnel

The case study personnel were gathered from four different industries, to give a higher degree of sampling from different prospectives. The case study was performed in a manner that tried to remove all bias in relation to requirement specification gathering and documenting. This was aided by the manner in which questions were asked and ensured that interviewees were allowed to answer all questions in full and with detail.

4. Statistical variations

The statistical variations were limited in the statistical tools that were used for the results and analysis chapter. The main error in the statistical analysis was the rounding of decimal places to whole numbers.

5.6 Project Summary

The annotated bibliography, literature review, methodology, questionnaire and case study were performed throughout the timelines of second-year course modules. The remaining chapters of introduction, results and analysis, project history, and conclusion were performed

following the completion of the course modules from July to August 2010. The literature review discussed in detail the relevant standards to the thesis project and the generation and documenting of requirement specification, taking into account risk analysis, lifecycle and case studies. The advantages and disadvantages of implementing standards during requirement specification analysis were discussed, taking into account risk management, development and process improvements.

The methodology chapter went on to discuss the process of procedures taken during the questionnaire and interviews for the case study part of the thesis project. The results and analysis section used statistical analysis on the data gathered, and then further discussed the statistical results and analysis. The thesis was then concluded within the project history chapter, and the conclusion chapter. However, the main obstacles to completing the thesis were time lines and constraints.

Chapter 6 – Conclusions

6.1 Main Findings

The purpose of this study is to examine the hypotheses that the requirement specification stage of the project lifecycle of computerised systems is a neglected stage. Therefore, this research examines the techniques used for requirement gathering and managing, including a case study, with influence from the relevant standard bodies. In an influential book, Robertson & Robertson (2005) wrote, "requirement specification importance has grown significantly over the past few years". The main fact to be investigated is if the requirement specifications are implemented with relevant standards as guidance, would the errors and problems found during the phase of the project lifecycle be minimised. In a major article reporting on this research, Raja (2007) wrote that "6% of the errors found in a project are associated with the requirement specification stage of the project lifecycle." According to Robertson & Robertson (2005), they "estimate 60% of the errors are associated with the requirement specifications stage of the project lifecycle being neglected." This research supports these two theories by examining the effect that standards have on reducing the errors occurring in requirement specification stages of a project. It extends the research by acknowledging the importance of requirement specification in present-day projects, and their effect on project schedule, budget and quality.

6.2 Summary of Results Integration

A series of structured figures are used to demonstrate the results calculated during the results section of the research. It is found that the participants are spread equally over the medical device, pharmaceutical, software and electronics industries. The research group consisted of 90% of the participants using requirement specifications on a daily basis, and 10% on a weekly basis, 40% referencing industrial standards, and 95% referencing both industrial and

in-house standards. This demonstrates that standards within today‘s industry are referenced on a regular basis, either in their original form, or as in-house standards, that have been documented in a more relevant manner for the industry of use.

The research proceeded to examine gathering of requirement specifications. Figure 1-4 examine documentation for requirement specifications, including the first model, which discusses the main purpose of requirement specification in relation to project lifecycle, documentation activities and other activities. Figure 2 demonstrates the use of models and figures, benchmarking and use cases to aid with requirement gathering. Figure 3 follows on to examine requirement gathering in a more detailed structure, examining eight different tools that can be used to aid with requirement gathering, including stakeholder analysis, task analysis, and content of use analysis. Finally, the last figure within the section of requirement gathering (Figure 4) relates to the contribution and input of stakeholders at this stage of the requirement specification.

The results then proceed to examine the documenting of requirement specification. This section consists of three figures, including a figure describing the use of frameworks within the requirement specification documentation, and discusses criteria frameworks and prioritisation frameworks. The documentation section also contains a figure, discussing the documenting of requirements in a testable manner, and finally the use of languages within the documentation section of requirements. This is the one section of the requirement specification that was most neglected by the participants, and not fully explored or understood. The results then proceed to discuss the main conflicts experienced by the participants, in connection with level differences, inter-group difference, and personality differences, which are demonstrated as part of Figure 5.

The results section then proceeds into the interview section. This section discusses the indicated correlation between hours spent on requirement specifications and overall costs of the project. The main indication from this section is that there is a linear relationship, with the size of the project increasing as the requirement specifications also increase. This is also demonstrated in the first table of the section. The section then proceeds to discuss requirement creep and requirement leakage. Figure 10 demonstrates that requirement creep is responsible for delays within the project‘s schedule. It also found that on average between 40% - 60% of the requirement creep is due to insufficient or inaccurate requirements documentation. This is in line with the findings of both Raja (2007) and Robertson & Robertson (2006), which is disappointing as relevant standards have been used on these projects. The section also examines the use of prototypes and manuals for ensuring all requirement specifications are gathered and documented. The final section of the results is in relation to distinguishing criteria and risk management.

6.3 Summary of Conclusions Integration

The main conclusions drawn from the literature review are that there are four standards (IEEE STD 15288, IEEE STD 12207, IEEE STD 1233 and ESA Board for Software Standardisation and Control) which are of relevance to the requirement specification of the project lifecycle. The requirements definition is the most crucial part of the project. Incorrect, inaccurate, or excessive definition of requirements must necessarily result in schedule delays, wasted resources or customer dissatisfaction. The requirements analysis should begin with business or organisational requirements, and translate these into project requirements, which are evident from the results and analysis examined in this research. The three main types of problem with requirements are:

1. Definitions written in natural language which lack clarity,
2. Requirements confusion from requirements not being fully traceable,
3. Requirement non–co-operation.

By examining requirements, it is shown that good requirements practices can accelerate software development. The process of defining business requirements aligns the stakeholders with shared vision, goals, and expectations.

The main conclusion drawn from the methodology chapter is that a case study, with statistical analysis, is the correct analytical method to use for this research. The case study is based on a questionnaire that was distributed to 25 participants from four different industries. The case study is further examined by interviewing five people from the four different industries. The one problem experienced during the interview process was time limitations. The main areas to be examined as part of the case study are industry standards, requirement specification generation, documentation, and generation of requirement specification incorporating standards. The identification of patterns was found via statistical analysis. Finally, an overall portrait of the case was constructed, and conclusions were drawn from the case study and statistics. A research report was also prepared for the case study as part of Chapter 4

The main findings from Chapter 4 (Results and Analysis) are that requirement specifications are not necessarily a neglected phase of the project lifecycle. It is evident that personnel perform requirement specifications in combination with standards, but not necessarily in the correct manner. From the research and analysis, it was found that some industries only fulfil attention to requirement specification because they are regulated to do so but do not necessarily give them the correct level of detail. Although other companies are not as highly regulated in relation to the requirement specifications, they still implement standards within the

requirement specification, but in a more controlled manner. On reflection, requirement specification and relevant standards are used to aid with the generation and documenting of requirement specification. Therefore, it is not a neglected stage of the project lifecycle, but it is a stage that is misunderstood and is not always completely reflective of the project itself.

The main findings of Chapter 5 (Project History) are that the thesis consistently performed to schedule from June 2009 to August 2010. The project history also discussed the lessons learned throughout the project. The main lessons learned for the management of the project were as follows:

1. All interviews were productive.
2. Relevance to my profession was beneficial.
3. An examination of requirement specification was performed in cross-functional industries.
4. Schedule was met.
5. All relevant standards were researched.
6. Questionnaire required a large degree of accuracy and clarity.

Finally, the last Chapter 6 (Conclusion) discusses the general learning and findings of the results and conclusions, including recommendations and reflections.

In summary, one of the foremost benefits of having proper user requirements is that the project is able to be planned and estimated, thereby saving the likelihood of cost and time overruns. From the research this is evident. Requirement specification are performed by following the relevant standards, but projects still overrun and cost extra, mainly due to the fact that the requirement specification stage of the project lifecycle is still a misunderstood area. I strongly believe that successful projects can only be produced through competent and careful

requirement gathering. Requirement gathering provides us with the opportunity to learn about the proposed projects. It also gives the client the opportunity to look at the project blueprint on paper.

Based upon the requirement specification, the project manager will work with the client / stakeholder to translate these requirements into a detailed project specification. At this stage, the team will resolve any conflicting views of the overall projects goals, define the interaction of the project with each member of the requirements team, and then go about establishing a cost, quality and deliverable schedule for the project. After the initial requirements-gathering exercise, they are able to calculate the cost and deliverable of the project. One word of warning, at this early stage of the project the estimates are likely to be vague as they are based on some degree of uncertainty. The further the project progresses, the more solid the requirements become and the degree of uncertainty is reduced. Additionally, all captured requirements can be stored within a common central database for use by the project team.

The main lesson learned in relation to the requirement specification, in combination with relevant standards, from this research is:

1. Clarify any ambiguities – it will save time.
2. If the scope of the project is well defined, the project will succeed.
3. Developers assume they know what the users / stakeholders want.
4. Manager should not decide what the system should look like and exclude the users / stakeholders – involve them from the start.
5. If requirements are incomplete and are inaccurate at the start of a project, developers are more likely to see the project failing than succeeding.

6. Requirements must be reviewed by all stakeholders on the project and not be limited to just the client.

7. If there is uncertainty about gathering user requirements, follow a requirement gathering methodology / standard.

8. Have knowledgeable and experienced analysts or developers assist in the requirements definition or exploratory phase of the project.

6.4 Study Limitations

The present study offers several important findings to the research area. Yet there are some limitations to the study as well. The first limitation is the sampling. The sample size is 25 participants. The proportion of participants within the four relevant industries was excellent, as 25% of the participants were from each industry. There were a proportionate number of men and woman (10 women to 15 men), 5 of the men were removed from the study for various reasons; so the ratio was now 1:1. The results are important because this is a population that continually works with requirement specifications.

The second limitation to this research was the lack of response to the distinguishing of requirement specifications. This question was answered by all participants, but the results were not consistent, and demonstrated that 50% used informal and 50% used formal procedures. Upon further investigation, during the interview process it was found that distinguishing between requirements was performed but not in a formal/informal manner as discussed in the questionnaire. A better procedure in the future for the questionnaire would be to produce a more accurate questionnaire.

A final limitation of the study was in only interviewing five participants, and witnessing requirement specification gathering as part of their daily jobs. Due to the questionnaire data

being fraught with problems, and to people‘s perception of the questions, it would have been more beneficial to have performed a more comprehensive research. This would include actual physical witnessing of the requirement specification gathering, documenting and incorporating standards. The problem here is that committing to more interviews and witnessing more processes would require a great deal of extra time and willing participants.

6.5 Recommendations and Reflections

The main recommendation that can be drawn from this research is the importance of time management. To further expand on this topic, I would perform further research via the interview process. One other area that I feel requires further research is the relevance of standards to the requirement specification stage of the project lifecycle. From the literature review research, it is evident that there are numerous standards of relevance to this stage of the project lifecycle, with four standards being of major relevance. From my field research I found that standards are implemented, but vary with the perception and discretion of the personnel documenting the requirements. This area requires further research into the usability and application of the standards to requirement specification, with special attention to the language used, within the standards, and the generality of the standards. Another area requiring further research is the indication of the linear correlation between project size, and requirement specification document size. This area is also relevant to requirement creep, and requirement leakage, which requires further examination in association with standards.

On reflection, the questionnaire produced very useful data and is an excellent building block to the interview process and observation. Although the interview process is time-consuming, it yielded much useful information. The interview process area bore the least

accurate results, due to the questionnaire being open to interpretation and misunderstanding, due to poor clarity within the questionnaire.

The main lessons learned from the process, in relation to beginning a thesis research again, would be to ensure time and work constraints are not such a major stumbling block as shown in this thesis. I would also ensure that the questionnaire is written in a more scientific and self-explanatory way.

6.6 Final Conclusion

In conclusion, careful user requirement gathering is an essential first step in any project. In my experience, the time spent upfront gathering detailed user requirements eliminates unnecessary delays, improves system quality and greatly reduces requirement creep and its associated cost overruns. More importantly, it also takes courage to really listen to what users / stakeholders want, as many organisations aren‘t prepared to listen to employees, suppliers, contractors and consultants. Today more and more companies are being forced to adopt the use of the IEEE standards, ISO standards or ESA standards, implying that there is a greater emphasis being placed on gathering user requirements, which is beneficial to both the cost and quality of a project. The one major lesson taken, from this research is to remember that effective requirements management is the first step to becoming more mature in the development process of project management.

References

Aliport, H. and Isazadeh, A. (2008). Software Reliability Assessment Based on a Formal Requirement Specification. *Journal of IEEE, 311–316*. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=4581454.

Berry, D. (2003). User‘s Manual as a Requirement Specification: Case Studies. *Computer Society Journal,* 1-25. Retrieved from http://se.uwaterloo.ca/~dberry/FTP_SITE/reprints.journals.conferences/users.man.pdf.

Bourque, P. & Lethbridge, T. (2002). Improvements to the Guide to the Software Engineering Body of Knowledge (SWEBOK) and to the Software Engineering Education Body of Knowledge (SEEK). *Journal of Software Technology and Engineering Practice, 10,* 1. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=1267594.

Branstad, M. & Powell, P. (1982). Software Engineering Project Standards. *IEEE Transactions on Software Engineering, 10,* 73–78. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=5010201.

Brooks, F. (1995). *The Mythical Man month* (4th ed.). Boston, MA: Addison-Wesley.

Carew, D., Exton, C. and Buckley, J. (2005). An Empirical Investigation of the comprehensibility of requirement specifications. *Software Engineering Journal, 10*, 256-265. Retrieved from

http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=1541834.

Child Care and Early Education Research Connections. (n.d.). Retrieved from http://www.childcareresearch.org/childcare/datamethods/preexperimental.jsp;jsessionid=F3629DA363FBBD7B33411A769172579F.

Emmet, L. and Bloomfield, R. (1997). Viewpoints on improving the standards making process: Document Factory or Consensus Management. *IEEE Journal, 51*, 207–216. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=595972.

ESA Board for software standardisation and control. (1991). ESA Software Engineering Standards. *ESA PSSS Journal, 2,* 1 -330. Retrieved from http://www.esa.int/TEC/Software_engineering_and_standardisation/TECA5CUXBQE_0.html.

Glass, R. (2009). Doubt and Software Standards. *IEEE Computer Society Journal, 1,* 1-2. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=5222806.

Glinz, M. (2008). A Risk Based, Value Oriented Approach to Quality Requirements. *IEEE Computer Society Journal, 21,* 34–41. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=4455629.

Harauz, J. and Poon, P. (1999). System Engineering in the Twenty First Century. *ISESS'99 Panel, 1,* 1-2. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=767662.

Heitmeyer, C. and McLean, J. (1983). Abstract Requirements Specification: a New Approach and Its Application. *IEEE Transaction on Software Engineering Journal, 5,* 580-590. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=1703098.

Hesslink, H. (1995). A comparison of standards for software engineering based on DO-178B for certification of avionics system. Elsevier Science Microprocessors and Microsystems, 19, 559 -563. Retrieved from http://www.sciencedirect.com.libgate.library.nuigalway.ie/science?_ob=ArticleURL&_udi=B6V0X-3YB9T0J-1&_user=103680&_coverDate=12%2F31%2F1995&_alid=1429652798&_rdoc=1&_fmt=high&_orig=search&_cdi=5658&_sort=r&_docanchor=&view=c&_ct=18&_acct=C000007922&_version=1&_urlVersion=0&_userid=103680&md5=28a19ffdc69ae78f1e2692a48568b2dd.

Hunt, L. B. (1997). Getting the Requirements Right – a Professional Approach. IEEE Journal, 1, 464-472. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=615536

IEEE. (1998). IEEE STD 830, IEEE Recommended Practice for Software Requirements Specifications. *IEEE Computer Society Standard*, 1-39. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=720574.

IEEE. (2004). IEEE STD 1012 IEEE Standard for Software Verification and Validation, *IEEE Computer Society Standard*, 1-120. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=1488512.

IEEE. (2009). IEEE STD 1016 IEEE Standard for Information Technology—Systems Design—Software Design Descriptions. *IEEE Computer Society Standard*, 1-40. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=5167255.

IEEE. (2006). IEEE STD 1074, IEEE Standard for Developing a Software Project Lifecycle Process. *IEEE Computer Society Standard*, 1-116. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=1665059.

IEEE. (2005). IEEE STD 1220, Systems engineering — Application and management of the systems engineering process. *IEEE Computer Society Standard*, 1-101. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=1511885.

IEEE. (1994). IEEE STD 1228, IEEE Standard for Software Safety Plans. *IEEE Computer Society Standard*, 1-23. Retrieved from

http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=467427.

ISO / IEC. (1998). IEEE STD 1233, IEEE Guide for Developing System Requirement Specifications. *IEEE Computer Society Standard*, 1-36. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=502838.

IEEE. (1998). IEEE STD 1362, IEEE Guide for Information Technology—System Definition—Concept of Operations (ConOps) Document. *IEEE Computer Society Standard,* 1-21. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=761853.

IEEE. (2000). IEEE STD 1471, IEEE Recommended Practice for Architectural Description of Software-Intensive Systems. *IEEE Computer Society Standard*, 1-23. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=875998.

IEEE. (1999). IEEE STD 1517, IEEE Standard for Information Technology—Software Lifecycle Processes—Reuse Processes. *IEEE Computer Society Standard*, 1-51. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=1511021.

IEEE. (2001). IEEE STD 1540, IEEE Standard for Software Lifecycle Processes—Risk Management. *IEEE Computer Society Standard,* 1-30. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=914365.

ISO/IEC. (2008). ISO/IEC 12207 Systems and software engineering — Software lifecycle processes. *IEEE Computer Society Standard,* 1-138. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=669648.

ISO/IEC. (2001). ISO/IEC 15288 The System Lifecycle Process standard for the 21st century. *IEEE Computer Society Standard,* 1-20. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=1490129.

Jackson, M. (1995). *Software Requirements and Specification, A lexicon of practice, principles and prejudices* (3rd ed.). Harlow, England: Addison-Wesley.

Jang, H. (1994). A knowledge Based Analyser for requirement specification analysis. *IEEE Journal, 1,* 276-282. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=346480.

Laitenberger, O., Beil, T. and Schwinn, T. (2002). An Industrial Case Study to examine a non-traditional Inspection Implementation for Requirement Specification. *Journal from IEEE Symposium on Software Metrics, 2,* 1-10., Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=1011329.

Lee, Y., Lee, J. & Lee, Z. (2002). Integrating Software Lifecycle Process Standards with Security Engineering. *Computer and Security Journal, 21*, 345–355. Retrieved from http://www.sciencedirect.com.libgate.library.nuigalway.ie/science?_ob=MImg&_imagekey=B6V8G-46692F5-F-

H&_cdi=5870&_user=103680&_pii=S0167404802004133&_orig=search&_coverDate=08%2F01%2F2002&_sk=999789995&view=c&wchp=dGLbVlW-zSkzV&md5=4d26deaa5cfd77718500e0c47666ffff&ie=/sdarticle.pdf.

Leedy, P. D. and Ormond, J.E. (2005). *Practical Research: Planning and Design* (9th ed.). Upper Saddle River: Pearson Prentice Hall.

Loucopoulos, P. and Champion, R. (1990). Concept acquisition and analysis for requirement specification. *Software Engineering Journal, 1,* 116-124. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=54395.

Marriott, P. & Siefert, D. (1992). IEEE Software Engineering Standards Status and Perspective. *IEEE Journal, 1,* 1518. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=195208.

Pozagj, Z., Sertie, H., Boban, M. (2003). Effective requirement specification as a precondition for successful software development project. *International Conference Information Technology, 25,* 669-674. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=1225420.

Rada, R. (2001). Standardising Management of Software Engineering Project. *IEEE Journal, 10,* 1-7. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=926527.

Raja, A. (2007). Empirical Studies of Requirements Validation Techniques. *IEEE Journal, 1*, 1-9. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=4909209.

Robertson, S & Robertson, J. (2006). *Mastering the Requirements Process* (2nd ed.). Upper Saddle River, NJ: Addison-Wesley.

Singh, R. (2000). International Standard ISO/IEC 12207 Software Lifecycle Processes. *IEEE Journal, 40,* 1-18. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=471194.

Walker, A.J. (1998). Improving the quality of ISO 9001 audits in the field of software. *Information and Software Technology, 40*, 865-869. Retrieved from http://www.sciencedirect.com.libgate.library.nuigalway.ie/science?_ob=MImg&_imagekey=B6V0B-3VKBS26-J-3&_cdi=5642&_user=103680&_pii=S0950584998001049&_orig=search&_coverDate=12%2F01%2F1998&_sk=999599985&view=c&wchp=dGLzVzz-zSkzS&md5=ddbebdf4489c3177d844e23e3e9df93f&ie=/sdarticle.pdf.

Tse, T.H., (2000). Towards Harmonised Professional Standards for Software Engineers: Constraints, Conflicts and Concessions. *IEEE Journal, 7,* 346–347. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=884746.

Yau, S., Bae, D. and Yeon, K. (1994). An approach to Object oriented Requirements Verification in Software Development for Distributed Computing Systems. *IEEE Journal, 7,* 96-102.

Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=342825.

Yau, S. and Liu, C. (1988). An approach to software requirement specification. *IEEE Journal, 2,* 83-88. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=17154.

Zenmyot, T., Kobayashi, T. and Sackit, M. (2008). A Technique to Check the Implementatability of Behavioural Specifications with Frameworks. *Software Engineering Conference, 15,* 111-118. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=4724538.

Appendix A – Annotated Bibliography

Requirement Specification stage of the project lifecycle of computerised systems & the standards that can be implemented: An Annotated Bibliography

Jang, H. (1994). A knowledge Based Analyser for requirement specification analysis. *IEEE Journal, 1,* 276-282. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=346480.

In this paper, the author proposes a knowledge-based analyser of a formal requirement specification language of real time systems (RT-FRORL) for requirement specification analysis. As part of the introduction the RT-FRORL analyser is discussed including how it is based on an underlying verification framework and associated verification methodologies. The remainder of the paper discusses the RT-FRORL as a requirement specification language of real time systems and the analysis used as part of the requirement specification and how the framework originates from an integration of rapid prototyping, operational specification and transformational implementation.

In conclusion, it was found that the RT-FRORL analyser is based on an underlying development and verification framework and associated verification methodologies. The verification methodologies consist of a combination of resolution refutation, anomaly detection matrix and algorithms methods.

Yau, S. and Liu, C. (1988). An approach to software requirement specification. *IEEE Journal, 2,* 83-88. Retrieved from

http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=17154

In this paper, an approach to software requirement specification using a structured bipartite inheritance network is presented. In the introduction and definitions sections a bipartite inheritance network is introduced which is a network with two different kinds of basic nodes: data entity and action, which are independent of each other and structured into an inheritance hierarchy. This approach is further discussed in the model for software requirement specifications section which states the advantages experienced in the software requirement specification.

In the discussion section, the authors have developed an approach to software requirement specification using a structured bipartite inheritance network. Their approach emphasises that data entities and actions be stated explicitly in the software requirement specification, and the data entities and actions be refined in a unified way.

Hunt, L. B. (1997). Getting the Requirements Right – a Professional Approach. IEEE Journal, 1, 464-472. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=615536

This paper describes a radical new approach to producing re-usable requirement specifications which achieves levels of clarity and precision hitherto unattainable. Above all the approach provides the ability to demonstrate clearly to a client the actual content of a specification as opposed to its supposed content - in an information sense. Such ability is vital if the professional engineer is to carry his client with him or if the client is

to be convinced when changes are required before any serious commitment to consequential or candidate design is made.

The paper discusses this approach as part of the introduction, problems areas and methodology including general basic framework evolution and finally real application. This paper draws attention to the fact that, due to the rising perception of the need to produce improved specifications, appropriate tools are now being developed.

Aliport, H. and Isazadeh, A. (2008). Software Reliability Assessment Based on a Formal Requirement Specification. *Journal of IEEE, 311–316*. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=4581454.

Assessment of reliability using characteristics of software development process phases is one of the discussions which have been attracting more and more attention during the recent three decades. Most of the techniques and models use the results of design implementation and test phases. There are only a few models that are employed at the early phase of software development. Assessment of software reliability in the early phases of the software development process, however, is very important for better prognosis and management of risks. In this paper, the authors propose an approach for early software reliability assessment based on software behavioural requirements. They use a formal method called view-charts to specify the behaviour of software systems. In conclusion, the authors introduce a new approach for representing the behaviour of software systems and using it for early software reliability assessment.

ISO/IEC. (2001). ISO/IEC 15288 The System Lifecycle Process standard for the 21st century. *IEEE Computer Society Standard,* 1-20. Retrieved from

http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=1490129.

The ISO/IEC 15288 Standard discusses the system lifecycle process for the 21st Century. The main agenda of the standard is in relation to the background, history, ISO/IEC 15288 overview and benefits. The standard discusses in further detail the scope of the ISO/IEC, applicability of ISO/IEC, use of the standards and lifecycle processes and lifecycle of the system processes. It discusses the structure of the standard, title, purpose, outcomes and activity. The standard concludes by discussing the benefits of using ISO/IEC 15288. The main conclusion drawn from the standard is the ISO/IEC 15288 is a key reference for any situation where systems are concerned and the lifecycle models are a key concept for successful systems.

ISO/IEC. (2008). ISO/IEC 12207 Systems and software engineering — Software lifecycle processes. *IEEE Computer Society Standard,* 1-138. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=669648

This International Standard establishes a common framework for software lifecycle processes, with well-defined terminology, that can be referenced by the software industry. It applies to the acquisition of systems and software products and services, to the supply, development, operation, maintenance, and disposal of software products and the software portion of a system, whether performed internally or externally to an organisation. Those aspects of system definition needed to provide the context for software products and services are included. Software includes the software portion of firmware. The standard consists of an explanation of the application of the standard, software lifecycle processes

and software specific processes. The standards also assist in the area of requirement specification by providing (via an annex) a tailoring process description and a process reference model.

ISO / IEC. (1998). IEEE STD 1233, IEEE Guide for Developing System Requirement Specifications. *IEEE Computer Society Standard*, 1-36. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=502838

The standard provides guidance for the development of a set of requirements that will satisfy an expressed need. In the standard a set of requirements will be called the System Requirements Specification (SyRS). Developing a SyRS includes the identification, organisation, presentation and modification of the requirements. The standard guide addresses conditions for incorporating operational concepts, design constraints and design configuration requirements into the specification as per the SyRS development process overview section. The standard also addresses the necessary characteristics and qualities of individual requirements as per the well formed requirements section. The standard also contains a SyRS development section.

ESA Board for software standardisation and control. (1991). ESA Software Engineering Standards. *ESA PSSS Journal, 2,* 1 -330. Retrieved from http://www.esa.int/TEC/Software_engineering_and_standardisation/TECA5CUXBQE_0.html

This standard discusses in great detail the purpose, structure and classes of the standards. The standard is split into different parts to allow for easier reading. Part 1 is product standards which discuss the software lifecycle, user requirement definition

phases, software requirement definition phase, architectural design phase, detailed design and production phase, transfer phase operations and maintenance phase. Part 2 discusses management of the lifecycle, software project management and software configuration management, software validation and verification and software quality assurance. The main chapters that are of emphasis to this thesis are Chapter 2 and 3 of the user requirement definition phase and software requirements definition phase respectively.

IEEE. (2005). IEEE STD 1220, Systems engineering — Application and management of the systems engineering process. *IEEE Computer Society Standard*, 1-101. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=1511885

The standard defines the interdisciplinary tasks that are required throughout a system‘s lifecycle to transform stakeholder needs, requirements and constraints into a system solution as per general requirements of the standard. This standard is applicable to the thesis research as it is intended to guide the development of systems for commercial, government, military, and space applications. The standard specifies the requirements of the Systems Engineering Process (SEP) and its application throughout the product lifecycle. The standard focuses on the engineering activities necessary to guide product development while ensuring that the product is properly designed to make it affordable to produce, own, operate, maintain and eventually to dispose of, without undue risk to health or the environment. The content of this standard describes an integrated approach to product development, which represents the total technical effort that is covered as part of the application of system engineering throughout the system lifecycle and system engineering process.

IEEE. (1994). IEEE STD 1228, IEEE Standard for Software Safety Plans. *IEEE Computer Society Standard*, 1-23. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=467427

This standard applies to the Plan used for the development, procurement, maintenance and retirement of safety-critical software, as part of the standard's overview and content of the software safety plan. The standard requires that the plan be prepared within the context of the system safety program.

IEEE. (1998). IEEE STD 1362, IEEE Guide for Information Technology—System Definition—Concept of Operations (ConOps) Document. *IEEE Computer Society Standard,* 1-21. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=761853

The standard discusses the format and content of the concept of the operation document. The ConOps is a user oriented document that describes system characteristics to be delivered to a system from the user's viewpoint. The guide consists of nine clauses. The first clause is in relation to scope including identification, document overview and system overview. The second clause is about referenced documents and the third clause is about the current system or situation. The fourth clause is about justification for and nature of changes which include priorities, desired changes and user classes. The fifth clause is for the concepts of proposed changes and the sixth clause is operational scenarios. The seventh clause is a summary of impacts including operational impacts, organisational impacts and impacts during development. The eighth clause is the analysis

of proposed systems where a summary of improvement, disadvantage limitations and alterations and trade off is considered. The final clause is in relation to notes and appendices.

IEEE. (2000). IEEE STD 1471, IEEE Recommended Practice for Architectural Description of Software-Intensive Systems. *IEEE Computer Society Standard*, 1-23. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=875998

This recommended practice addresses the architectural description of software-intensive systems. The scope of this recommended practice encompasses those products of system development that capture architectural information. The standard consists of five chapters. Chapter 1 is an overview of the scope, purpose, intended users and conformance to this recommended practice. Chapter 2 is about references and Chapter 3 about definition. The main chapter of interest to the thesis research is Chapter 4, which is in relation to the conceptual framework including architectural description, stakeholders, activities in connection to lifecycle and uses of architectural description. Chapter 5 is in relation to architectural description practices. This aids with the thesis research as it discusses the importance of architectural design with requirement specifications.

IEEE. (2006). IEEE STD 1074, IEEE Standard for Developing a Software Project Lifecycle Process. *IEEE Computer Society Standard*, 1-116. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=1665059

This standard provides for creating a software project lifecycle process (SPLCP). This methodology begins with the selection of an appropriate Software Project Lifecycle

Model (SPLCM) for use on the specific project. It continues through the definition of the software project lifecycle (SPLCP). The standard explains the background and importance of SPLCP in the overview of the standards and the key concepts. The main area of the standard that is relevant to the thesis research is that it emphasises the importance of requirements being identified and gathered before the SPLCP is established. The standard then proceeds to discuss the implementation of SPLCP.

IEEE. (1999). IEEE STD 1517, IEEE Standard for Information Technology—Software Lifecycle Processes—Reuse Processes. *IEEE Computer Society Standard*, 1-51. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=1511021

This standard provides a common framework for extending the software lifecycle processes to include the systematic practice of software reuse. It specifies the processes, activities and tasks to be applied during each phase of a software lifecycle to enable a software product to be constructed from assets. This standard also specifies the processes, activities and tasks to enable the identification, construction, maintenance and management of assets. The standard consists of the following general areas, application of reuse, integration of reuse and reuse supporting phase of the project.

IEEE. (1998). IEEE STD 830, IEEE Recommended Practice for Software Requirement Specifications. *IEEE Computer Society Standard*, 1-39. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=720574

This recommended practice describes recommended approaches for the specification of software requirements. It is divided into five clauses. Clause 1 explains

the scope of this recommended practice. Clause 2 lists the references made to other standards. Clause 3 provides definitions of specific terms used. Clause 4 provides background information for writing a good Software Requirement Specification SRS. Clause 5 discusses each of the essential parts of an SRS. This recommended practice also has two annexes, one which provides alternate format templates and one which provides guidelines for compliance with IEEE 12207.1-1997.

This standard is of relevance to the thesis research as it is a recommended practice for writing software requirement specifications. It describes the content and qualities of a good software requirement specification (SRS) and presents several sample SRS outlines. The standard is aimed at specifying software to be developed and can be applied to assist in the selection of in-house and commercial software products.

IEEE. (2001). IEEE STD 1540, IEEE Standard for Software Lifecycle Processes—Risk Management. *IEEE Computer Society Standard,* 1-30. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=914365

This standard prescribes a continuous process for software risk management. Clause 1 provides an overview and describes the purpose, scope and field of application, as well as prescribing the conformance criteria. Clause 2 lists the normative references informative references are provided in Annex E. Clause 3 provides definitions. Clause 4 describes how risk management may be applied to the software lifecycle. Clause 5 prescribes the requirements for a risk management process. There are several informative annexes. Annex A, Annex B, and Annex C recommends content of three documents: Risk Management Plan, Risk Action Request and Risk Treatment

Plan. The standard describes a process for the management of risk during software acquisition, supply, development, operations and maintenance. It is intended that both technical and managerial personnel throughout an organisation apply this standard.

The main aim of this standard and its purpose in relation to thesis research is for it to provide software suppliers, acquirers, developers and managers with a single set of process requirements suitable for the management of a broad variety of risks. This standard does not provide detailed risk management techniques, but instead focuses on defining a process for risk management in which any of several techniques may be applied.

IEEE. (2004). IEEE STD 1012 IEEE Standard for Software Verification and Validation, *IEEE Computer Society Standard*, 1-120. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=1488512

This Verification and Validation (V&V) standard is a process that addresses all software lifecycle processes including acquisition, supply, development, operation and maintenance. This standard is compatible with all lifecycle models; however, not all lifecycle models use all of the lifecycle processes listed in this standard.

Software V&V processes determines whether the development products of a given activity conform to the requirements of that activity and whether the software satisfies its intended use and user needs. This determination may include analysis, evaluation, review, inspection, assessment and testing of software products and processes.

The user of this standard may invoke those software lifecycle processes and the associated V&V processes that apply to the project. This standard applies to software being acquired, developed, maintained, or reused [legacy, modified, commercial off-the-shelf (COTS), non-developmental items (NDI)]. The term software also includes firmware, microcode and documentation. Software V&V processes consists of the verification process and validation process. The standard is applicable to the research topic as it demonstrates the use of requirement specification further along in the project lifecycle (i.e. validation and verification). The standard discusses the importance that correct and accurate requirement specification can have on the validation verification of a project and the negative affect if not correctly in place.

IEEE. (2009). IEEE STD 1016 IEEE Standard for Information Technology—Systems Design—Software Design Descriptions. *IEEE Computer Society Standard*, 1-40. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=5167255

This standard describes software designs and establishes the information content and organisation of a Software Design Description (SDD). This standard is intended for use in design situations. These situations include traditional software construction activities where design leads to code, and ~~"r~~everse engineering" situations where a design description is recovered from an existing implementation. This standard can be applied to commercial, scientific or military software that runs on digital computers. Applicability is not restricted by the size, complexity or criticality of the software. The standard consists of definition, conceptual model for software design, design description, information content and design viewpoints. The main points of interest to the thesis topic

are in chapter 4 which relates to design views, elements, overlays, rationale and languages, with special emphasis on stakeholders and design in a combination which is applicable to requirement specification generation.

Hesslink, H. (1995). A comparison of standards for software engineering based on DO-178B for certification of avionics system. Elsevier Science Microprocessors and Microsystems, 19, 559 -563. Retrieved from http://www.sciencedirect.com.libgate.library.nuigalway.ie/science?_ob=ArticleURL&_udi=B6V0X-3YB9T0J-1&_user=103680&_coverDate=12%2F31%2F1995&_alid=1429652798&_rdoc=1&_fmt=high&_orig=search&_cdi=5658&_sort=r&_docanchor=&view=c&_ct=18&_acct=C000007922&_version=1&_urlVersion=0&_userid=103680&md5=28a19ffdc69ae78f1e2692a48568b2dd.

The DO-178B standard provides guidelines for software certification. Re-use of software is emerging, partly enabled by the integrated modular avionics concept and imposed by a reduction of lifecycle costs. The standards discussed in this paper are Do-178B, DOD-STD-2167A, ESA PSS-05-0 and IEC65A (secretarial) 122.

Every standard has its particular emphasis and way of dealing with the software lifecycle. DO-178B provides a high level view on the definition of phases. DOD-STD-2167A emphasises tests. ESA PSS-05-0 provides the most support in the requirements phase. Finally, IEC65A is a standard which does not reflect a specific emphasis within the lifecycle, although it could be argued that this standard has more involvement in the design phase than the other standards. Again, not all standards have the same scope. The design and coding phases may be omitted from the lifecycle of DO-178B. DO-178B and

DOD-STD-2167A include a survey of system requirements at the start of each development project (N.B. ESA and IEC65A have equivalent hardware standards). DO-178B specifies the target hardware, in advance of the development as a system requirement, whereas the other standards choose a suitable computer system in the course of the development process. ESA PSS-05-0 shows the most and the only explicit user involvement. It is the only standard that includes a phase which is fully the responsibility of the (future) user, the User Requirements Definition. The DO-178B prescribed software lifecycle phases are based on the waterfall model. The lifecycle phases of the other standards are based on the V-model. In conclusion it was found that all examined standards guide the software development process.

Hesslink graduated at the University of Groningen in information and artificial intelligence. He has performed a number of studies in the area of avionics software certification. This paper is relevant to the thesis research as it demonstrates the vast array of standards in the software engineering industry that directly affect other industries including avionics.

Harauz, J. and Poon, P. (1999). System Engineering in the Twenty First Century. *ISESS'99 Panel, 1,* 1-2. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=767662.

This paper discusses the history to the International Standard on System Lifecycle Processes, ISO/IEC 15288, and its completion and is distributed worldwide. It also defines a top-level cradle-to-grave lifecycle framework for managing modern systems configured with hardware, computers, software and humans. The main areas discussed in

the paper are how the standard can be applied to the acquisition, supply, development, operation and maintenance of systems, how the standard supports the process through configuration management and quality assurance. It also discusses the standard can be used as an internal framework by an enterprise and can be used in developing an agreement between two parties, or as a reference standard for lifecycle processes for further standardisation, guidance and tools development.

The paper further discusses on a number of issues, including the relationship between ISO/IEC 15288 (i.e., systems engineering lifecycle) and ISO/IEC 12207 (i.e., software engineering lifecycle) and how to maximise the efficient use of ISO/IEC 15288 and ISO/IEC 12207. It discusses what impact ISO/IEC 15228 has on the way systems engineering is performed in various countries and how to ensure the adoption of ISO/IEC 15288 as an International Standard will lead to wide usage of this Standard. Finally, to ensure that the revision of ISO/IEC 12207, subsequent to the publication of ISO/IEC 15288, will lead to consistency and interoperability with ISO/IEC 15288 and what approaches should be taken in gathering the experience of using ISO/IEC 15288.

John Harauz is a senior specialist in the Engineering Standards Department in Ontario Hydro‘s nuclear division. John has been involved in the development and regulatory licensing of safety-related and safety-critical real-time systems since 1978. He received the corporate New Technology award in 1995 for his contribution to developing innovative safety-critical software engineering technology. John is a member of the Association of Professional Engineers of Ontario and a technical expert in ISO/IECJTC1/ Software and System Engineering Conference (SC7) and IEC TC45. Dr. Peter Poon is the Telecommunications and Mission Services Manager for the French, German and Italian

space missions at the Jet Propulsion Laboratory, California Institute of Technology. He is a member of the IEEE Software Engineering Standards Executive Committee; a Technical Expert for ISO/IEC JTC1/SC7 and the United States of America Technical Advisory Group for SC7; and is Program Chair for SES'98. This paper is of major relevance to the thesis topic as it discusses software engineering standards in present day industries.

Singh, R. (2000). International Standard ISO/IEC 12207 Software Lifecycle Processes. *IEEE Journal, 40,* 1-18. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=471194.

The paper discusses the history in relation to the development and deployment of ISO/IEC 12207 standard. The paper then proceeds to discuss the basic concepts of the standards including the software lifecycle architecture, modularity, responsibility, lifecycle processes. The authors further discuss the primary processes including acquisition, supply, development, operation and maintenance. The supporting and tailoring processes are also discussed in detail. The author then proceeds to examine the guidance and interactions among the processes and proceeds onto the application of the standard to a project by examining the role of the lifecycle, organisational policies, system lifecycle, developments models and types of software, documentation and evaluation. In a case study the author examines the application of the standard to an organisation. In conclusion it found that the ISO/IEC 12207 is the first international standard that provides a complete set of processes for acquiring and supplying software

product and services. These processes may be employed also to manage, engineer, use and improve software throughout its lifecycle.

The author is a member of the federal aviation administration and actively works within the software industry in relation to standards. This paper is highly applicable to the research as it discusses in detail the ISO/IEC 12207 standard which has a major affect on the requirement stage of the project lifecycle.

Rada, R. (2001). Standardising Management of Software Engineering Project. *IEEE Journal, 10,* 1-7. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=926527

In a software engineering division of a company, the most important standards are those used for the management of the software engineering projects. Although numerous relevant *de jure* software engineering standards exist, national guidelines such as the Department of Defence‘s Capability Maturity Model and corporate standards, such as the Microsoft Solutions Framework exert a significant influence on the marketplace. A review of the existing standards shows significant similarity across them. The hypothesis is advanced as a major factor in determining that the adoption of one standard over another is the environment of the adopter. The following methods were employed by the author for one case study, two surveys and one content analysis of various companies. The results show that the choice of a software engineering management standard follows the preference of a major strategic partner or customer. A company that depends on Microsoft in important business ways is inclined to adopt the Microsoft Solutions Framework. Likewise a company that is a major customer of the United States

Department of Defence is inclined to use the Capability Maturity Model supported by the Department of Defence.

The author is based in the University of Maryland and is performing research in the department of information systems. The paper was presented as part of the IEEE conference in 2001. The paper is relevant to the research as it discusses the most important standards used in software engineering projects and which are applicable to requirement specifications.

Glass, R. (2009). Doubt and Software Standards. *IEEE Computer Society Journal, 1,* 1-2. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=5222806.

Standards for requirement documents are receiving criticism due to standard bodies consistently producing standards that fail the basic criteria for good engineering. The author added that standards should be based on established scientific results and best industrial practice and argued that our contemporary software standards tend not to meet those criteria. Standards should be based on scientific results and best industrial practice. Standards should be subject to evaluation to ensure they really work in the environment for which they‘re intended. All of this is difficult to attain because standards tend to be produced in a highly politicised environment. The author believes that both the process and nature of our software standards demand objective review and change.

The paper was published as part of the IEEE Computer Society paper in 2009. The author is editor of Elsevier‘s Journal of Systems and Software and is a visiting professor at Griffith University where he‘s involved with the Australia Resource Centre for

Complex System. The paper supports the thesis research as it discusses the problems associated with requirement gathering documents and their inconsistencies.

Marriott, P. & Siefert, D. (1992). IEEE Software Engineering Standards Status and Perspective. *IEEE Journal, 1,* 1518. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=195208

The IEEE Computer Society sponsors the development of software engineering standards and recommended practices for the software industry. These standards address the requirements, design, test, verification, validation, measurement, plans and documentation aspects of software engineering projects. To date, 11 of these standards have been approved by the IEEE and ANSI and are published as ANSI/IEEE standards. An additional 13 standards are currently in development. This presentation describes what the IEEE Computer Society and other standards organisations are doing to promote a disciplined approach to software engineering practice and how aerospace systems engineering organisations can apply these standards to their projects. The presentation also highlights the software standardisation activities of national and international standards organisations (ANSI, X3, ISO, IEC), government organisations (NBS, Department of Defence (DoD)), professional societies (IEEE, American Society Quality Control (ASQC), and ACM) and trade associations (AIA, EIA, and NSIA).

The authors are of computer technology background with Masters of Science in Software Information Technology. This paper is relevant to my work as it discusses the large variety of standards and their relevance to other areas and new additional standards.

Tse, T.H., (2000). Towards Harmonised Professional Standards for Software Engineers: Constraints, Conflicts and Concessions. *IEEE Journal, 7,* 346–347. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=884746.

The harmonisation of professional standards usually means an attempt to unify the standards among different nations or states. It is a necessary step towards the maturity of a profession because of two factors: (a) that professional standards have been developed independently in different nations, and (b) that the standards thus developed are not uniform among nations. The paper discusses this concept during the introduction and then begins to discuss constraints and conflicts by examining the body of knowledge, professional ethics, public interest and Asian concerns.

The author is based in the department of Computer Science and Information Systems in the University of Hong Kong. The paper is relevant as it discusses the professional standards that have been presented in different countries. The author further discusses a comparison between the standards.

Robertson, S & Robertson, J. (2006). *Mastering the Requirements Process* (2nd ed.). Upper Saddle River, NJ: Addison-Wesley.

Mastering the Requirements Process sets out an industry proven process for gathering and verifying requirements, with an eye toward today‘s agile development environments. The authors show how to discover precisely what the customer wants and needs while doing the minimum according to the projects level of requirement.

The key features of this book are in relation to the chapters explaining what the requirement is and the requirement process. The next chapter of reference is Chapter 11

which discusses the quality gateway and requirements quality. The final chapter of reference is Chapter 14 reviewing the specification with special attention to inspections and finding missing requirements.

The authors have many years of experience and help companies improve their requirement techniques and move into system development. The Robertsons are principles of the Atlantic Systems Guild, a well know consultancy specialising in the human dimensions of complex system building.

Jackson, M. (1995). *Software Requirements and Specification, A lexicon of practice, principles and prejudices* (3rd ed.). Harlow, England: Addison-Wesley.

The subject discusses in depth the requirement specification software. The requirements specification is explained in the format of top down development, dataflow diagrams and the distinction between what and how. The main important topics in relation to the research performed in the thesis are the principle of evaluating development methods, new approaches for capturing and describing requirements and specifications based on relationships between the software systems and the problem contact.

The author has worked in software for over 30 years. After ten years in consultancy he started his own company offering courses, tools, consultancy and project support. In 1992, he received an Honorary DSc from the University of West of England for his work on software development methods. He is a visiting professor of University of West England (UWE) and holds visiting chairs at several other universities. He is now an independent consultant in software development methods as well as consulting part time at AT&T Bell Laboratories.

Heitmeyer, C. and McLean, J. (1983). Abstract Requirement Specification: a New Approach and Its Application. *IEEE Transaction on Software Engineering Journal, 5,* 580-590. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=1703098.

As abstract requirement specification states system requirements precisely without describing a real or a paradigm implementation. Although such specifications have important advantages, they are difficult to produce for complex systems and hence are seldom seen in the "real" programming world. This paper introduces an approach to producing abstract requirement specifications that applies to a significant class of real-world systems, including any system that must reconstruct data that has undergone a sequence of transformations. It also describes how the approach is used to produce a requirements document for Secure Copy Protection (SCP), a small, but nontrivial Navy communications system. The specification techniques used in the SCP requirements document are introduced and illustrated with examples. In conclusion it was found that the existence of the SCP requirements document demonstrates that an abstract requirement specification is feasible. The future work is in demonstrating that the SCP requirements document will also serve as a model.

Constance L. Heitmeyer received the M.A. degree in history and the M.A. degree in mathematics from the University of Michigan, Ann Arbor. From 1969 to 1971 she was a Research Assistant at the University of Michigan where she developed communications software for the MERIT computer network. During 1972 to1973 she was an Assistant Professor in the Department of Computer Systems, Florida Atlantic University and Boca

Raton. In 1973 she joined the Naval Research Laboratory, Washington, DC, where she is currently head of the Message Systems Automation Section of the Computer Science and Systems Branch. John D. McLean received the B.A. degree in mathematics in 1974 from Oberlin College, Oberlin, OH, and the M.A. degree in philosophy in 1976 and the Ph.D. degree in philosophy and the M.S. degree in computer science both in 1980 from the University of North Carolina, Chapel Hill. The paper is relevant to the thesis topic as it discusses requirement specification systems and how best to distinguish and gather requirements. This paper is largely relevant to the case study to be performed as part of the thesis topic.

Carew, D., Exton, C. and Buckley, J. (2005). An Empirical Investigation of the comprehensibility of requirement specifications. *Software Engineering Journal, 10*, 256-265. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=1541834

It is a commonly held view by Software Engineers that informal requirement specifications are easier to comprehend than formal requirement specifications. Moreover, the training time required to gain a sufficient level of understanding in formal notations is unknown. This paper presents an empirical study carried out to compare the comprehensibility of two specifications, formal specification and an informal (or semi-formal) specification, in an attempt to quantify the amount of training needed to understand formal methods. The two specifications used implemented the same logic. The paper begins by discussing the specification and introducing the background to this study. The paper then proceeds to perform experimental design including participants,

materials and experimental procedure. The authors then consider what the influencing attributes are and the qualitative data analysis to be performed. The paper discusses the problems encountered in the experiment.

The future work will be to conduct empirical studies similar to this one with different training periods given to participants. Knowing the amount of training required to comprehend formal specifications is crucial to universities and industry. Also, further experiments investigating the comprehensibility of different levels of formality in specification languages are needed to provide information and guidelines to both research and industrial communities.

The authors are all based at University of Limerick and both Exton and Buckley have doctorates. The paper is relevant to the research as it discusses the requirement specification stage of the project lifecycle and examines in detail both informal and formal requirements

Yau, S., Bae, D. and Yeon, K. (1994). An approach to Object oriented Requirements Verification in Software Development for Distributed Computing Systems. *IEEE Journal, 7,* 96-102. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=342825.

The authors begin their discussion by drawing attention to the problem of many errors in the source code that can be traced to the errors in the requirement specification. It is especially important to have effective verification techniques for the requirement specification. In this paper, an approach to verification of object-oriented requirement specification (OORS) in software development for distributed computing systems is

presented. The approach taken, the requirement specification generated by object-oriented analysis is described using a formal specification language, which is transformed into an information tree. Then, the completeness and consistency of the requirement specification expressed in terms of the information tree is verified by comparing it with the original requirements statement.

Many errors are found in the source code stem from inconsistent or incomplete requirement specification hence, the requirements verification is a very important part in developing reliable software for the distributed computing environment.

In conclusion, it was found that currently it is assumed that the requirement statements are unambiguous. Ambiguities in the requirement statements may lead to different interpretations of the software system, thus making the verification more difficult. Further research is needed to deal with the verification of requirement statements containing ambiguities.

The authors are both American based in Arizona and Florida universities respectively. The authors work within the computer and information science departments. The paper is of relevance to the research as it discusses common errors that occur in requirement specification gathering and supports the thesis topic of requirement specification being a neglected stage of the project lifecycle.

Emmet, L. and Bloomfield, R. (1997). Viewpoints on improving the standards making process: Document Factory or Consensus Management. *IEEE Journal, 51*, 207–216. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=595972.

In this paper the authors describe an application of PERE (Process Evaluation in Requirements Engineering) to the standards process. PERE provides an integrated process analysis that identifies improvement opportunities by considering process weaknesses and protections from both mechanistic and human factor viewpoints. The resulting analysis identified both classical resource allocation problems and also specific problems concerning the construction and management of consensus within a typical standards making body. A number of process improvement opportunities are identified that could be implemented to improve the standards process. The paper discusses these topics under the following headings, _an introduction to PERE', _PERE', and _process capture' and _process analysis results of the case study' and _conclusion that consensus problems are the real barrier to timely standards production'. Ironically the present trend for more distributed working and electronic support (via email etc.) may make the document factory aspect of standards production more efficient at the expense of consensus building.

The authors of the paper are based in the London University and work within the computer science industry. The paper is relevant to the research as it discusses both requirement specification gathering and the standards that are applicable to this stage.

Pozagj, Z., Sertie, H., Boban, M. (2003). Effective requirement specification as a precondition for successful software development project. *International Conference Information Technology, 25,* 669-674. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=1225420.

Software development is closely connected to requirements. To enable development of' software systems that satisfies most customer demands in this work, the authors propose methodologies for obtaining efficient requirement management infrastructure on a software development project. They propose methodologies for requirement discovery and organisation according to competent areas available on the project and involved risks. They also propose methodology for managing change of' requirements during the software development project in order to enable prosperous project conclusion even if major requirement change occurs. Presented methodologies provide means of elective requirement management that can significantly improve quality of complex software systems.

To solve the problems discussed in the paper and improve requirement use on a software development project, the authors propose three methodologies. These methodologies form best practice for requirement management according to their experience and should be performed in all phases of a software development project.

The authors presented this paper as part of the information technology interfaces conference in Croatia. The authors are based in the University of Split in departmental areas including economics, research and development and facility of law. The paper supports the thesis research in relation to methodology for obtaining efficient requirement specification and the correct gathering methods.

Glinz, M. (2008). A Risk Based, Value Oriented Approach to Quality Requirements. *IEEE Computer Society Journal, 21,* 34–41. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=4455629.

The main issue discussed in this paper is that when quality requirements are elicited from stakeholders, they are often stated qualitatively, such as –the response time must be fast" or –we need a highly available system." However, qualitatively represented requirements are ambiguous and thus difficult to verify. The paper further discusses the three kinds of problems associated with this requirement, which includes system developers building a system that delivers less than the stakeholders expect and this result in stakeholder dissatisfaction and might in extreme cases render a system useless. The system developers build a system that delivers more than the stakeholders need and this result in systems that are more expensive than necessary. The developers and the customer disagree on whether the delivered system meets a given quality requirement and there is no clear criterion to decide who is right. The author further discusses this in the advantages and drawbacks to requirement specification and risk based analysis and presentation.

In conclusion it was found that risk-based, value-oriented treatment of quality requirements extends the classic approach of making every quality requirement measurable. When the authors consider the value that the specification of a quality requirement adds, quantification is clearly not always the best way to represent a quality requirement. The approach shown here helps treat quality requirements adequately over a wide range of project situations and so help advance software quality.

Glinz is a full Professor of Informatics at the University of Zurich. His main areas of research include requirements and software engineering in particular modelling, validation and quality in software engineering education. He has received a Dr. in

computer science. The research from this paper is applicable to the research thesis topic as it discusses the risk management application to requirement specification gathering.

Loucopoulos, P. and Champion, R. (1990). Concept acquisition and analysis for requirement specification. *Software Engineering Journal, 1,* 116-124. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=54395.

This paper proposes that this approach needs to be augmented by the use of informal models which assist in the very early steps of the requirement specification process, i.e. during elicitation and analysis of concepts about the application domain. To this end, the paper discusses the use of a knowledge representation formalism, which provides the necessary foundation for capturing and analysing concepts about an application domain, and a prototype system which assists in this process. The paper introduces its concepts and then proceeds to the requirement specification section. To fully demonstrate this concept, a knowledge representation model is used and an example of a concept elicitation session is discussed,

This paper in conclusion has argued that because the task of requirement specification is the most critical of all tasks in the software development lifecycle, this task needs special attention in terms of modelling paradigms and support tools. In particular, the authors argue that the very early stages of requirement capture (namely the elicitation of concepts about the application domain and the analysis of these concepts) need to be supported by an approach which permits the construction of informal models and the use of scenarios about the modelled phenomena. These objectives have been addressed in the context of an approach which makes use of conceptual graphs

implemented in a frame-based system as the underlying knowledge representation scheme. A prototype system has also been developed which provides a range of facilities which support concept elicitation and analysis.

The authors performed this research as part of the Analyst Assist project. This was a collaboration project including Data Logic, University of Manchester Institute if Science and Technology (UMIST), Ministry of Defence and Istel.

The research of the paper is relevant, due to the discussion around the requirement specification including analysis concepts. The paper also supports the requirement specification stage of the project lifecycle which is the most critical part of the lifecycle.

Zenmyot, T., Kobayashi, T. and Sackit, M. (2008). A Technique to Check the Implementatability of Behavioural Specifications with Frameworks. *Software Engineering Conference, 15,* 111-118. Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=4724538.

In software development with frameworks, it is essential to use the framework with which given software requirements are implemental. This paper focuses on use cases of the requirement specifications and proposes a technique to check whether the given use cases are implementable with the framework in the introduction and approach section of the paper. The paper also discusses problems in checking implementability. To check the implementability, the requirement specification has to be checked as well as equivalence of action sequences between the frameworks and the requirement specification. To this end, a novel approach based on a satisfyability problem for deriving the consistent truth assignments of the branch conditions is introduced including

the modelling behaviour. The approach can be incorporated in bi-simulation checking for assuring the equivalence of the action sequences and therefore, the implement-ability can be checked. The authors use case descriptions which are modelled on Labelled Transition Systems (LTS) in their approach. Furthermore, this paper shows a feasibility of the proposed technique by using Compositional Reach-ability Analysis as a mean of bi-simulation checking.

The authors are all based at the Tokyo Institute of Technology in Japan and are based in the department of computer science. The paper was presented as part of the IEEE Computer Society in 2008. The paper is relevant to the research as it discusses in detail the framework used to support software requirements, and also discusses techniques to support this area.

Laitenberger, O., Beil, T. and Schwinn, T. (2002). An Industrial Case Study to examine a non-traditional Inspection Implementation for Requirement Specification. *Journal from IEEE Symposium on Software Metrics, 2,* 1-10., Retrieved from http://www.ieeexplore.ieee.org.libgate.library.nuigalway.ie/stamp/stamp.jsp?tp=&arnumber=1011329.

Software inspection is one of the key enablers for quality improvement and defect cost reduction. Although its benefits are shown in many studies, a major obstacle to implement and use inspection technologies in software projects is the costs associated with it. In this paper, the authors present and examine a non-traditional inspection implementation at DaimlerChrysler AG. The design of this inspection approach evolved over time as part of a continuous improvement effort and therefore integrates specific issues of the project environment as well as recent research findings. In addition to the

description of the inspection approach, the paper presents quantitative results to characterise the suggested inspection implementation and investigates some of the essential hypotheses in the inspection area.

Throughout the case study, data was collected to get qualitative and quantitative insight in this approach. Since this was part of a real development project at DaimlerChrysler, it was impossible to perform a (quasi-)controlled experiment to investigate cause-effect relationships. Hence, most of the conclusions require further study in the context of a more controlled study. However, the data confirm that this kind of inspection ensures that the inspectors perform an adequate preparation and that the effort for this pays off in terms of defects detected. Moreover, the results allow for the conclusion that in this project two inspectors were useful, i.e., each of the inspectors contributes to the results. The data also shows that the number of physical pages as a size measure leads to different conclusions than a more content oriented measure. This represents a fruitful area of research. Future research should also take a closer look at the impact of the individual inspectors and the author on inspection results.

The authors are based in the experimental software engineering department within the Institute for Experimental Software Engineering. The paper supports the thesis research as it examines the cost, budget and quality of the requirement specification on the project lifecycle. It examines this aspect in detail by discussing the pro and cons.

Appendix B -Questionnaire

Question: 1 what sector is you working in?

Pharmaceutical industry □

Medical device industry □

Software Industry □

Electronics Industry □

Other (Please Specify) □

Question 2 Do you use or produce requirement specification during your daily work?

Yes □

No □

Plan to do so □

Question 3 Does your company implement external / regulatory standards in relation to requirement specifications?

Yes □

No □

Question 4 Does your company implement in house standards in relation to requirement specifications?

Yes □

No □

Question 5 Is there any one particular standard you would reference for requirement specifications?

Yes ☐

No ☐

Yes, please specify:

_____

_____

Question 6 For what purpose is the requirement specification intended?

Project Lifecycle ☐

Documentation ☐

Other ☐

Question: 7 What tools do you use when gathering requirements?

Benchmarking ☐

Use Case Workshops ☐

Models and Diagrams ☐

Question: 8 When gathering managing user requirements do you use any of the following techniques?

Stakeholder analysis □

Secondary market research □

Context of use analysis □

Task analysis □

Rich pictures □

Field study □

Diary keeping □

Video recording □

Question 10 Do you meet with the stakeholders before finalising the requirement specifications?

Yes □

No □

Question: 11 Once user data has been collected, do you use any of the following techniques to identify the needs of the user?

Focus group □

Interviewing where users □

Scenarios and use cases □

Evaluating existing or competitor system □

Brainstorm session‘s □

Storyboards □

Prototyping □

User cost benefit analysis □

Design guidelines □

Question: 12 When user requirements have been agreed on and require frameworks for user requirement specification, do you use any of the following frameworks?

Prioritisation □

Criteria Setting □

None □

Question: 13 Do you ensure that the originating stakeholder understands and agree with the written requirements before it is passed downstream to other departments?

Yes □

No □

Question 14 Do you write the requirements are a testable manner?

Yes □

No □

Question: 15 Do you write the requirements in:

Business language □

Technical language □

Or Mixture □

Question 16 Do you distinguish between formal and informal requirements?

Yes □

No □

Question: 17 What are the main conflicts that occur during requirement specification gathering?

Personality clashes □

Inter-group Prejudice □

Level differences □

Appendix C – Project History

The first revision of the project plan that was drafted on the 11th of December 2009 is as per Figure 12.

ID	Task Name	Duration	Start	Finish
1	Idea Paper Research	21 days	Mon 01/06/09	Mon 29/06/09
2	Thesis Advisor Search	20 days	Mon 29/06/09	Sun 26/07/09
3	Thesis Advisor Found and Discussion	15 days	Mon 27/07/09	Sat 15/08/09
4	Thesis Idea Paper preparation	33 days	Wed 01/07/09	Fri 14/08/09
5	Thesis Idea Paper Submittal	1 day	Mon 17/08/09	Mon 17/08/09
6	Thesis Idea Paper Submittal Deadline	0 days	Fri 18/09/09	Fri 18/09/09
7	Thesis Module MCT-624 Part 1	40 days	Mon 19/10/09	Fri 11/12/09
8	Thesis Statement Preparation	20 days	Fri 30/10/09	Thu 26/11/09
9	Thesis Statement Submitted to Thesis Advisor	1 day	Mon 30/11/09	Mon 30/11/09
10	Re-Submittal of thesis statement following feed back	1 day	Mon 14/12/09	Mon 14/12/09
11	Completion of thesis module MCT-624 Part 1	0 days	Mon 14/12/09	Mon 14/12/09
12	Annotated Bibliography Research	16 days	Mon 21/12/09	Mon 11/01/10
13	Annotated Bibliography Drafting	20 days	Mon 11/01/10	Sat 06/02/10
14	Annotated bibliography Review	7 days	Sun 07/02/10	Sat 13/02/10
15	Annotated Bibliography Submittal	0 days	Sun 14/02/10	Sun 14/02/10
16	Questionnaire drafting	15 days?	Mon 15/02/10	Sun 07/03/10
17	Thesis Module MCT-624 Part 2	40 days	Mon 08/03/10	Fri 30/04/10
18	Completion of thesis module MCT-624 Part 2	0 days	Sun 02/05/10	Sun 02/05/10
19	Chapter 2 Literature Review	25 days	Fri 26/03/10	Thu 29/04/10
20	Chapter 2 Literature Review Submittal to thesis advisor	0 days	Thu 29/04/10	Thu 29/04/10
21	Questionnaire Distribution	20 days?	Mon 03/05/10	Sun 30/05/10
22	Questionnaire Review	5 days?	Mon 31/05/10	Sun 06/06/10
23	Interview Preparation and Organisation	5 days?	Mon 10/05/10	Sun 16/05/10
24	Interview Process and review	26 days?	Mon 17/05/10	Mon 21/06/10
25	Chapter 3 Methodology	29 days	Fri 21/05/10	Wed 30/06/10
26	Chapter 3 Methodology Review	4 days	Thu 01/07/10	Tue 06/07/10
27	Chapter 3 Methodology Submittal to thesis advisor	0 days	Wed 07/07/10	Wed 07/07/10
28	Chapter 4 Results Section	5 days	Mon 28/06/10	Sun 04/07/10
29	Questionnaire and Interview Trending and Statistics	3 days?	Wed 30/06/10	Sun 04/07/10
30	Chapter 4 Results Section Review	0 days	Sun 04/07/10	Sun 04/07/10
31	Chapter 4 Analysis Section	5 days	Mon 05/07/10	Sun 11/07/10
32	Chapter 4 Analysis Section Review	7 days	Sun 11/07/10	Mon 19/07/10
33	Chapter 4 Results and Analysis Submittal to thesis advisor	0 days	Mon 19/07/10	Mon 19/07/10
34	Chapter 5 Project History	4 days	Wed 14/07/10	Sun 18/07/10
35	Chapter 5 Project History Review	1 day	Mon 19/07/10	Mon 19/07/10
36	Chapter 5 Project History Submittal to thesis advisor	0 days	Mon 19/07/10	Mon 19/07/10
37	Chapter 6 Conclusion	4 days	Tue 20/07/10	Sun 25/07/10
38	Chapter 6 Conclusion Review	1 day	Mon 26/07/10	Mon 26/07/10
39	Chapter 6 Conclusion Submittal to thesis advisor	0 days	Mon 26/07/10	Mon 26/07/10
40	Chapter 1 Introduction	7 days	Thu 29/07/10	Fri 06/08/10
41	Chapter 1 Introduction Review	1 day	Fri 06/08/10	Fri 06/08/10
42	Chapter 1 Introduction Submittal to thesis advisor	0 days	Sun 08/08/10	Sun 08/08/10
43	Compile Appendices	8 days	Thu 29/07/10	Sun 08/08/10
44	Format and Tidy appendices	10 days	Mon 09/08/10	Sun 22/08/10
45	Final Review and re-formatting	10 days	Mon 09/08/10	Sun 22/08/10
46	Preparation for Submittal of final thesis	3 days	Mon 23/08/10	Wed 25/08/10
47	Submit Thesis	0 days	Fri 27/08/10	Fri 27/08/10

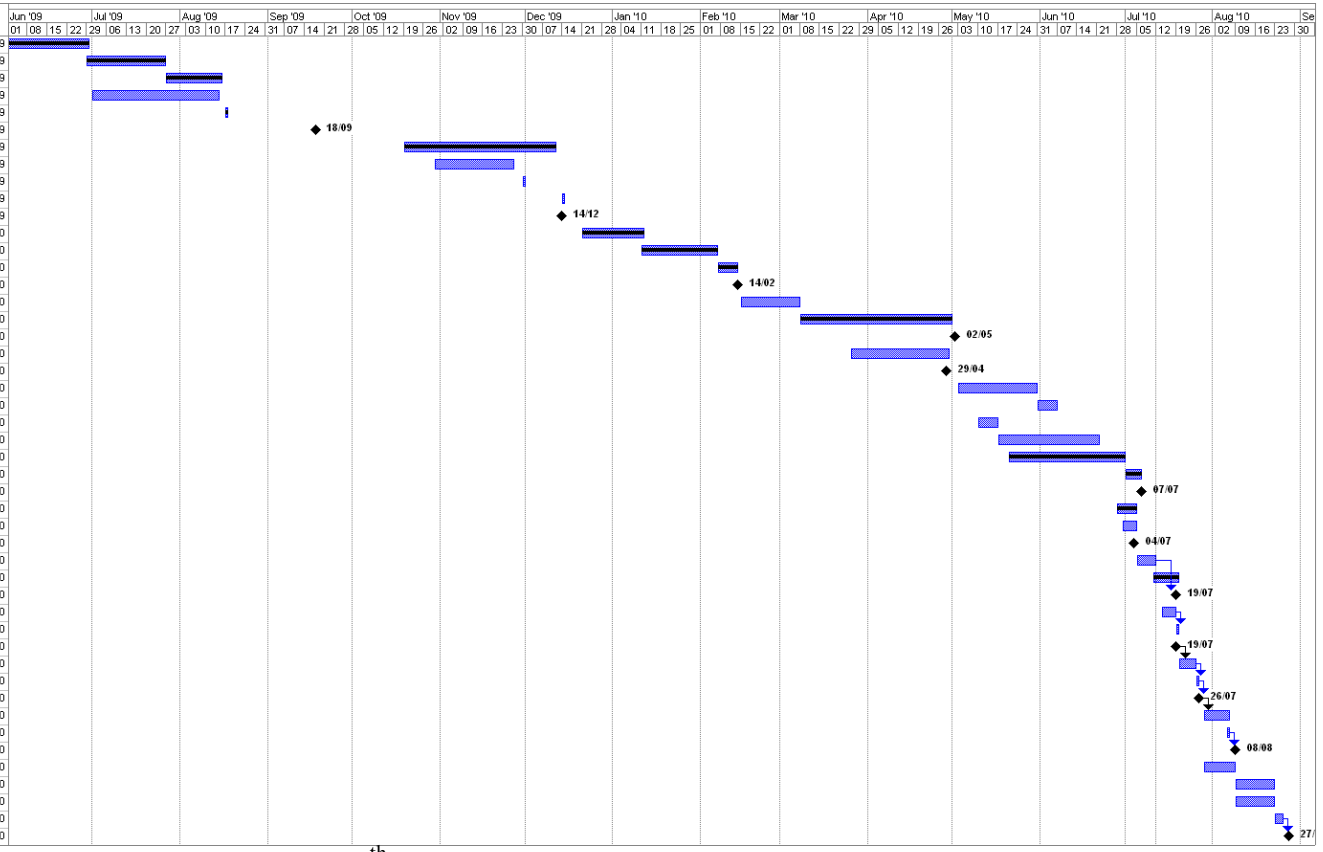

Figure 12. Project Plan as per 11th of December 2009.

The second draft of the project plan as drafted on the 14 of February 2010 as per Figure 13:

ID	Task Name	Duration	Start	Finish
1	Idea Paper Research	21 days	Mon 01/06/09	Mon 29/06/09
2	Thesis Advisor Search	20 days	Mon 29/06/09	Sun 26/07/09
3	Thesis Advisor Found and Discussion	15 days	Mon 27/07/09	Sat 15/08/09
4	Thesis Idea Paper preparation	33 days	Wed 01/07/09	Fri 14/08/09
5	Thesis Idea Paper Submittal	1 day	Mon 17/08/09	Mon 17/08/09
6	Thesis Idea Paper Submittal Deadline	0 days	Fri 18/09/09	Fri 18/09/09
7	Thesis Module MCT-624 Part 1	40 days	Mon 19/10/09	Fri 11/12/09
8	Thesis Statement Preparation	20 days	Fri 30/10/09	Thu 26/11/09
9	Thesis Statement Submitted to Thesis Advisor	1 day	Mon 30/11/09	Mon 30/11/09
10	Re-Submittal of thesis statement following feed back	1 day	Mon 14/12/09	Mon 14/12/09
11	Completion of thesis module MCT-624 Part 1	0 days	Mon 14/12/09	Mon 14/12/09
12	Annotated Bibliography Research	16 days	Mon 21/12/09	Mon 11/01/10
13	Annotated Bibliography Drafting	20 days	Mon 11/01/10	Sat 06/02/10
14	Annotated bibliography Review	7 days	Sun 07/02/10	Sat 13/02/10
15	Annotated Bibliography Submittal	0 days	Sun 14/02/10	Sun 14/02/10
16	Thesis Module MCT-624 Part 2	40 days	Mon 08/03/10	Fri 30/04/10
17	Completion of thesis module MCT-624 Part 2	0 days	Sun 02/05/10	Sun 02/05/10
18	Chapter 2 Literature Review	25 days	Fri 26/03/10	Thu 29/04/10
19	Chapter 2 Literature Review Submittal to thesis advisor	0 days	Thu 29/04/10	Thu 29/04/10
20	Chapter 3 Methodology	29 days	Fri 21/05/10	Wed 30/06/10
21	Chapter 3 Methodology Review	4 days	Thu 01/07/10	Tue 06/07/10
22	Chapter 3 Methodology Submittal to thesis advisor	0 days	Wed 07/07/10	Wed 07/07/10
23	Chapter 4 Results Section	5 days	Mon 28/06/10	Sun 04/07/10
24	Chapter 4 Results Section Review	0 days	Sun 04/07/10	Sun 04/07/10
25	Chapter 4 Analysis Section	5 days	Mon 05/07/10	Sun 11/07/10
26	Chapter 4 Analysis Section Review	7 days	Sun 11/07/10	Mon 19/07/10
27	Chapter 4 Results and Analysis Submittal to thesis advisor	0 days	Mon 19/07/10	Mon 19/07/10
28	Chapter 5 Project History	4 days	Wed 14/07/10	Sun 18/07/10
29	Chapter 5 Project History Review	1 day	Mon 19/07/10	Mon 19/07/10
30	Chapter 5 Project History Submittal to thesis advisor	0 days	Mon 19/07/10	Mon 19/07/10
31	Chapter 6 Conclusion	4 days	Tue 20/07/10	Sun 25/07/10
32	Chapter 6 Conclusion Review	1 day	Mon 26/07/10	Mon 26/07/10
33	Chapter 6 Conclusion Submittal to thesis advisor	0 days	Mon 26/07/10	Mon 26/07/10
34	Chapter 1 Introduction	7 days	Thu 29/07/10	Fri 06/08/10
35	Chapter 1 Introduction Review	1 day	Fri 06/08/10	Fri 06/08/10
36	Chapter 1 Introduction Submittal to thesis advisor	0 days	Sun 08/08/10	Sun 08/08/10
37	Compile Appendices	8 days	Thu 29/07/10	Sun 08/08/10
38	Format and Tidy appendices	10 days	Mon 09/08/10	Sun 22/08/10
39	Final Review and re-formatting	10 days	Mon 09/08/10	Sun 22/08/10
40	Preparation for Submittal of final thesis	3 days	Mon 23/08/10	Wed 25/08/10
41	Submit Thesis	0 days	Fri 27/08/10	Fri 27/08/10

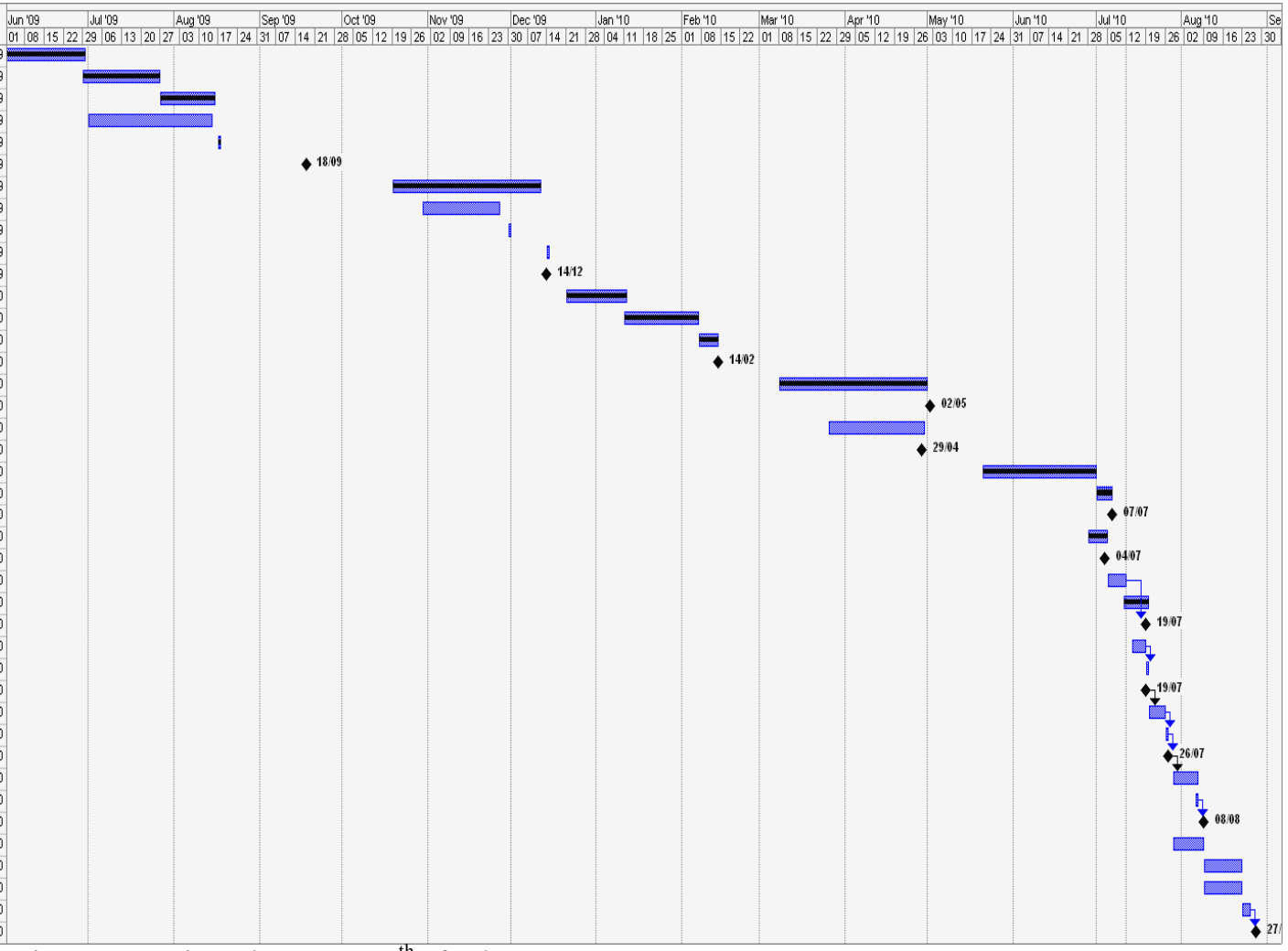

Figure 13. Project Plan as per 14th of February 2009

Glossary

ACM	Association of Computing Machinery
AIA	Aerospace Industries Association
ANSI	American National Standards Institute
APA	American Psychological Association
ASQC	American Society Quality Control
BOM	Bill of Materials
BSSC	Board for Software Standardisation and control
C2	Chi Square
CFR	Code of Federal Regulations
COTS	Commercial off-the-shelf
DoD	Department of Defence
EHS	Environment, Health and Safety
EIA	Electronics Industries Association
ESA	European Space Agency
FDA	Food and Drugs Administration
IEC	International Electro-technical Commission
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Organisation of Standardisation
LTS	Labelled Transition system
MES	Manufacturing Execution Systems
N	Number of test cases
NBS	National Bureau of Standards

NDI	Non-development items
NSIA	National Standards of Authority Ireland
NUI Galway	National University Institute Galway
OPA	Organisational Process Assets
OORS	Object Oriented Requirement Specification
P	Probability
RT- FRORL	Requirement Specification Language of Real-Time Systems
SC7	Software and System Engineering Conference
SCADA	Supervisory Control And Data Acquisition
SCP	Secure Copy Protection
SDD	Software design description
SEP	System engineering process
SPLCP	Software Project Lifecycle Process
SPLCM	Software Project Lifecycle Model
SPLC	Software Project Lifecycle
SRS	Software Requirement Specification
SWEBOK	Software Engineering Body of Knowledge
SyRS	System Requirement Specification
UMIST	University of Manchester Institute of Science and Technology
US	United States
UWE	University of West England
V&V	Verification and Validation
WG7	Working Group 7