QUALITY-FOCUSED DESIGN OF MEDICAL IoT QUALITY MANAGEMENT SYSTEMS

A TECHNICAL REPORT SUBMITTED TO AUCKLAND UNIVERSITY OF TECHNOLOGY
IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF COMPUTER AND INFORMATION SCIENCES

Supervisor

Associate Professor Roopak Sinha

Professor Andrew Lowe

February 2021

By

Shihui Han

School of Engineering, Computer and Mathematical Sciences

Abstract

Risk management of medical equipment before it is put into service is essential to reduce the risk of medical devices failing. Manufacturers of medical devices are required to minimize the risks associated with the use of medical devices. As medical devices are a mission critical product, any degree of error can result in patient death or health compromise. The purpose of this study is to develop a quality management system based on medical device standards to help medical device manufacturers more effectively reduce the risk of medical device use and provide 100% reliable medical products.

A system literature review (SLR) was implemented to demonstrate the connection between medical device standards (ISO 13485, ISO 14971, IEC 62366 and IEC 62304) and software architecture documents (ISO 25010 and 4+1 Views Model) as well as the relationship between recent reference medical device software architectures and software architecture documents. The main objective of the SLR is to comply with the requirements of the medical device standard in order to reduce the risk of medical device use and to select the best medical device related architecture that supports the architecture documents.

ISO 14971 is the standard related to risk management for medical devices. In this study, we used the risk management process from ISO 14971 and selected three quality attributes from ISO 25010 including, learnability, user error protection and modifiability to design the software architecture and develop a quality management system. The second outcome of this research is to develop a Risk Management system to help

medical device manufacturers reduce the risk of using medical devices. The process of risk management was based on the risk management process in ISO 14971.

Contents

List of Tables

List of Figures

Attestation of Authorship

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the qualification of any other degree or diploma of a university or other institution of higher learning.

Signature of student

Publications

Han, S., Sinha, R. Lowe, A. (2020). Assessing support for industry standards in reference medical software architectures. In IECON 2020 the 46th annual Conference of the IEEE Industrial Electronics Society (p.3403-3407).

Acknowledgements

This research has been an encompassing challenge for me, coupled with the unexpected Covid-19 outbreak that we have encountered. I am grateful to those who helped me during this year's research work. First of all, I would like to thank my supervisors, Associate Professor Roopak Sinha and Professor Andrew Lowe. They have always been able to clear my mind when I needed academic help. I also would like to thank my wife Jing, for her financial, spiritual and life support during the past year. Without her help, I could not have finished the research. Finally, I would like to thank all the researchers at EMSOFT for their friendship and support.
Shihui Han, March, 2021

Chapter 1

Introduction

1.1 Introduction

The development of mission-critical systems can be a complicated experience. As software failures in medical devices may lead to devastating outcomes, a number of standards were established to manage the development of these products in the field of medical devices. Regulations and industry standards based on regulatory requirements for information technology products. Regulatory requirements are becoming more prominent, and whether industry standards have this flexibility to be able to capture regulatory requirements as they arise on the basis that regulators will continue to adopt regulations. (Regan, Mc Caffery, Mc Daid & Flood, 2013).

It is a complicated procedure to develop safety-focused software. As devastating results can be caused by errors in the software of a medical device, there are a number of relevant standards regulating software development in the medical device area.

This research involves four aspects:

- The relationship between medical device standards including ISO 14971 (2019), ISO 13485 (2016), IEC 62366 (2015) and IEC 62304 (2015) and software architecture related documents: 4+1 Views Model (Meng et al., 2010) and ISO 25010

(2011).

- The extent to which medical device standards support reference software architectures.
- Design novel quality management system structure according to the requirements of medical device standards.
- Develop a medical device risk management system.

The section 1.3 turns these aspects of the research into four research questions.

1.2 Background

The Medical IoT is positively impacting the medical field. It can help reduce costs, improve treatment, make disease diagnosis faster, proactive treatment, and reduce errors. However, medical device IoT systems, just like any other system, come with unavoidable risks. To reduce these risks, which can cause different levels of harm to the user, and develop excellent IoT systems for medical devices, designers and developers need to comply with standards such as ISO 13485, ISO 14971, IEC 62366 and IEC 62304. Also, the introduction of the international standard ISO 14971, which provides the requirements for risk management processes motivate producers to recognize and manage the risks linked to medical devices under development. Furthermore, ISO 13485 introduces the Quality Management System (QMS), and this study will demonstrate a prototype of the QMS. ISO 25010 and 4+1 Views Model are architecture-related standards and models that QMS will design system architectures based on.

1.2.1 Quality Management System

QMS is a collection of business processes that concentrate on continuously meeting customer demands and increasing customer satisfaction. It is designed to align with the purpose and strategic direction of the organization (ISO 9001, 2015). It is represented by the organization's purpose and expectations, policies, procedures, recorded inputs and the resources required to implement and maintain it. In earlier times, quality management systems were focused on the predictable results of industrial product lines, using basic measurements and sample randomization. As the 20th century approached, labour inputs were generally the most expensive inputs in a majority of industrially-oriented countries, so the emphasis switched to collaboration and dynamics in the team, particularly in signalling problems early through continuous quality improvement loops. QMSs are often merged with initiatives for sustainability and transparency in the 21st century, as the satisfaction of investors and customers and the perception of quality are growing in relation to these factors. Among quality management systems, the ISO 9000 family of standards is possibly the most widely implemented - The ISO 19011 audit system addresses both standards and involves quality and sustainability and their integration.

1.2.2 Medical Device Standards

1. **ISO 13485:2016** specifies the demands of a quality management system and the need for organisations to provide evidence of their capabilities to deliver medical devices and relevant services in a manner that consistently meets customer and applicable regulatory requirements. These organizations may be engaged in any one or more phases of the life cycle, including the design and development, manufacture, storage and distribution, installation, or service of medical devices, as well as design and development or supply-related activities. ISO 13485:2016

may also be utilised either by providers or outside parties delivering products, which includes the provision of services related to quality management systems to such organisations (ISO13485, 2016).

2. **ISO 14971:2019** establishes the principles, terminology, and procedures for medical devices to do risk management, which includes software and in vitro diagnostic medical devices used as medical devices. The procedures described in this document are designed to assist medical device producers to identify the risks linked to medical devices, to evaluate and assess the associated risks, to control these risks and to monitor their control efficiency (ISO14971, 2019).

3. The international standard **IEC 62304** - Medical Device Software - Software Lifecycle Process is a standard that defines the lifecycle needs of medical software and software development within medical devices. The standard has been accepted in the EU and the US and can therefore be used as a benchmark to meet the regulatory requirements of both markets (IEC62304, 2015).

4. **IEC 62366-1:2015** provides a protocol to manufacturers for analysing, specifying, developing, and evaluating the usability of medical devices as it relates to safety. This human factors-related usability engineering protocol permits manufacturers to estimate and reduce the risks linked to correct use and misuse. It can be used to recognize but not measure or minimize the risks that are related to abnormal use (IEC62366-1, 2015).

1.2.3 4+1 Views Model

Kruchten (1995) stated that the 4+1 view model (Figure 1.1) is intended to illustrate the architecture of a software-intensive system, based on using a number of concurrent views. For each of the five views, some element of the system is prominently featured

and described together with a diagram to accompany it. These multiple views can be used to solve the issues of different stakeholder concerns independently. It can picture the architecture from various perspectives and expose the essential views to diverse stakeholders. These views have adopted an architecture-centric, scenario-driven way of designing, iterating on the development process.

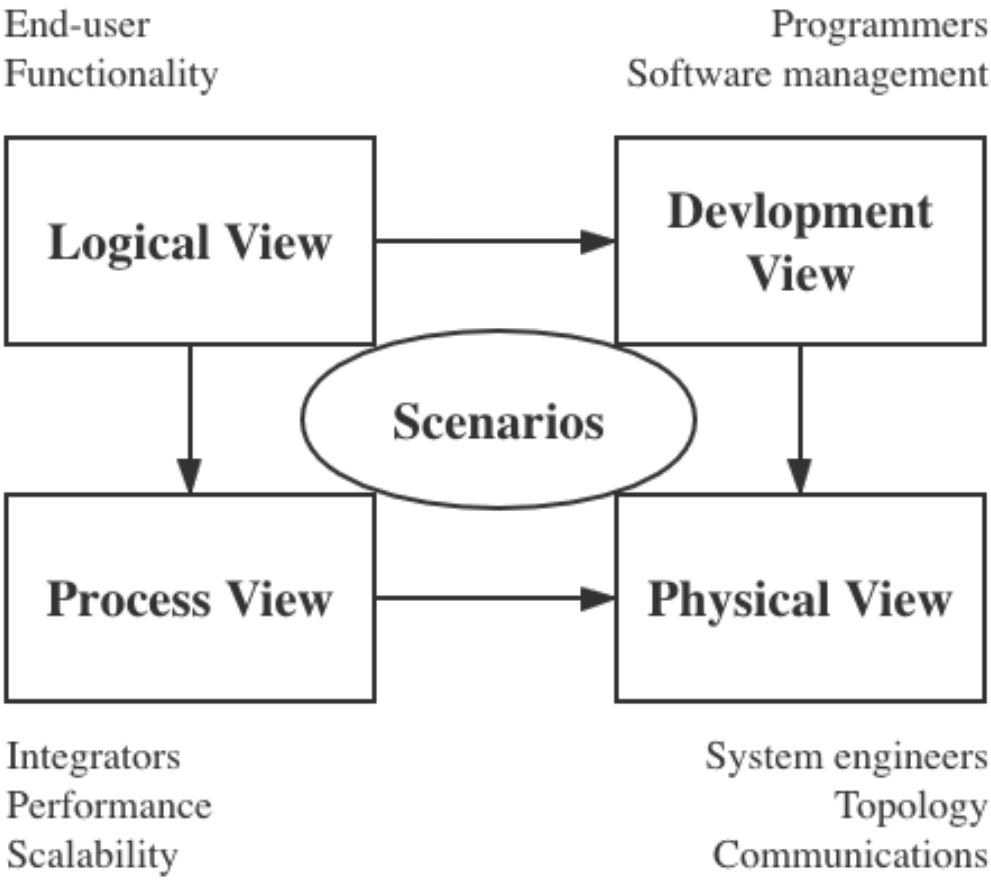

Figure 1.1: 4+1 Views Model (Meng et al., 2010)

1.2.4 ISO 25010

Raharja and Siahaan (2019) introduced that ISO / IEC 25010 is composed of eight quality characteristics, which are divided into 31 quality sub-characteristics. Every

characteristic from this product quality model enables measurement and evaluation. However, the model does not contain guidelines for measuring and evaluating these characteristics. The product quality model provides a representation of a checklist of the quality factors of the system, which have to be monitored and measured based on special requirements to satisfy assurance.

1.3 Research Questions

There are four research questions in this research:

RQ1 How relevant are the requirements from IEC 62366:2015, ISO 13485:2016, ISO 14971:2012 and IEC 62304:2006 in relation to the concrete ISO/IEC 25010:2011 attributes and views on product quality in the 4+1 Views framework?

RQ2 To what extent to which the current reference software architecture for the development of medical device software provides the support for the quality and perspectives specified in RQ1?

RQ3 How can a novel QMS be architected to support the requirement identified in RQ1?

RQ4 How efficient is the QMS for managing requirements from medical standards?

1.4 Solution and Contribution

The solution developed to address the four research questions associates medical device-related standards, the sub-characteristics of ISO 25010, and the views in 4+1 views model. Afterwards, a quality management system was developed following the risk management process of ISO 14971. The solution presents the architecture of QMS in

Chapter 4 and a case study of QMS in Chapter 5 to answer the research questions. The contributions of this thesis are from three major perspectives.

1. **A Systematic Literature Review on Reference Medical Architectures and Medical Standards.** To answer the RQ1 and RQ2, this research reveal a systematic literature review (SLR). The work done by other researchers in the field of medical architecture is described in SLR. Also, The requirements in medical standards were classified and refereed these requirements to software architectural documents.

2. **A Novel Architecture for Quality Management System based on medical standard.** To answer the RQ3, a novel architecture which can be applied to develop a quality management system was created. To make sure the novelty, the second supervisor provided architectural significant requirements based on medical risk management standard. Attribute-driven design method was used to decompose each QA scenario to complete a comprehensive architecture.

3. **Developing A Quality Management System.** To answer the RQ4, a case study of using quality management system to do risk management was conducted. This case study can show how efficient is the QMS for managing medical risk management standard and help medical device manufacturer to do risk management. The source code of QMS can be found in our GitHub repository `https://github.com/SimonHan1126/Master_QMS_system`.

1.5 Significance

This research includes multiple outcomes including a systematic literature review, designing standard-based architecture and a QMS prototype. In the process of writing systematic literature review, medical device software architectures are linked to

architecture-related documents (e.g., ISO 25010 and 4+1 Views Model). Also, the requirements of the standard with each sub-feature of ISO 25010 and each view of the 4+1 Views Model were mapped and documented. This will be useful for future researchers in the development of medical device software. The architecture of this study is based on the medical device risk management standard and uses the attribute-driven method, thus developing a unique architecture. In addition, this study culminated in the development of a QMS that allows medical device manufacturers to complete the entire risk management process in ISO 14971.

1.6 Thesis Structure

The arrangement of the rest of this thesis is as follows. Chapter 2 provides a systematic literature review to figure out the architectural works of other researchers on medical and the connections between medical device standards and architectural documents. Chapter 3 deals with the research methodology we used to answer the last two research questions. Chapter 4 shows the architecture designing by using attribute-driven design method. Chapter 5 displays a case study of quality management system. Chapter 6 evaluates the architecture which created in chapter 4 and answer the third and fourth research questions. Chapter 7 offers a summary to this thesis.

Chapter 2

Systematic Literature Review

2.1 Introduction

The field of medical device design and development is attracting the attention of a growing number of relevant researchers. Although there are quite a few recent research papers dealing with medical device standards, there is still a gap where these studies do not sufficiently support the requirements of medical device standards from a software architecture perspective.

The software architecture usually affects the ability of a complex system to display required qualities and quality attributes (Lundberg, Bosch, Häggander & Bengtsson, 1999). The product quality model referred to in ISO/IEC 25010: 2011 can help address issues involved in software development projects, software engineering specifications, and the assessment of non-functional requirements, and consists of eight characteristics and 31 sub-characteristics (Raharja & Siahaan, 2019).

Independent and high-level evaluation of design and risk to confirm that the completed software architecture demonstrates the desired quality has been a critical input to the software architecture.

The 4+1 Views model (Meng et al., 2010) is an industry-standard framework to help

architects build software-intensive systems. The four architecture views include logical, physical, development and process to illustrate various perspectives of the architecture. The logical view provides a description of the design of an object model as the software designer designs the system in the object-oriented method, the process view shows concurrency, synchronization, and operation, the physical view shows the mapping of software to hardware, and the development view shows the organizational structure of the software (Kruchten, 1995).

Medical device software should be designed and developed in accordance with medical standards to ensure the safety of the software. The medical device software should be designed and developed to match medical standards and guarantee the security of the software. The model created by IEC 62304:2006 contains the planning, collection of requirements, implementation, validation, assembly testing, systematic tests and publication events. Furthermore, it provides life cycle requirements in medical software development. ISO 14971 delivers standards for medical device risk management. Otherwise use ISO 3485:2016 for quality management, an extension of the ISO 9001 standard for medical quality control (Laukkarinen, Kuusinen & Mikkonen, 2017). In medical device design and development, IEC 62366:2015 standard enables the unification of usability engineering processes (UEP) (Bras Da Costa, Beuscart-Zéphir, Christian Bastien & Pelayo, 2015).

We conducted this systematic literature review to answer two research questions:

RQ1 How relevant are the requirements from ISO 14971:2012, ISO 13485:2016, IEC 62366:2015 and IEC 62304:2006 in relation to the concrete ISO/IEC 25010:2011 attributes and views on product quality in the 4+1 Views framework?

RQ2 The extent to which the current reference software architecture for the development of medical device software provides the support for the quality and perspectives specified in RQ1?

Standards for healthcare enable architects to create robust systems that perform to the quality expected from the ISO/IEC 25010:2011 model for building quality. ISO 13485:2016 offers a standardized procedure in the creation of medical devices and correlates well with the development views in the 4+1 views. In addition, there are excellent links between the perspectives in ISO 13485:2016 on quality management and those in the 4+1 view. Consequently, the migration of medical device standards with both software architecture documents provides developers with a blueprint on how to implement standards-driven medical device software design.

To answering RQ1, the correlation of the requirements in the medical device standard to ISO 25010 sub-characteristics and 4+1 views were carried out (Section. 2.2). To answer RQ2, a systematic literature review was conducted in Section. 2.3, to find medical software architectures: service-oriented medical device architecture (Kasparick et al., 2018), fuzzy-based modular (Aguwa, Monplaisir, Sylajakumari & Muni, 2010), intrinsically secure, open, and safe cyber-physically enabled, life-critical essential services architectures (Harp, Carpenter & Hatcliff, 2018), the sensor messaging system for serving retirees as well as supporting living (Liu et al., 2005), and model-based systems engineering (Corns, Thukral & Thukral, 2014). Next, a mapping to express how well these medical device architectures relate to the software architecture document was created in Section. 2.2.

2.2 Finding

2.2.1 The Mapping of Standards to Software Architecture Views for Medical Devices

In order to map sub-characteristics and characteristics in ISO/IEC 25010:2011 to the requirements in the medical standards (ISO 13485, ISO 14971, IEC 62304 and IEC

62366) in a 4+1 view, four tables have been generated.

In order to map characteristics and sub-characteristics and views in ISO/IEC 25010:2011 to the requirements in the medical standards (ISO 13485, ISO 14971, IEC 62304 and IEC 62366) in a 4+1 view, four tables have been generated (Table. 2.1, Table. 2.2, Table. 2.3, Table. 2.4). The `R[number]` column in an individual table presents the number of requirements determined from every particular ISO/IEC 25010:2011 (sub)characteristic, and the `View` column shows the specific architecture view in the 4+1 view associated with the recognized (sub)-characteristic.

ISO 13485:2016

In ISO 13485:2016 has 134 non-functional or quality requirements. A majority of them can be correlated to the functional applicability in ISO/IEC 25010:2011 and the logical view in the 4+1 view. See table for comprehensive mapping in `https://drive.google.com/file/d/1ieQhaPDyxI_X5nu89kCLk7qV5WIgue-9/view?usp=sharing`

Sub-features in ISO 25010	**R[num]**	**View**
Security-Integrity	1	View of Process
Maintainability-Modifiability	3	View of Development
Functional suitability-Functional appropriateness	124	View of Logical
Portability-Installability	3	View of Physical
Usability-Appropriateness recognizability	2	View of Logical

Table 2.1: The ISO 13485 requires a mapping to ISO 25010 sub-features 4+1 Views Model

A number of examples were demonstrated from this mapping. The standard requirement $R60$ is documented as follows, *"When product requirements are changed, the organization shall ensure that relevant documents are amended and that relevant personnel are made aware of the changed requirements."* (ISO 13485, 2016, p. 13). It is a requirement for maintainability and has therefore been mapped to ISO/IEC 25010:2011 for the maintainability-modifiability sub-characteristic. Furthermore, as maintainability refers to the development aspect, a development view has been mapped to this requirement. The standard requirement $R77$ is documented as follows, *"Validation shall be completed prior to release for use of the product to the customer."* (ISO 13485, 2016, p. 16). It is a usability requirement and is therefore mapped to the ISO/IEC 25010:2011 usability-appropriateness-identifiability sub-characteristic. Moreover, we mapped this requirement to the logical view as usability is a concern in the logical view.

ISO 14971:2012

In ISO 14971:2012 44 non-functional or quality requirements are included, the majority of them could be related to the functional suitability in ISO/IEC 25010:2011 and the logical view in the 4+1 view. In addition, it relates to maintainability and reliability as well as the process view. Table. 2.2 displays the mapping of requirements from ISO 14971:2012. The detailed mapping is presented in `https://drive.google.com/file/d/1Qd7M-N8De470XU1ZG967L575UWvZm6DY/view?usp=sharing`

Two examples of this mapping are presented. The requirement $R26$ is described for the standard as follows, *"The risk control measures selected shall be recorded in the risk management file."* (ISO 14971, 2012, p. 11). A maintainability requirement and therefore mapped to the maintainability-modifiability sub-feature of ISO/IEC 25010:2011. Also, as maintainability is a development issue, this requirement has been mapped to the development perspective. The standard requirement $R17$ is documented as follows, *"The manufacturer shall compile documentation on known and foreseeable*

Sub-features in ISO 25010	**R[num]**	**View**
Maintainability-Modifiability	4	View of Logical
Functional suitability-Functional appropriateness	39	View of Logical
Reliability-Fault tolerance	1	View of Process

Table 2.2: The ISO 14971 requires a mapping to ISO 25010 sub-features 4+1 Views Model

hazards associated with the medical device in both normal and fault conditions." (ISO 14971, 2012, p. 9). It is a reliability requirement and is therefore mapped to the ISO/IEC 25010:2011 reliability - fault tolerance sub-characteristic. It has been mapped to the process view in relation to runtime.

IEC 62304:2006

There are 127 quality requirements in IEC 62304:2006. In particular it relates to ISO/IEC 25010:2011 on functional suitability, co-existence and modifiability as well as to the 4+1 view on logic. Table. 2.3 establishes requirements for the mapping of IEC 62304:2006. Details of the mapping can be found at `https://drive.google.com/file/d/1sX9Ccfnv19ffIIvvwOflqtznvoCv-dFa/view?usp=sharing`

Two examples are listed from this mapping. The standard has the requirement $R131$ described as follows, *"The MANUFACTURER shall implement the change as specified in the CHANGE REQUEST. The MANUFACTURER shall identify and perform any ACTIVITY that needs to be repeated as a result of the change, including changes to the software safety classification of SOFTWARE SYSTEMS and SOFTWARE ITEMS. [Class A, B, C]"* (IEC 62304, 2006, p. 37). It is a compatibility requirement and is therefore mapped to the ISO/IEC 25010:2011 compatibility - coexistence sub-feature. A developmental view is mapped to this requirement. Standard requirement $R110$ is

Sub-features in ISO 25010	`R[num]`	`View`
Compatibility-Co-existence	27	View of Development
Security-Integrity	1	View of Process
Maintainability-Modifiability	12	View of Development
Functional suitability-Functional appropriateness	67	View of Logical
Usability-User interface aesthetics	1	View of Logical
Maintainability-Testability	9	View of Development
Security-Accountability	1	View of Process
Maintainability-Modularity	1	View of Development
Security-Authenticity	6	View of Process
Reliability-Availability	1	View of Process

Table 2.3: The IEC 62304 requires a mapping to ISO 25010 sub-features 4+1 Views Model

described as follows, *"The MANUFACTURER shall EVALUATE and approve CHANGE REQUESTS which modify released MEDICAL DEVICE SOFTWARE. [Class A, B, C]"* (IEC 62304, 2006, p. 33). It is a maintainability requirement and is therefore mapped to ISO/IEC 25010:2011 in the maintainability-modifiability sub-characteristic and development view.

IEC 62366:2015

54 quality requirements in IEC 62366:2015, most of which involve the aesthetic and logical view of the user interface in usability in ISO/IEC 25010:2011. Table. 2.4 supplies the requirements mapping for IEC 62366:2015. The mapping in detail is shown in

https://drive.google.com/file/d/1Mn8kkYU8vU-gwsAV0BWIy4HZa3vtGXL3/view?usp=sharing

Sub-features in ISO 25010	**R[num]**	**View**
Usability-User interface aesthetics	41	View of Logical
Functional suitability-Functional appropriateness	7	View of Logical
Maintainability-Testability	2	View of Development
Reliability-Maturity	4	View of Development

Table 2.4: The IEC 62366 requires a mapping to ISO 25010 sub-features 4+1 Views Model

Examples of such mappings are provided. Standard requirement $R3$ is described as follows, *"Where a documented product realization PROCESS exists, such as that described in Clause 7 of ISO 13485:2003 [11], it shall incorporate the appropriate parts of or reference the USABILITY ENGINEERING PROCESS."* (IEC 62366, 2015, p. 13). It is a usability requirement and is therefore mapped into ISO/IEC 25010:2011 Usability - User interface aesthetics sub-characteristics. In addition, as usability represents a consideration within the logical view, as such this requirement has been mapped to the logical view.

Requirement $R19$ in the standard has the following description, *"The MANUFACTURER shall identify and describe the reasonably foreseeable HAZARD-RELATED USE SCENARIOS associated with the identified HAZARDS and HAZARDOUS SITUATIONS. The description of each identified HAZARD-RELATED USE SCENARIO shall include all TASKS and their sequences as well as the SEVERITY of the associated HARM."* (IEC 62366, 2015, p. 13). It is a reliability requirement and is therefore mapped to the ISO/IEC 25010:2011 reliability - maturity sub-feature. In addition, this

requirement has been mapped to the process perspective as reliability is a concern in the process perspective.

2.2.2 Support from Current Architectures

This systematic literature review recognized five medical device software related architectures including: service-oriented medical device architecture (SOMDA) (Kasparick et al., 2018), Fuzzy-Based modular (Aguwa et al., 2010), intrinsically secure, open, and safe cyber-physically enabled, life-critical essential services (ISOSCELES) architectures (Harp et al., 2018), sensor information systems for active retires and assisted living (SISARL) (Liu et al., 2005) and model-based systems engineering (MBSE) (Corns et al., 2014).

In order to find out how each architecture supports the software architecture document (ISO 25100 and 4+1 Views), The following steps are applied to inspect them:

1. Classification of the sub-characteristics in Table. 2.1–Table. 2.4 according to the 4+1 view types. For instance, the five sub-characteristics Maintainability - Modifiability, Maintainability - Testability, Maintainability - Modularity, Compatibility - Coexistence and Reliability - Maturity are categorised as Development View-related qualities. Likewise, nine sub-characteristics are covered by the process view.

2. The support rate for each view of each reference architecture is calculated as the percentage of the sub-characteristics addressed explicitly in that architecture out of the total number of sub-characteristics established within step 1 for that view. For example, SOMDA's view of the process provides direct backing for Security Confidentiality and Security Integrity, which are two of the nine sub-features that make up the process view.

Table. 2.5 exhibits the outcomes of our mapping. Probably because medical devices are standalone and not connected to each other, none of these architectures support physical views. They have varying degrees of support for other views, but all are less than 50 per cent. So, in order to make them more standards-compliant, we need to further tune these architectures.

2.3 The Process of Systematic Literature Review

This chapter uses a systematic literature review (SLR) as the preliminary approach to address the need to identify and analyse existing preliminary research in response to two research questions (Kitchenham, 2004). Figure. 2.1 demonstrates the SLR process. Figure. 2.1 illustrates the process of SLR.

This Software Literature Review (SLR) was searched for terms such as: medical device software architecture, medical IoT architecture, medical IoT architecture, IEC 62304, ISO 14971, ISO 13485 and IEC 62366. A combination of a few of these terms has been applied to the search of various peer reviewed research databases Springer, Scopus, Science Direct and IEEEXplore, and Google Scholar for completeness. The query string has been reduced to its final form as follows: (medical device software architecture) OR (medical IoT architecture) OR (medical internet of things architecture) OR (IEC 62304 OR ISO 14971 OR ISO 13485 OR IEC 62366)

A preliminary search identified a total of 137 studies. For the purpose of informing of the research which is most pertinent for the development of software as well as its associated areas, the study have applied inclusion and exclusion criteria (Malhotra & Chug, 2016). The study have included every article in this chapter following a complete manual review of titles and summaries only, which reduced the total number within the critical studies to 50. We then read through them in detail and removed 16 papers of little relevance. Duplicate articles were excluded using quality assessment criteria,

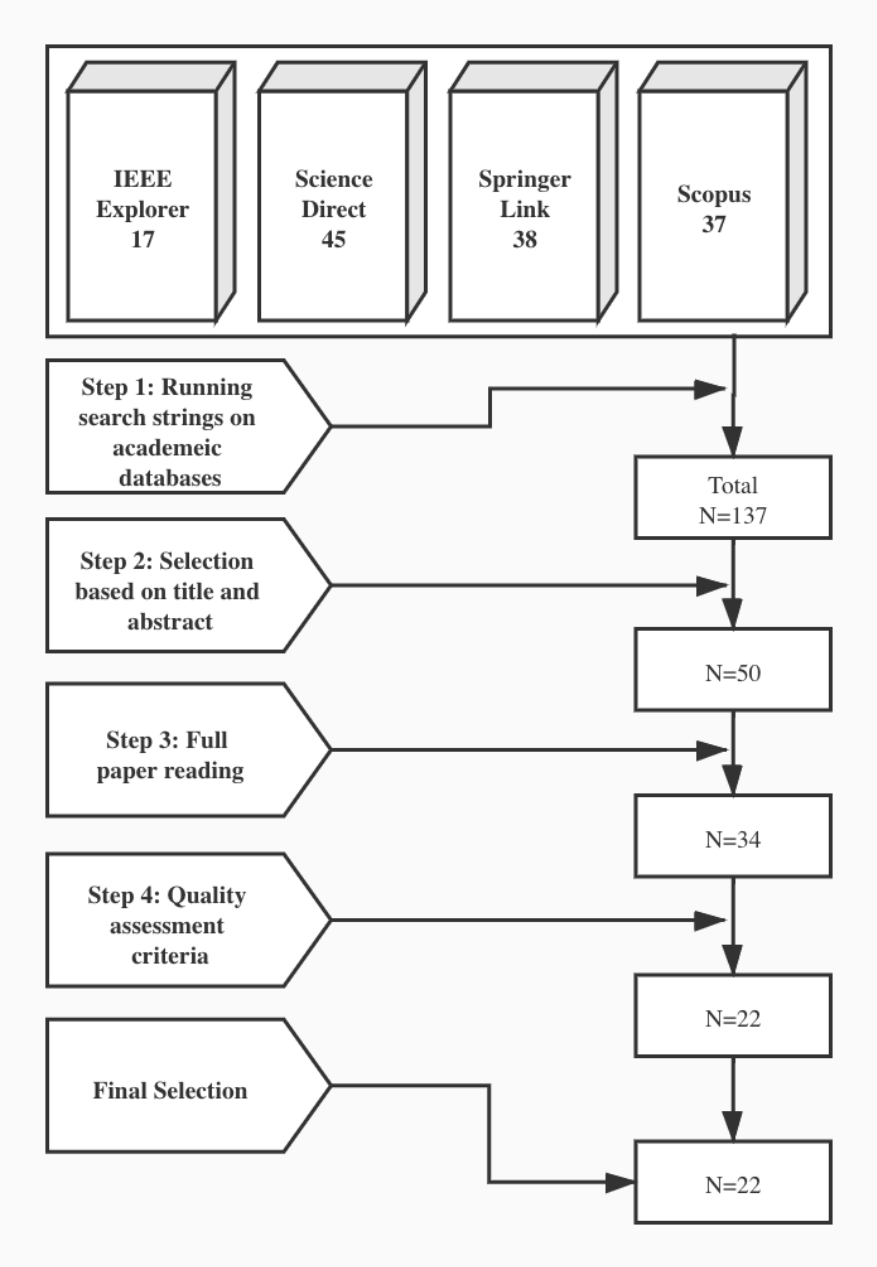

Figure 2.1: Systematic literature review process

resulting in the exclusion of 22 papers.

For data extraction and synthesis, the study have created tables to show the mapping between the requirements and software architecture documents for medical device standards. These tables will be used to answer the research questions and will be presented in subsequent sections.

2.4 Conclusion

There are four detailed mappings created to illustrate the relationship between the requirements of the medical device standard and the software architecture documentation. ISO 13485:2016 basically describes support for ISO/IEC 25010:2011 for functional applicability and the logical view in the 4+1 view; and also for ISO/IEC 25010:2011 for other characteristics such as security, maintainability, portability and usability. Other characteristics of ISO/IEC 25010:2011 such as security, maintainability, portability, and usability, as well as the process, development and physical views of the 4+1 view are also supported (Table. 2.1). While ISO 14971:2012 supports primarily an ISO/IEC 25010:2011 functional suitability characteristic and the logical view of the 4+1 view, it also addresses the ISO/IEC 25010:2011 maintainability and reliability characteristics and the process view of the 4+1 view (Table. 2.2). This IEC 62304:2006 tackles ISO/IEC 25010:2011 in terms of functional suitability, compatibility and maintainability, and the logical view and development view of the 4+1 view (Table. 2.3). The IEC 62366 focuses on usability and functional appropriateness, which is connected to the logical view in the 4+1 view (Table. 2.4). As shown in Table. 2.5, These additional mappings highlight that the current medical software reference architectures only partially support those ISO/IEC 25010:2011 characteristics which were identified during the previous mappings. It is suggested a refinement of these architectures could be made to provide more direct support for the required ISO/IEC 25010:2011 features, thereby reducing the software compliance burden to medical standards.

Limitations of this SLR include the potential for human error and subjectivity when linking the requirements of the standard to the relevant characteristics in ISO/IEC 25010:2011 and the views in the 4+1 view respectively. For the future, this issue can be resolved by a separate assessment of the diagram by subject matter experts. Furthermore, there were only seven studies established as answering RQ2. While the SLR process

followed is robust, some studies suggest that the results may not be scalable to other reference architectures.

The future orientation of this research contains the identification of possible improvements to the reference architecture, incorporating further views or an architectural strategy to provide better support for healthcare standards. A mapping could also be performed similar with the ISO 25010 process quality model.

Reference Architecture	View of Logical	View of Physical	View of Process	View of Development
SOMDA	0%	0%	22% Unsupported Reliability-Maturity, Fault Tolerance, Recoverability, Availability Security-Authenticity, Non-repudiation, Accountability	20% Unsupported: Performance Efficiency- Resource Utilization, Time Behaviour, Capacity Maintainability-Modularity, Analysability, Modifiability, Reusability, Testability
Fuzzy-Based Modular	0%	0%	22% Unsupported: Reliability-Maturity, Recoverability, Fault Tolerance Security-Authenticity,Confidentiality, Accountability, Non-repudiation	10% Unsupported: Performance Efficiency-Capacity, Resource Utilization, Time Behaviour Compatibility - Co-existence, Interoperability Maintainability-Reusability, Analysability, Modifiability, Testability
ISOSCELES	10% Unsupported: Functional Suitability - Functional Completeness, Functional Correctness, Functional Appropriateness Usability- Appropriateness Recognizability, User Error Protection, Accessibility, Learnability, Operability	0%	22% Unsupported: Reliability-Fault Tolerance, Recoverability, Availability, Maturity Security-Integrity, Non-repudiation, Confidentiality	30% Unsupported: Performance Efficiency - Resource Utilization, Time Behaviour, Capacity Maintainability-Testability Analysability, Modifiability, Reusability
SISARL	20% Unsupported: Functional Suitability - Functional Completeness, Functional Correctness, Functional Appropriateness Usability - Operability, Appropriateness Recognizability, User Error Protection, Learnability	0%	0%	40% Unsupported: Performance Efficiency - Resource Utilization, Time Behaviour, Capacity Maintainability-Testability Analysability, Reusability
MBSE	10% Unsupported: Functional Suitability - Functional Completeness, Functional Correctness, Functional Appropriateness Usability - Operability, Appropriateness Recognizability, User Interface Aesthetics, Learnability, Accessibility	0%	11% Unsupported: Reliability-Availability, Fault Tolerance, Recoverability	40% Unsupported: Performance Efficiency - Time Behaviour, Resource Utilization, Capacity Maintainability-Testability, Analysability, Reusability

Table 2.5: The ISO 25010 sub-characteristics are supported on a per reference architecture view

Chapter 3

Research Methodology

This chapter discusses the methods used to answer each of the following four research questions:

RQ1 How relevant are the requirements from ISO 14971:2012, ISO 13485:2016, IEC 62366:2015 and IEC 62304:2006 in relation to the concrete ISO/IEC 25010:2011 attributes and views on product quality in the 4+1 Views framework?

RQ2 To what extent to which the current reference software architecture for the development of medical device software provides the support for the quality and perspectives specified in RQ1?

RQ3 How can a novel QMS be architected to support the requirement identified in RQ1?

RQ4 How efficient is the QMS for managing requirements from medical standards?

In Chapter 2, a systematic literature review was undertaken to answer RQ1 and RQ2. The remainder of this chapter aims to explore ways of answering RQ3 and RQ4. 566

Research methodology is generally considered to show the researcher's way of thinking about a particular phenomenon and the method of research (Alturki, Gable & Bandara, 2013). Also, Research methodology links the selection and usage of methods and is a pavement for a plan, strategy, procedure, or design (Alturki et al., 2013).

Design science research is a method by which research is built and manipulated when the expected outcome is an artifact or proposal. Besides, design science-based research can be conducted in both academic and organizational settings (Dresch, Lacerda & Antunes, 2015). Bachmann and Bass (2001) introduced that Attribute-Driven Design (ADD) is a procedure for designing the software architecture of a system or group of systems based on an explicitly stated goal of quality attributes of the system. The procedure is specifically addressed to any quality attribute like performance, modifiability, security, reliability, or availability (Bachmann & Bass, 2001).

3.1 Appropriate Selection of Research Methods

The selection of research methods for evidence-based software engineering research is questionable, as the advantages and challenges of applying each method have not been well categorized (Easterbrook, Singer, Storey & Damian, 2008). As we intend to address the diverse research objectives of RQ3 and RQ4 in this case, we consider a hybrid methods strategy that integrates qualitative and quantitative methods. However, a number of particular methods are available for a given study. Therefore, many options were discussed prior to selecting a specific research method (Easterbrook et al., 2008):

Action Research generates deeply relational research findings that are based on actual practice and designed to address the problem situation at hand, while critically providing information for theory (Baskerville, 1999).

Survey Research involves the collection of data from a large number of sources through questionnaires, interviews, published and unpublished statistics, using a set

of methods based on qualitative and quantitative analysis, and the use of statistical techniques to analyse the data (Gable, 1994).

Controlled Experiments and Quasi-Experiments have been widely practiced in scientific research. What they enable us to do is to survey the relationship between the various variables and whether there is a causal relationship between them on the basis of necessarily testable hypotheses (Easterbrook et al., 2008).

Ethnographies are scientifically established methods for identifying and surveying both social and cultural models as well as significance among diverse communities, organisations, and other social contexts (Schensul, Schensul & LeCompte, 1999).

Case Studies can be especially valuable in the case of particular issues and situations that require in-depth study and explanation (Noor, 2008).

Design science concretely operates in academic or organizational settings in which such research can be conducted when an artifact or a proposal is considered a target of the intended research (Dresch et al., 2015).

Attribute-driven design approach enables the definition of software architecture by defining the software architecture based on the quality attributes that must be satisfied by the software during the design process and, therefore, it completes the defined functional candidate architecture (Nord, Wood & Clements, n.d.).

3.2 Our Approach

Considering the characteristics of the listed methodologies above, in this case, Design Science Sec. 3.3, Attribute-driven design Sec. 3.4 and Case Study Sec. 3.5 were mixed to achieve the research goals of RQ3 and RQ4. The rationale of the selected methods to answer the corresponding research questions is explained in Table. 3.1

Research Question	Methodology	Rationale
RQ3	Design Science& Attribute-Driven Design	1. Design Science presents the process and protocols based on seven key aspects. 2. Following the steps in Attribute-Driven to design the architecture of QMS
RQ4	Case Study	Using QMS to do risk management

Table 3.1: Selection of Research Methods

3.3 Design Science

A framework for undertaking design science research was suggested in 2004, suggesting processes and protocols to focus on seven key features (Hevner, March, Park, Ram et al., 2004). Individual items in these guidelines were used to support the creation of the Quality Management System (QMS) architecture.

Guideline 1: an Artifact Design

The consequence of scientific research in information system design is a targeted information technology product created to solve an essential organizational issue. It must be efficiently depicted so that it can be implemented and applied in the appropriate field (Hevner et al., 2004). This study develops a QMS and demonstrates its architecture design and prototype. The QMS architecture, including the primary requirements, business concerns, technical concerns and quality attributes, and prototypes based on the QMS architecture, are introduced in Chapter 4 and Chapter 5.

Guideline 2: Relevance of the problem

The goal of information systems research is to gain the knowledge and comprehension to allow the construction and implementation of technology-based answers to important commercial problems hitherto unaddressed (Hevner et al., 2004). The standard ISO 14971 on risk management has been discussed in SLR in chapter 2. It requires manufacturers of medical devices to offer forms for hazard recognition and recommended methods for making judgments about safety, including risk acceptability (Lincoln, 2009).

Guideline 3: Evaluation of Design

The qualified design of artifact must be strictly illustrated through well-established assessment procedures which is a key element of the research process (Hevner et al., 2004). The architecture of the QMS in Chapter 4 deals with the three sub-characteristics of ISO 25010. ATAM has been a notorious scenario-based approach system architecture evaluation method. When performing the evaluation, it is essential to consider a group of scenarios that concentrates on the non-functional aspects of the system or on the qualities demonstrated in the system. ATAM was followed to evaluate the QMS architecture in Chapter 6.

Guideline 4: Contributions of Research

An explicit contribution in terms of design artifacts, design construction knowledge and/or design evaluation knowledge is a necessary component of efficient design-science research (Hevner et al., 2004). Worthwhile research contributions can range from software products that inspire research by other participants to novel solutions that are widely accepted as extending the field (Weber, of Australia & Zealand, 1997). The contribution of this study is the instantiation of the QMS.

Guideline 5: Research rigor

Strictness refers to the way in which research is performed. Design science research demands the use of a disciplined approach in terms of both the creation and assessment of design work (Hevner et al., 2004). The next section of this CHAPTER will show the creation and evaluation of a detailed design.

Guideline 6: Design as a search process

Seeking the best or optimal design is usually challenging to solve realistic information systems problems (Hevner et al., 2004). For a design science that is essentially iterative, the property of the design process is characterized as a generate/test cycle (Simon, 1996). A QMS prototype is created based on the requirements model from the first iteration.

Guideline 7: Communication of research

Design science research has to be presented to both technically and managerially driven viewers (Hevner et al., 2004). During the QMS design process, an architecture driver document is created. The primary functional requirements, technical constraints, and business constraints in the document describe artifact for management-oriented audiences, and the primary functional requirements and quality attributes describe artifact for technology-oriented audiences.

3.4 Attribute-driven design

We mentioned in the previous section that we want to design a QMS architecture. We have chosen the ADD method to perform one or more iterations to decompose each quality attribute and make architectural decisions respectively.

The ADD method performing steps was mentioned (Bass, Klein & Bachmann, 2001) as following:

1 Select design features for decomposition. The description element to start with is usually the whole system. For this design element all required inputs should be available constraints, functional requirements and quality requirements.

2 Refine the feature based on the below steps:

   1). Select architectural drivers from a collection of quality scenarios and functional requirements. This step determines the important features of this decomposition.

   2). Select the attribute primitives and child design factor types to match the architectural drivers. This step is designed to comply with quality requirements.

   3). Instantiate design features and use multiple views to assign functionality from the use case. This step is designed to meet functional requirements.

   4). Verification and refinement of the use cases and quality scenarios, and binding them to sub-design features. In this step, it is confirmed that nothing critical is missing, and sub-design features are prepared for further decomposition or implementation.

3 Repeat these steps for each design feature that requires further decomposition.

Chapter 4 will follow ADD method to create an architecture for QMS.

3.5 Case Study

In the case study, the development of QMS and using QMS to do risk management for a data logger will be presented. Also, the risk management process is selected in the

Table 3.2: Research Plan

Activities	Month											
	1	2	3	4	5	6	7	8	9	10	11	12
Systematic Literature Review	X	X	X									
Research Methodology				X	X							
Background									X			
Outcome 1: Architecture					X			X	X			
Outcome 2: Case Study						X	X	X	X	X		
Discussion										X		
Conclusion											X	
Introduction												X

risk management standard ISO 14971.

3.6 Research Plan

The research plan was developed during the research. The plan shows the timeline for the research, which includes the time points of the activities mentioned in the research methodology. Table. 3.2 shows the research plan of this study.

3.7 Conclusion

This chapter introduces design science, attribute-driven design and Case study. Design science suggests a framework of processes and protocols, and this framework includes seven key features to enable us to complete a proper design and the development of a design-based system. In Chapter 4, We designed the architecture of the QMS using the ADD method, which required us to decompose the quality attributes and make some architectural decisions over one or more iterations. In Chapter 5, we show the development of a case study for a QMS based on the architecture of Chapter 4. This case study uses the QMS we have developed for a data logger to do the risk management

mentioned in ISO 14971.

Chapter 4

Architecture Design of QMS

This chapter details the process of architecting the QMS. Software architecture design involves the procedure of decomposing requirements to a structure or structure of a system, which consists of elements of software, their outwardly transparent attributes and the relationships between them (Bass, Clements & Kazman, 2003).

For the rest of this chapter, the architecture design is ADD-based. Section. 4.1 introduces architecturally significant requirement. Section. 4.2 presents the architecture design using ADD.

4.1 Characterizing Architecturally Significant Requirements

Chen, Babar and Nuseibeh (2012) introduced a new framework describing architecturally important requirements based on an empirical study with practitioners from 90 organisations of different sizes and domains. The framework includes four sets of characteristics: definition, descriptions, indicators, and heuristics. Figure 4.1 presents the sub-characteristics.

The ASRs were defined as requirements that measurably affect the architecture

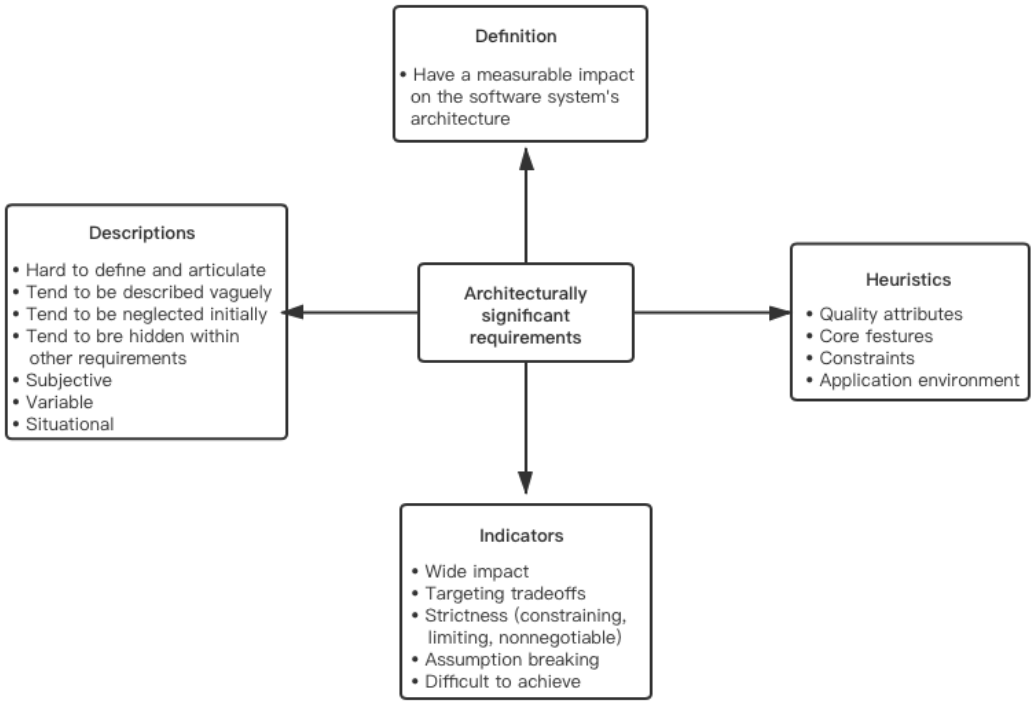

Figure 4.1: An architecturally significant requirements characterisation framework (Chen et al., 2012)

of a software system. Basically, the definition restricts a part of the requirements to those that affect the system architecture in a measurable way (Chen et al., 2012). Quality attributes, core features and Constraints as the architecturally significant requirements were selected for this research after meeting with stakeholders including project manager, developer and end user. During the meeting with stakeholders, the project manager provided the core attributes and constraints in accordance with the ISO 14971 risk management process, and also selected the highest priority quality attributes through a vote.

Quality attributes

It is typically architecturally important when a requirement specifies the quality attributes of a software system.

Core features

Reference to the requirements of the core functionality of a software system is probably of architectural importance. Core functions define the problems that the software is intended to solve, they usually catch the nature of the behaviour of the software system and characterise the key expectations that users have from it. These serve explicitly to accomplish the objectives for building the system.

Constraints

It is generally architecturally important to impose constraints on the requirements of a software system.

Section. 4.1 described architecturally significant requirements (ASRs). Constraints, quality attributes and core features which can also be functional requirements were selected.

4.1.1 Functional Requirements

Functional requirements in software engineering and systems engineering specify the functionality of a system or its elements, where functionality is stated as a specification of the behaviours between outputs and inputs (Fulton & Vandermolen, 2017). Project manager provides the following functional requirements:

1. Risk Management Procedure:

    1). Distinguish between Team Member, Manager, QA.

    2). Anyone can draft Severity categories, Severity descriptions, Probability categories, and Probability descriptions.

    3). QA must approve definitions before use.

4). Anyone can draft Risk Acceptability Matrix (6.3.3): risk = acceptable | unacceptable + other score (e.g. low | medium | high).

5). QA must approve table above.

6). Define types of risk control and whether severity and frequency can change. (e.g. “Labelling” -> cannot change either frequency or severity; “Inherent safety”-> can change both).

2. FMEA table:

1). Anyone can draft FMEA table as Design FMEA document.

2). Anyone can modify table and Only QA can approve it.

3). Auto calculate risk acceptability.

4). Risk control.

3 Versioning of procedure and FMEA table. (Must be consistent e.g. warning to users).

4 Generate risk management report (Word or PDF).

4.1.2 Design Constraints

Three design constraints are necessary to achieve the above functional requirements, including user permission, approval of risk procedures and FMEA forms, and user input protection. Project managers expected that this QMS could be quite scalable, therefore the fourth Constraints is Scalability.

User Permission

The project manager requires the users of the system to be classified as Team Member, Manager, QA and System Administrator.

Approving of Risk Procedure and FMEA Table

Only the QA user can perform the approving action to make the Risk Procedure or FMEA table to final effect.

User Inputs Protection

As users add and modify risk procedures and FMEA tables they may enter any possible data. In order to ensure the correctness of the final data during the risk management process and the stability of the system, it is necessary to filter the data entered by the user or to alert the system when the user enters invalid data.

Scalability

As this thesis will only develop a QMS prototype in compliance with the ISO 14971 standard, this prototype will merely include the most basic components for risk control. We will prepare it for further development so that subsequent developers can add new features or modify existing modules with minimal cost.

4.1.3 Quality Attributes Requirements of QMS

Quality attributes are functional features that are important to the system (Zhang & Xi, 2008). For the design of QMS, its basic qualities to achieve are learnability, user error protection and modifiability. Barbacci, Ellison, Lattanze, Stafford and Weinstock (2003) stated that quality attribute scenarios for describing interactions with the system are vignettes demonstrating particular quality attributes that are critical to the system.

Barbacci et al. (2003) also introduced a quality attribute workshop (QAW) for generating, prioritizing, and refining quality attributes scenarios until the software architecture has been established.

Listed below are the eight steps involved in QAW:

1. **Introduction and Presentation of QAW.** For this step, the QAW promoter introduces the motive for the QAW and gives an explanation of the individual processes of the procedure. A standard slide presentation can be used, which can be customised to suit the requirements of the facilitator.

2. **Presentation of Business or Mission.** Throughout the presentation, the stakeholders listen attentively and catch relevant information which might illuminate the drivers of the quality attributes. The quality attributes that will be developed in the following stages will be principally taken from the business/mission requirements presented in this step.

3. **Presentation of Architecture Plan.** Although there may not be a fine-grained system architecture, a high-level system definition, background diagram or other product may have been generated that provides technical information on the system.

4. **Identify Architectural Drivers.** In steps **Presentation of Business or Mission** and **Presentation of Architecture Plan**, the promoter obtains information about the architectural drivers that are critical to achieving the objectives of the quality attributes in the system. In general, as a consequence, these drivers include high-level demands, business/mission concerns, goals and objectives, and diverse quality attributes.

5. **Brainstorming of Quality Attribute Scenario.** After identifying the architectural drivers, the moderator initiates a brainstorming procedure to allow stakeholders to propose scenarios. The moderator revises the various elements of a well-formulated scenario and makes sure that every scenario has been well constructed throughout the workshop.

Table 4.1: Quality Attribute: Learnability

QA Scenario	Learnability
Source of stimulus	End user
Stimulus	Wants to learn system features
Environment	At runtime or configure time
Affected resources	System
Response	Interface is familiar to user; interface is usable in an unfamiliar context
Response measure	User take less than 1 minutes to know how to access to the system; User take less than 2 minutes to read unfamiliar context description;

6. **Consolidation of Quality Attribute Scenario.** In this step, the moderator requests stakeholders to pick out scenarios where the content is highly comparable. The comparable scenarios will be merged once the people presenting the scenarios accept and consider them not to be watered down in the process.

7. **Prioritization of Quality Attribute Scenario.** Prioritization of scenarios was done by assigning for each stakeholder a percentage of votes equal to 30% of the total number of scenarios resulting from the merger. The actual number of votes allocated to stakeholders was rounded up to an even number at the discretion of the moderator.

8. **Refinement of Quality Attribute Scenario.** After prioritization, the top four or five scenarios are refined in more detail, depending on the amount of time remaining.

The eight steps mentioned above were followed to complete the QAW and collected three high priority quality attribute scenarios including Learnability, User Error Protection and Modifiability. Table. 4.1, Table. 4.2 and Table. 4.3 present these QMS quality attributes scenarios.

Table 4.2: Quality Attribute: User Error Protection

QA Scenario	User error protection
Source of stimulus	End user
Stimulus	Minimize impact of error
Environment	At runtime or configure time
Affected resources	System
Response	Recognize and correct user incorrect input or operation;
Response measure	Ensure that user actions do not cause system errors

Table 4.3: Quality Attribute: Modifiability

QA Scenario	Modifiability
Source of stimulus	End user, Developer
Stimulus	Wishes to add/delete/modify/vary functionality, quality attribute, capacity
Environment	At runtime or compile time, build time, design time
Affected resources	System user interface, platform, environment; system that interoperates with target system
Response	Locates places in architecture to be modified; makes modification without affecting other functionality; tests modification; deploys modification; choice of common languages and frame works;
Response measure	fixes bugs about user input should be less than 10 minutes; modifies a component of frontend should be less than 30 minutes; adds a component of frontend should be less than 1 hour; modifies a page of frontend should be less than 1 hour; adds a page of frontend should be less than 10 hours; modifies a REST API should be less than 1 hours; adds a REST API should be less than 2 hours; adda a new feature should be less than 2 days;

4.2 Architecture Design Using ADD

Wojcik et al. (2006) stated that a minimum of two iterations are required during the ADD process in order to develop an architecture that meets the requirements of the proposed system architecture. The resultant architecture is presented in Figure. 4.2 and described as below:

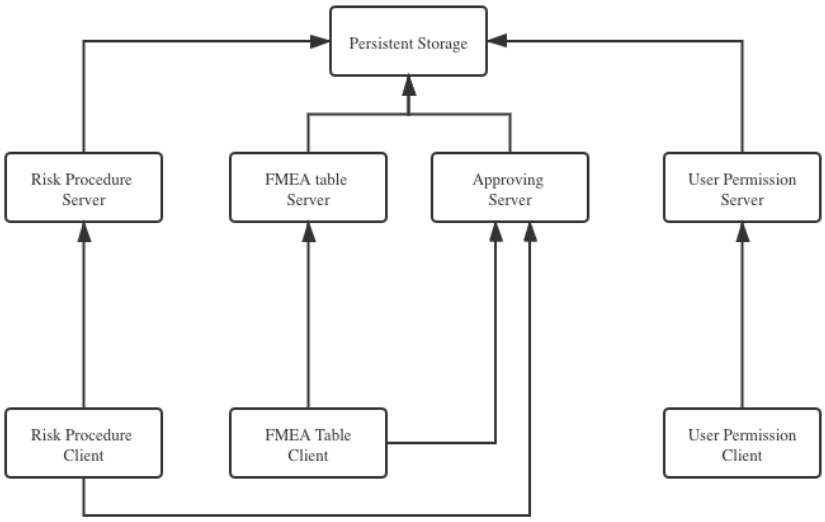

Figure 4.2: Software Element Primary Connectivity

1. Our design is based on a client-server model. The different module servers provide services for their respective clients. In the case of the FMEA table module, for example, the FMEA table server serves the FMEA table client.

2. The User Permission module is used to manage users in different roles.

3. Risk Procedure module and FMEA table module contain the approving function and therefore a separate server is used to handle the approving process. Only when the QA user login to the system the system will display the approving UI.

4. Servers store the data of the corresponding modules in persistent storage.

Table 4.4: Architectural Driver Priorities Determined in QAW

#	Architectural Drivers	Importance	Difficulty
1	Scenario 1 Learnability	medium	low
2	Scenario 2 User Error Protection	high	medium
3	Scenario 3 Modifiability	high	high

4.2.1 Summarizing Architecturally Significant Requirements

In Section. 4.1, ASRs including functional requirements, design constraints and quality attribute requirements were selected, which is the first iteration of ADD. For functional requirements, QMS contains three modules: risks procedure management module, FMEA table management module and user permission management module. As project manager expected, there are four design constraints: user permission, approving of risk procedure and FMEA table, user inputs protection and scalability. We also selected three quality attributes: learnability, user error protection and modifiability.

4.2.2 Identify Candidate Architectural Drivers

In Section. 4.1.3, We have collected three QAs according to the QAW methodology to meet with stakeholders to discuss the priorities of these three QAs, and the priorities we have discussed and decided on are shown in the Table. 4.4.

Consider the following points as you read Table. 4.4:

1. The drivers marked (high, high) are linked to the modification of the system and the addition of new features. Time requirements range from half a day to several weeks. This category is the most difficult to meet and is the highest priority for the drivers.

2. The drives marked (high, medium) are related to operations relevant to the user.

3. Scenario 1 is only the least important driver in terms of the experience of user operation. Thus, only six architectural drivers should be taken into account.

4.2.3 Select A Design Idea That Meets the Architectural Drivers

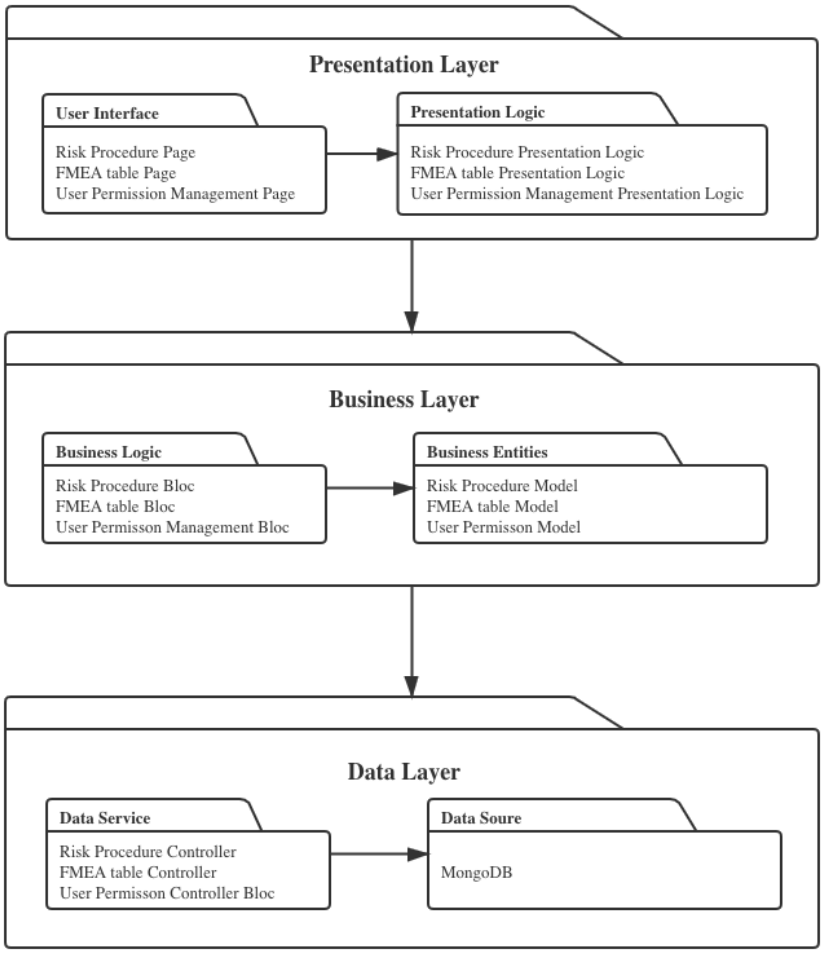

Figure 4.3: A Package Diagram of Three Layers Architecture

1. **Identify Design Concerns.** Bass et al. (2003) introduced three design concerns associated with modifiability:

   1). **Localize Modifications:** This concern refers to a set of tactics whose objective is to attribute responsibility to modules during the design process so

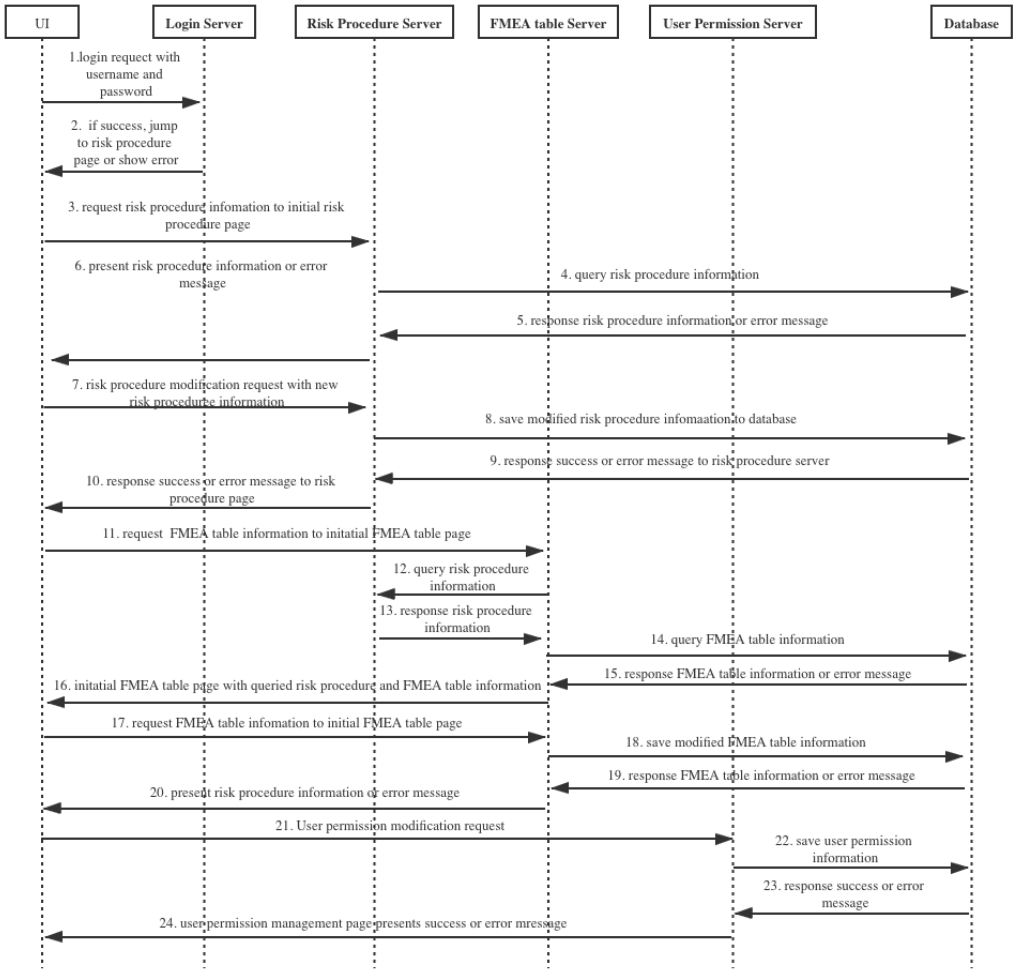

Figure 4.4: A Sequence Diagram Between Client-Server-Database

that the expected changes are limited in scope.

2). **Prevent Ripple Effects:** The concern relates to the tactics for dealing with the ripple effect.

3). **Defer Binding Time:** This concern relates to the tactics of intending to have impact at loadtime or runtime.

2. **List Alternative Patterns to address the design concern**.

1). **Localize Modifications.** Bass et al. (2003) presented four patterns to address localize modification as below:

a. **Maintain semantic coherence:** A contextual consistency relates to the relationship between the duties in a module.

b. **Abstract Common Services:** generally, are regarded as supportive of re-use.

c. **Anticipate expected changes:** A method for evaluating the allocation of specific responsibilities is provided, taking into account a range of scenario variations.

d. **Generalize the module:** To make a module more general, allow it to calculate a wider range of functions depending on the input.

e. **Limit possible options:** Limiting the possibilities would reduce the effectiveness of these modifications.

2). **Prevent Ripple Effects.** Bass et al. (2003) proposed four patterns to prevent ripple effects as below:

a. **Hide Information:** The hiding of information is the breaking down of the responsibility of an individual entity that chooses which information is private and which is public into bite-sized portions.

b. **Maintain Existing Interfaces:** Maintaining this interface and its syntax allows B to remain unchanged if interface B depends on the name and signature of one of the interfaces of A.

c. **Restrict Communication paths:** Limit the modules that share data with a particular module.

d. **Use an Intermediary:** An intermediary can be inserted between module B and module A to manage the activities associated with the dependency if module B has any type of dependency on module A other than semantics.

3). **Defer Binding Time.** Bass et al. (2003) presented five patterns to defer

binding time as below:

a. **Runtime Registration** allows plug-and-play functionality, but requires additional overhead to manage registration. For example, publish/subscribe registration can be done at runtime or at load time.

b. **Configuration Files** is used to set arguments at launch.

c. **Polymorphism** enables method invocations to be post-bound.

d. **Component Replacement** support to bind at load time.

e. **Adherence to Defined Protocols** permit independent process runtime binding.

3. **Select Patterns from the List.** This action consists of picking a pattern from the list for each group of alternatives. While making our choice, we need to reason out which choice is the most appropriate. The ADD method requires showing the interrelationships between patterns and their advantages and disadvantages for each architectural driver. It assumes that there will be a reasonable number of possible patterns and that tables are a good way of showing the alternatives. Each of the following sections has three parts. (1) a reasoning paragraph stating the pros and cons of each model, (2) a decision statement highlighting the chosen model, and (3) an implied statement indicating the impact of this decision, including any obvious limitations on the choices that have not yet been made.

1). **Localize Modifications**

a. **Reasoning:** Bass et al. (2003) mentioned that the more general a module, the more possibilities to modify by adjustment of the input language instead of modifying the module. However, not all modifications can be foreseen. The Maintain semantic coherence model is focused on

maintaining the relationship between the responsibilities within a module and the need to ensure data consistency between the Risk Procedure and the FMEA table in the QMS system. Abstract common services pattern are normally considered to support re-use, it is ideally suited to a project that requires long-term maintenance.

b. **Decision:** Use the abstract common services.

c. **Implications:** All modules and self-modules are reused as much as possible.

2). **Prevent Ripple Effects**

a. **Reasoning:** Bass et al. (2003) stated hide information pattern is closely associated with the expected anticipate changes pattern discussed earlier. Also, if Module A is semantically dependent on Module B, it is not always feasible to maintain existing interface pattern. Although restrict communication paths pattern reduces respectively the number of modules consuming the data generated by a given module and the number of modules consuming the data generated, use an intermediary pattern allows for various types of dependencies between two modules.

b. **Decision:** Use an intermediary pattern.

c. **Implications:** The Risk procedure client communicates with the FMEA table client via a middleware.

3). **Defer Binding Time**

a. **Reasoning:** While Runtime registration pattern, Polymorphism pattern and Adherence to defined protocols pattern support run time, Configuration files pattern and Component replacement pattern support load time. Configuration files pattern allows non-developers to participate in modifications to the system.

Table 4.5: Pattern/Driver Mapping of Modifiability

#	Pattern Types	Pattern Selected	Architectural Driver Response Measure
1	Localize Modifications	Abstract Common Service	Bugs fixing, New Feature Addition, Functional modifications
2	Prevent Ripple Effects	Use an Intermediary	Bugs fixing, New Feature Addition,
3	Defer Binding Time	Configuration Files	New Feature Addition,

b. **Decision:** Use the configuration files pattern.

c. **Implications:** The configuration file is only maintained for the individual modules in order to handle the frequently necessary modifications.

4. Determine Relationship Between Patterns and Drivers. A summary of the selected models is shown in Table. 4.5

5. **Capture Preliminary Architectural Views.** In this section, we present preliminary architectural views including process view and development view.

   1). **Process View.** Figure. 4.3 is a package diagram to present the three layers architecture including the presentation layer, business layer and data layer.

   2). **Development View.** Figure. 4.4 is a sequence diagram to show the front-end and back-end interaction relationships. the UI page sends HTTP requests to different servers, and each specific server then gets the data from the database and returns it to the UI page.

4.2.4 Instantiate Architectural Elements and Allocate Responsibilities

1. **UI pages.** This QMS has four UI pages, including the login page, the risk procedure management page, the FMEA table management page and the user

permission management page.

1). **Login Page.** The login page allows users to log in and start using the system after signing in with a username and password.

2). **Risk Procedure Management Page.** The Risk Procedure Management page enables users to add, modify or delete risk procedure management.

3). **FMEA Table Management Page.** The FMEA table management page allows users to select a group of risk procedure data that has been added and add, modify, or delete the FMEA table data corresponding to this group of risk procedure data.

4). **User Permission Management Page.** The User Permissions Management page gives system administrators the ability to add, modify and delete users and their permissions. Users who are not administrators cannot access this page.

2. **Servers.** There are four servers on the backend of the QMS, namely the login server, the risk procedure management server, the FMEA table management server and the user permission management server.

1). **Login Server.** The login server is responsible for processes authentication of user login.

2). **Risk Procedure Management Server.** The Risk Procedure Management Server is used to process requests from the Risk Procedure Management page to add, modify, and delete Risk Procedure Management data and return the latest Risk Procedure Management data.

3). **FMEA Table Management Server.** FMEA Table Management Server is used to process requests from the FMEA Table Management page.

4). **User Permission Management Server.** User Permission Management Server is used to modify user data.

3. **Databases.** The QMS backend uses MongoDB as the database, which stores risk program data, FMEA table data and user data.

4.2.5 Verification and Refinement of Demands and Making Them Constraints on Instantiated Elements

In this step, the decomposition is verified to satisfy the functional requirements, quality attribute requirements, and design constraints, and we demonstrate how these requirements and constraints also constrain the instantiated elements.

4.2.6 Next Iteration

Now that we have completed steps 1 through 7, we have generated a collection of responsibilities, each with design constraints, quality attribute requirements, and functional requirements. From there we could back up to the decomposition step in which to choose the next factor to decompose. In our case, we have no sub element to decompose further.

4.3 Conclusion

In this chapter, we identified the architectural significant requirements and then designed the QMS architecture based on the ASRs and following the ADD approach. For the identification of ASRs, three requirements were identified: core features, quality attributes and constraints. For the QMS architecture, we obtained a view of the architecture and architectural decisions through two iterations and decomposition of the ASRs. In

the next chapter, we will present a case study of a QMS based on the architecture developed in this chapter.

Chapter 5

Building QMS and Conducting Risk Management

Viriansky and Shaposhnikov (2019) stated that the Quality Management System (QMS) is a dedicated system which is developed and utilized in an organization for the goal of formulating the targets and principles of the organization's actions as well as meeting the requirements in the areas of product/service quality.

ISO 14971 (2019) provides a procedure for medical device manufacturers to recognize the hazards related to medical devices, the estimation and assessment of the associated risks, the control of these risks and the monitoring of the effectiveness of the controls.

This chapter demonstrates a QMS was developed that follows the ISO 14971 risk management process, which is illustrated in Figure. 5.1. Section. 5.1 introduces risk management process. In Section. 5.2 illustrates how we developed quality management system. Section. 5.3 presents we used QMS to do a risk management for a data logger. Section. 5.4 shows the extent of the QMS to support quality attributes.

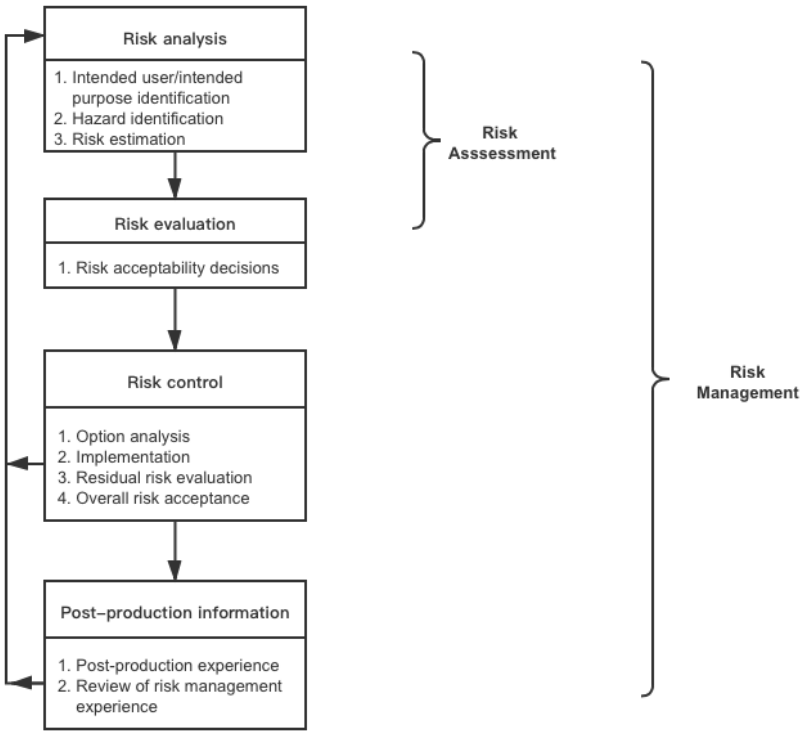

Figure 5.1: Risk management process ISO 14971 (2019)

5.1 Risk Management Process

ISO 14971 (2019) requires manufacturers to create, perform, record and maintain a continuous process that incorporates risk analysis, risk evaluation, risk control and production and post-production activities. In the QMS, the Risk Procedure Management component is responsible for risk analysis and the FMEA table Management component is used for risk evaluation and risk control.

5.1.1 Risk Analysis

ISO 14971 (2019) introduced that the specific medical device should be conducted risk analysis. Risk analysis process includes intended purpose identification, hazard identification and risk estimation.

Intended Purpose and Identification

The manufacturer should specify the intended use/intended purpose and any rationally predictable abuse for the particular medical device or accessory under consideration. The manufacturer is expected to identify all qualitative and quantitative features that potentially influence the safety of the medical device and, where possible, list their definitional limits. One approach is to develop a set of issues linked to the manufacturing, usage and eventual disposability of medical devices. The list of issues in a set could be helpful to assist the user in recognising all the hazards potentially associated with medical devices (Hegde, 2011).

Hazard Identification

The manufacturer should prepare a list of known or foreseeable hazards connected with the medical device, including hazards under normal and malfunctioning conditions. Hazards from prior identifications should be specified (Hegde, 2011).

Risk Estimation

For every recognized hazard, available information or data should be used to evaluate the risk under normal and failure conditions. The system for classifying probability estimates or severity, either qualitatively or quantitatively, should be documented in a risk management file (Hegde, 2011).

5.1.2 Risk Evaluation

For every recognized hazard, the manufacturer is expected to determine the requirement for risk reduction using the criteria specified in the risk management plan. The results of this risk evaluation should be documented in the risk management file (Hegde, 2011).

5.1.3 Risk Control

When risk reduction is required, manufacturers should comply with a procedure that includes options analysis, implementation of risk control measures, residual risk evaluation, risk/benefit analysis, other generated hazards, and completeness of risk evaluation to control risk so that the residual risk relevant to each hazard is judged to be acceptable (Hegde, 2011).

Option analysis

Hegde (2011) stated that the manufacturer Should determine the risk control measures applicable to reduce the risk to an acceptable level. Risk controls should include a comprehensive methodology, and the manufacturer should use one or more in the below priority order listed:

a) Design and manufacture of inherent safety.

b) Medical devices themselves or protection measures in the manufacturing process.

c) Security information and, where appropriate, user training.

Implementation of risk control measure(s)

Manufacturers should apply the risk control measures selected in the options analysis. The measures taken to manage risks should have been documented as part of the management of risk file. Verification of the efficiency of control measures for risk should be carried out and the results should be documented for the management of risk file. Verification of the implementation of risk control measures should be carried out. This verification should also be documented for the management of risk file (Hegde, 2011).

Remaining risk assessment

Remaining risks which continue after implementation of controls over risk should be assessed using the criteria set out in the risk management plan. The results of this assessment should be documented in the risk management file. If the residual risk does not conform to these standards, further risk control measures should be implemented. If the residual risk is deemed acceptable, all relevant information required to explain the residual risk should be included in the relevant supporting documentation provided by the manufacturer (Hegde, 2011).

Risk/benefit analysis

When residual risk is determined to not be acceptable utilising the criterion set out in a risk managed programme where additional controls for risk are impractical, data and literature on the medical benefits of the intended use/intended purpose should be collected and reviewed by the manufacturer to establish whether these benefits exceed the residual risk. To the extent that this evidence does not support the conclusion that the medical benefits outweigh the residual risks, the risks are still not acceptable. if the medical benefits outnumber the residual risks, then move on to the next step - other resulting harms. The pertinent pieces of information required to justify the residual risk have to be contained inside the proper companion documents from the manufacturer. The outcome of this assessment should be documented in the management of risk file (Hegde, 2011).

Other produced hazards

Controls for risk which ought to be inspected to identify whether extra risks have been imposed. If any risk controls introduce any new hazards, the related risks should be evaluated. The outcome of such a review should be documented in the risk management

file (Hegde, 2011).

Completeness of risk evaluation

Manufacturers are required to make sure that an evaluation of any hazards which have been observed is carried out. The results of such assessments should be recorded in the management of risk file (Hegde, 2011).

5.1.4 Overview assessment of remaining risks

After implementing and verifying all necessary risk controls, the manufacturer should determine whether the residual overall risk presented for the medical device would be acceptable to the manufacturer utilising the standard set out in the risk management plan. Where the overall residual risk is determined to be unacceptable utilising the standard established in the risk management plan, the producer is expected to gather and examine evidence and documentation on the medical benefits of the anticipated usage purpose to establish whether they exceed the overall residual risk. When this evidence does not support the conclusion that the medical benefits outnumber the overall residual risk, it remains an unacceptable risk. The outcome of the overall residual risk assessment shall be documented in the management of risk file (Hegde, 2011).

5.1.5 Risk management report

Outcomes of the risk management process should be documented on risk management reports. There should be a risk management report that offers traceability and risk analysis for each risk, risk assessment, application, and validation of measures for risk control, and an acceptable evaluation of remaining risks. The risk management report should be part of the management file (Hegde, 2011).

5.1.6 Information on later production

Hegde (2011) stated that Producers are expected to develop and implement procedures for the systematic review of data acquired at the post-production stage in relation to medical devices or similar equipment. This information should be evaluated for safety relevance, in particular the following:

a) If there is a pre-existing unknown hazard.

b) If the hazard risk is estimated to be outside the range of no longer be acceptable.

c) If the initial assessment is invalid.

The evaluation results should be fed back as input to the risk management process if any of the above conditions are met. Based on this safety-related information, appropriate steps for reviewing the risk management process for the medical device should be considered. If residual risk or its acceptability is likely to change, an evaluation should be made of the impact on previously implemented risk control measures (Hegde, 2011).

5.2 The implementation of QMS

This study was conducted by developing a quality management system to support medical device manufacturers with the risk management process mentioned in ISO 14971. This quality management system is developed using the client-server model, where the main work is the development of the client side, while the server side only does the work of data storage. The two most important components of the client are risk procedure management component and FMEA table management component. Risk Procedure management component can be used to do risk analysis and FMEA table management component can be used to do risk evaluation and risk control.

5.2.1 Three Layers Architecture

Figure. 5.2 and Figure. 5.3 show the relationship of each class in the risk procedure management component and FMEA table procedure management component.

The system consists of three layers: presentation layer, logical layer and data layer. The presentation layer can be a UI layer, in this system each individual element is a widget, and each widget's display content is determined by the logic layer. The logical layer gets the data from the data layer and stores it in the Bloc, when the data stored in the Bloc layer changes it triggers the UI layer to update the associated widgets.

When the user uses the system and triggers an operation, the UI layer will first accept the operation request, then send the operation request to the data layer to request data, the data layer will get the data from the Server and return the data to the logic layer, the logic layer will process the data and then send it to the UI layer for display, finally completing an operation of the user.

5.3 Using QMS To Do Risk Management for A Data Logger

This section will show how we use QMS to manage risk for data loggers through a case study. As we have designed and developed a QMS based on the risk management process in ISO 14971, our risk management will be in compliance with the medical device standard. In addition, we will also identify some of the risks or potential risks of the medical device data logger used in this case. data logger manufacturers can minimise the risks of the device before it is finally delivered for use.

5.3.1 Data Logger

A data logger which is an electrical device that captures data over a period of time or in relation to position and has built-in sensors (Purwadi et al., 2011). They are growing in number but are not exclusively based on digital processors (or computers). They are usually small, battery-powered, convenient to carry and provided with a microprocessor, built-in memory for data storage and sensors. Whereas a few data loggers are connected to a PC and utilise software that allows the data logger to inspect and analyse the data collected, others feature a device with a local interface (keyboard, LCD display) that allows them to function as stand-alone devices.

5.3.2 Management of risk solutions

A top priority of the programme is the preparation of a risk management programme covering the product lifecycle, from the planning and construction phase to the disposition phase. This programme includes:

a) Risk acceptance standard for establishing acceptable risk.

b) The range of risk management activities in the plan, and the life-cycle stages to be applied to each element of the plan.

c) Distribution of responsibility and authority.

d) Review requirements for risk management activities.

e) Activities of verifying risk control measures.

f) Activities relating to assembling and reviewing pertinent manufacturing and post-production materials.

5.3.3 Acceptability of risk standard

ISO 14971 does not provide a definition of acceptable risk levels or acceptability criteria. The equipment producer should use this definition depending on the type of equipment. The risk acceptability criteria used in this procedure is shown in Figure. 5.4 below. Descriptions, definitions, criteria, and explanations of acceptability are shown directly below the diagram.

5.3.4 Risk Management Activities

Table. 5.1 lists the activities related to risk management (based on the 13 steps outlined in ISO 14971) carried out during the various stages of a data logger development project. The activities enable the identification of hazards, the estimation and assessment of associated risks, the control of those risks, and the monitoring of the effectiveness of controls throughout the life cycle of the product.

5.3.5 Instruments and frameworks utilised in the project

A QMS which is a web system based on the architecture in the previous chapter was developed for risk management. It is developed using Google's cross-platform framework flutter and has two main pages: the Risk Program Management page and the FMEA Form Management page.

Risk Procedure Management Page

The first step of risk analysis in the risk management process introduced at the beginning of this chapter includes Intend purpose identification, hazard identification and risk estimation. Figure. 5.5 is a screenshot of the Risk Procedure Management Page of QMS. It shows that the user can edit the harm for purpose identification, and edit the severity

	Activity	Project Phase	Input	Output
1	Identify product use, requirements and characteristics	Requirements and Planning	Product Requirement Document	Initial Hazards Analysis
2	Complete answers to questions contained in ISO 14971:2007 Medical devices-Application of risk management to medical Device, Annex C	System Design	ISO 14971:2007 Annex C question set User/Misuse model, IEC62366 - Medical device-Application of usability engineering to medical devices	Use/misuse scenarios and device information
3	Identify product hazards/harm, causes of hazards/harm and misuse	System Design	Functional Failure Analysis; Complaint Information; Medical Device Records (MDR); ISO 1491:2007 Annex C question set	Detailed Hazards Analysis and initial risk assessment
4	Research and document field events and recalls	System Design		Field Failures
5	Estimate and assign risk levels for each hazard	System Design	Panel of risk expers	Harm outcomes table in risk assessment document
6	Identify the control measures to be implemented in the design	System Design	Design engineers, risk expers	Control Measures (CM) Section of risk assessment
7	Create detailed failure analysis of device hardware, if needed	Detailed Design	dFMEA (Component level)	Detailed Failure Analysis
8	Monitor testing of control measures	Detailed Design	Verification & Validation (V&V) Test Plans, Control Measures Trace Matrix	Update "Control Measures (CM) Section" in risk assessment
9	Create production, supplier and service risk control plans	Detailed Design	Production, supply-chain and service related CM's from RA	Validation reports
10	Verification of control measures	Verification	V&V test reports, Quality Centre data, Post production support groups, Risk control outputs	Control Measures (CM) verification table
11	Produce Final Risk Assessment Document	Validation	All previous phases of the Risk Management Program	Final Risk Assessment
12	Periodic quality reviews of devices in the field, to verify/validate data in risk assessment	Post Transfer	Field Complaint, Post Market Risk Assessments	Identification of quality and risk associated items that are showing early trends of higher than expected frequencies. Updated risk assessment file

Table 5.1: Risk Management related activities

and probability for hazard identification. The table at the bottom of the figure is the risk estimation table.

FMEA table Management Page

The FMEA table management interface is used to handle risk evaluation and risk control, which are the second and third steps of the risk management process, respectively.

5.3.6 Management of risk documents

The risk control document contains the management plan, initial risk analysis, utilisation/misuse model, extended risk analysis, FMEA, risk evaluation, monitoring matrix of measures of control, report on verification of measures of control, control plan for production-supplier-services risks, post-market risk assessment plan and final risk report. Continuous updating of the management of risk profile as new risks are identified or changes in the occurrence/severity of existing risks are made and updated as required. It is often practical to undertake a potentially post-market risk evaluation after a product has been launched to determine whether on-site action is required in the event of negative events occurring on site.

5.4 The Support from QMS for Quality Attributes

In Section. 4.1, We introduced architectural significant requirements including quality attributes and we selected three quality attributes that are relevant to our requirements: learnability, user error protection and modifiability. In this section we would discuss the association of the QMS we developed with these three quality attributes.

5.4.1 Learnability

In the previous section we have mentioned that learnability is the least important quality attribute of the system, but the learnability of the system with some details have been achieved.

The main way to achieve learnability in this system is to reduce the learning time before users use the system. Reducing the learning time of the system means that we have a user-friendly UI and some instructions to reduce the user's troubles.

5.4.2 User Error Protection

User Error Protection is a critical quality attribute for any system as it has a direct impact on the user experience. If a simple user action can cause the system to crash, the user-friendliness of the system will be very low.

In order to reduce system risks caused by user operations. The user action was monitored to check if it is correct or the user would be asked to retract the action.

5.4.3 Modifiability

For any system, modifiability is considered to be one of the most significant quality attributes, and systems have to go through one iteration after another. Therefore, the performance of a system in terms of modifiability has a long-term impact on the system.

A three-tier architecture was used to improve the performance of the system in terms of modifiability. When there are new requirements for the UI, only the presentation layer need to be modified. Only the business logic layer need to be modified in terms of the execution logic of the system changes. Also, only the server layer need to be modified to meet the changes of data.

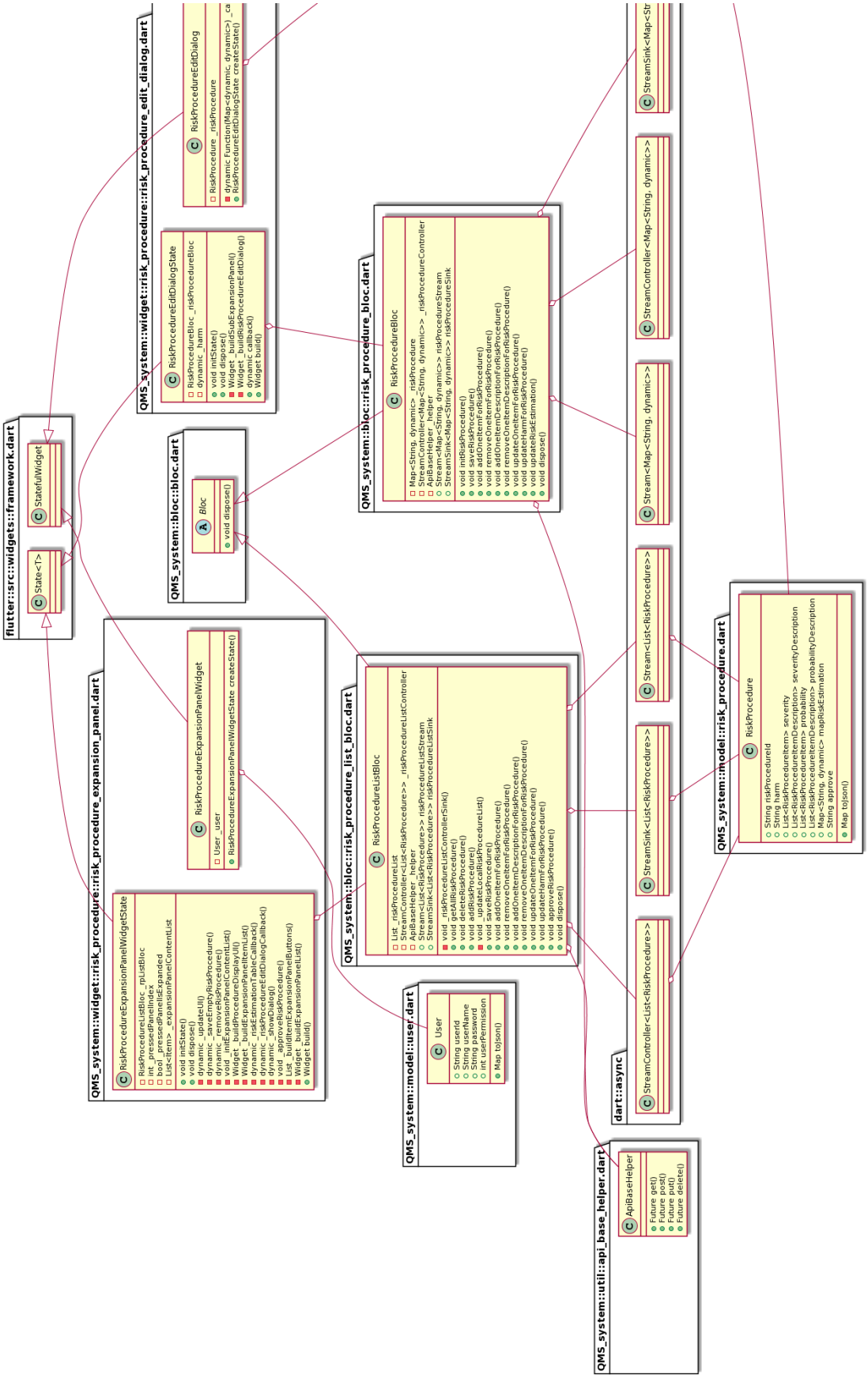

Figure 5.2: Class Diagram of Risk Procedure

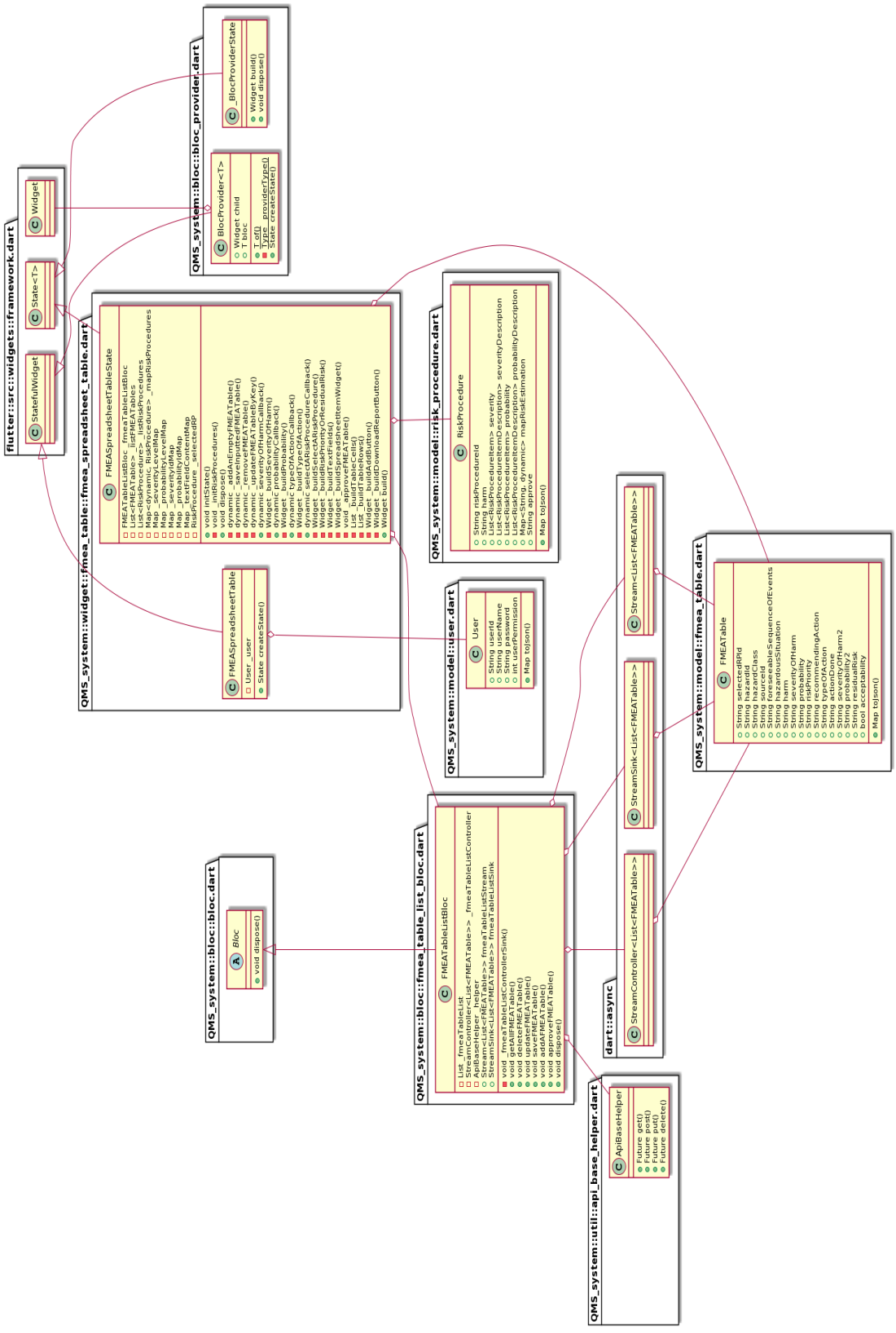

Figure 5.3: Class Diagram of FMEA Table

Risk Level Table		Probability Rating			
		1 - Unlikely	2 - Occasional	3 - Probable	4 - Frequent
Severity Rating	1 - Negligable	1	2	3	4
	2 - Minor	2	4	6	8
	3 - Major	3	6	9	12
	4 - Critical	4	8	12	16

Figure 5.4: Acceptability of risk standard

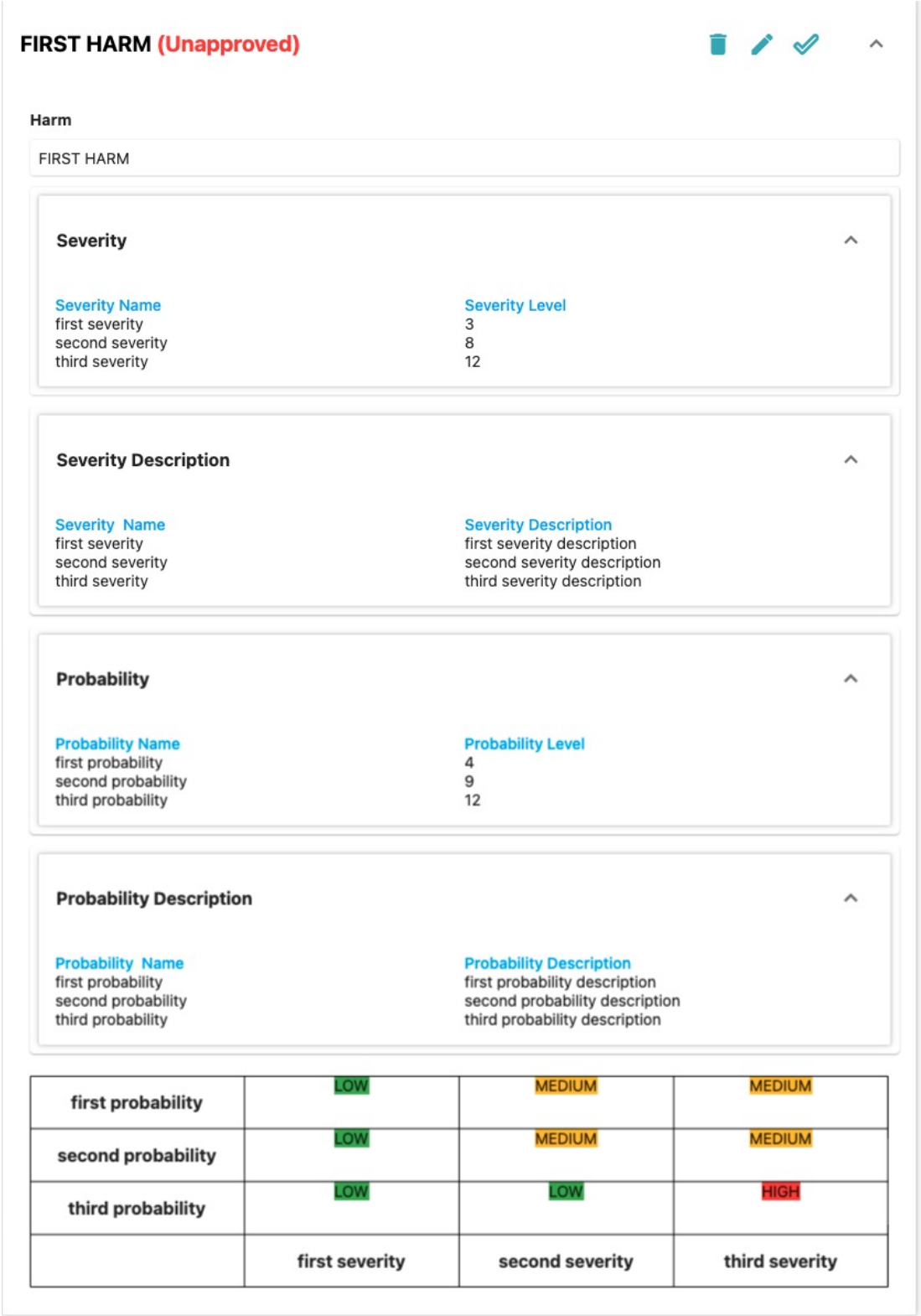

Figure 5.5: QMS Risk Procedure Management Page

Chapter 6

Evaluation and Answering RQs

6.1 Introduction

Architecture Tradeoff Analysis Method (ATAM) is an approach to the evaluation of architecture-level designs. It takes into account a variety of quality attributes. These attributes include modifiability, performance, reliability, and security. It provides insight into whether the architectural complete avatar is able to satisfy its requirements (Kazman et al., 1998). In this chapter, the architecture which was defined in Chapter. 4 based on ATAM was evaluated in Section. 6.2 and answered the RQ3 and RQ4 in Section. 6.3.

6.2 Evaluation

6.2.1 The Steps of the ATAM

Kazman, Klein and Clements (2000) divided ATAM into nine steps: Presentation of ATAM, Presentation of Business Drives, Presentation of Architecture, Identify Architectural Approaches, Generating a Utility Tree of Quality Attributes, Analyse Architectural

Approaches, Brainstorming and Prioritisation Scenarios, Analyse Architectural Approaches and Presentation of Results.

1. **Presentation of ATAM.** In this step, an evaluation team leader introduces the ATAM to the gathered stakeholders. This time is used to explain the process that each person will follow, leaving time to respond to questions with context and set expectations for the upcoming activities. What is critical is for each individual to know what is going to be captured. In this study, we show the two stakeholders, in this step, the steps of the ATAM, describing the activities involved in the ATAM and the outputs of this evaluation based on the technical aspects.

2. **Presentation of Business Drives.** The system being assessed in the evaluation requires understanding by all those involved. In this step, the project manager describes an outline of the system from a business context. The system itself must first be presented in a highly abstract manner, typically describing the most important functional requirements, technical constraints, business goals, primary stakeholders, and architectural drivers. The business drives were presented in Section. 4.1

3. **Presentation of Architecture.** The architecture will be presented in an appropriate degree of detailed by the architecture team. It is an important step as the quantity of architecture information available and the information documented will directly impact the likelihood of analysis and the quality of the analysis. The evaluation team will regularly need to specify additional architectural information that needs to be collected and recorded before a more substantive analysis can be performed. The architecture was displayed in Section. 4.2

4. **Identify Architectural Approaches.** The focus of ATAM is to analyse an architecture by understanding the architectural methods. In this step, these methods are

identified by the architect and caught by the analysis team, but not analysed. In the Chapter. 3, the ADD method was selected to design the architecture to iterate through the three quality attributes including (learnability, user error protection and modifiability). In addition, UML diagrams were drawn to show the different architectural views and architectural decisions.

5. **Generating a Utility Tree of Quality Attributes.** In this step, the assessment team works with the architecture team, managers, and customer representatives to identify, prioritize, and refine the most important quality attribute goals for the system. This is a critical step, as it guides the rest of the analysis. The architecture is most critical to the success of the system. We determined the priority and importance of the different quality attributes through multiple meetings involving different stakeholders, and used a utility tree Figure. 6.1 to show the outcome.

6. **Analyse Architectural Approaches.** As soon as the evaluation scope is defined by the utility tree, the evaluation team is able to explore architectural approaches to achieve key quality attributes. At this point, attention is given to documenting these architectural decisions and identifying their risks, sensitivities, and trade-offs.

   This study will discuss three architectural decisions which were mentioned in Table. 4.5, related to modifiability and the identification of their risks, sensitivities, and trade-offs as below:

   1). **Use the abstract common services.** All modules are reusable to the maximum extent possible. Minimal code changes are required if the system needs to be modified. In the case of adding system features, new modules can be added without affecting the rest of the system.

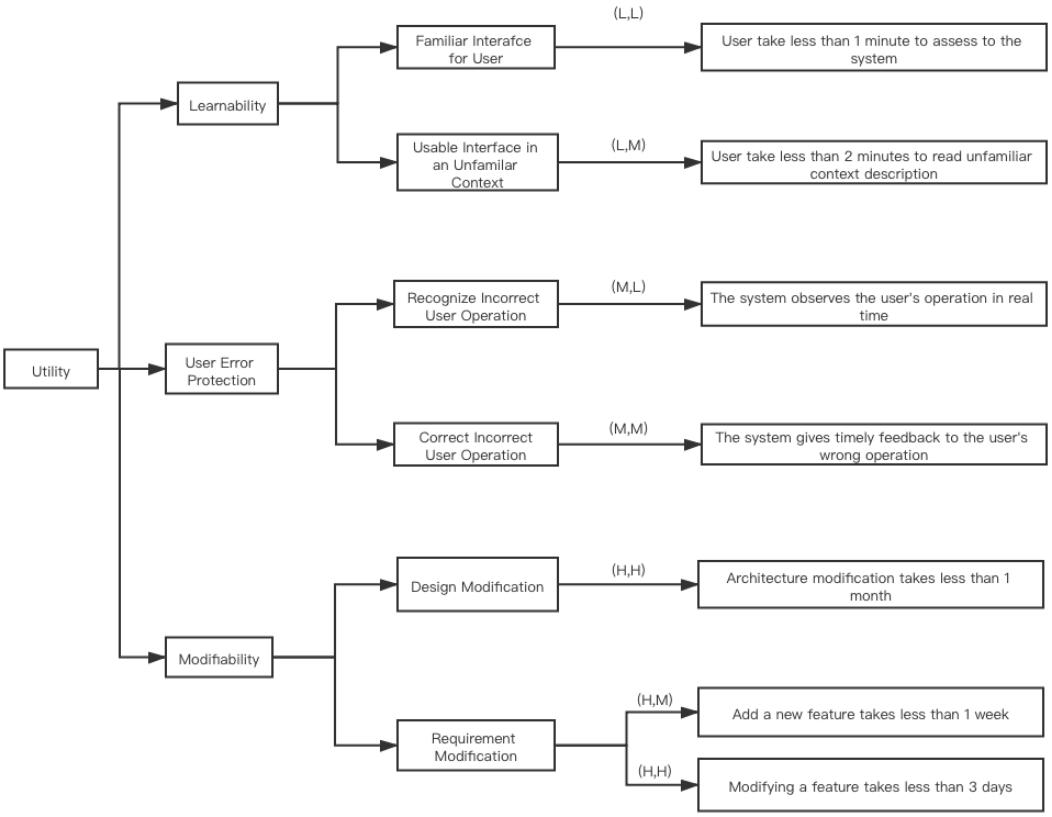

Figure 6.1: Utility Tree

2). **Use an intermediary.** When there are any dependencies between two modules, it is necessary to use a middleware to manage the activities involving the dependencies. Even if one of the modules needs to be modified, the performance of the other module will not be affected. If a new dependent module needs to be added then only the middleware needs to be modified.

3). **Use the configuration file.** For data that needs to be initialised during the system boot process, we need to use a configuration file to allow your system to be loaded at boot time and used where needed. If a configuration parameter needs to be changed then simply add the change or delete the required parameter.

7. **Brainstorming and Prioritisation of Scenarios.** Scenarios are the engine that

drives the testing phase of the ATAM. It proved to be a great tool for discussion and brainstorming to generate a set of scenarios when more stakeholders came together to participate in the assessment mechanism. A prioritization of these scenarios is necessary once they have been collected. This research generally accomplish this through a voting process in which each stakeholder receives as many votes as 30% of number of scenarios, then rounding up. The utility tree Figure. 6.1 shows that modifiability is most important quality attribute.

8. **Analyse Architectural Approaches.** In this step, step 6 was reiterated and the freshly generated top-ranked scenarios to the architectural building blocks found so far were mapped. If new pieces of information were discovered, then this is a failure of our practical tree exercise and of the architectural approach it leads us to study.

   For the first architectural decision - use the abstract common services. This is considered an architectural tradeoff point as achieving a high level of system reusability is complex and requires a high level of expertise.

   For the second architectural decision - use an intermediary. It should be a non-risk point in the architecture, as a change in either of the two modules on which the middleware manages dependencies will not affect the performance of the other module.

   For the third architectural decision - configuration file. It should be a non-risk point in the architecture, as the system can be started normally as long as the configuration file has exactly the proper configuration parameters.

9. **Presentation of Results.** Finally, the information collected from the ATAM needs to be summarized and fed back to the stakeholders. ATAM outputs include architectural approaches, a set of scenarios and their prioritization, a set

of attribute-based questions, utility tree, the risks discovered, the non-risks documented and the sensitivity points and tradeoff points found. All architectural decisions are already presented and analysed, and no modifications or additions are required.

6.3 Answering Research Questions

As previously mentioned, this research have four research questions, RQ1 and RQ2 have been answered explicitly in systematic literature review. In this section, we will discuss the answers to the remaining two questions:

RQ3 How can a novel QMS be architected to support the requirement identified in RQ1?

RQ4 How efficient is the QMS for managing requirements from medical standards?

6.3.1 Research Question 3

In Chapter. 2, the requirements in the medical device standard ISO 14971 were linked to the architecture documentation (ISO 25010 and 4+1 View model). However, the risk management process mentioned in ISO 14971 includes almost all the requirements in ISO 14971.

In Chapter. 4, the architecture of the QMS was designed in two steps:

- **Identifying Architecturally Significant Requirement** In Section. 4.1, it was found that there was a need to figure out constraints and core features to prepare for the subsequent architectural design. Developers worked with stakeholders to come up with functional requirements to meet the risk management process in ISO 14971.The project manager supported us with several business constraints, while

the developers, under the guidance of the architect, provided several technical constraints. Also, we followed the QAW steps to derive three quality attributes.

- **Architecture Design Using ADD** In Section. 4.2, using the Attribute-driven Design Method, after two iterations, we identified architectural drivers based on architecturally significant requirements, and then made the corresponding architectural decisions for these drivers. We were also able to show these architectural decisions in terms of architectural views.

6.3.2 Research Question 4

In Chapter. 5, we presented a QMS which was developed following the ISO 14971 risk management process. As Figure. 5.1 showed, risk management process in ISO 14971 includes four steps: Risk Analysis, Risk Evaluation, Risk Control and Post-production process.

- **Risk Analysis** The first functional requirement for QMS is the ability for users to add or edit Risk Procedures to define how many risk factors are included in a Procedure and the level of risk factors.

- **Risk Evaluation** After defining the Risk Procedure, the user is provided with an automatically updated risk evaluation table. This table shows the user the severity level of the different risks and the frequency with which the risk is likely to occur. After a Risk Procedure is modified, QA will determine if the modification is valid.

- **Risk Control** The second functional requirement is Failure Mode and Effects Analysis (FMEA). The user can select an already defined Risk Procedure to do Risk Control. The user can select each individual risk in the Risk Procedure to re-evaluate whether it is acceptable or not. Ultimately the validity of the modified FMEA table will be reviewed by QA and a decision will be made.

- **Post-production process** This study focuses on Risk management and Post-production process will be discussed in future work.

Chapter 7

Conclusion and Future works

This chapter summarizes the entire study. This chapter summarizes the entire study, a design architecture based on medical device standards and software architecture documents and using the ADD methodology, a QMS was developed and an architecture evaluation was done using ATAM.

The rest of the chapter is arranged as follows. Section. 7.1 provides a summary of the entire study and discusses what has been shown in the previous chapters. Section. 7.2 addresses the main contributions of this study. Section. 7.4 provides two possible orientations for future work.

7.1 Summary

The main purpose of this study is to explore the relationship between medical device standards and software architecture and to design a software architecture to develop a quality management system.

The relationship between medical device standards and software architecture documentation and the relationship between medical device reference architectures and software architecture documentation is discussed in the systematic literature review

chapter. We classify the requirements in ISO 14971, ISO 13485, IEC 62366 and IEC 62304 based on the sub-features in ISO 25010 and the four views in the 4+1 view model, giving us an idea of the extent to which the requirements in the medical device standards support the software architecture.

In the architecture design process, we first identify architecturally significant requirements including functional requirements, constraints and quality attribute. The architecture of the QMS was designed using the attribute-driven design method in two iterations and the architecture view was used to present the architecture. This provided a blueprint for the next step in the development of the QMS.

In the QMS case study chapter, we show the development of a quality management system based on the previously designed architecture. This system incorporates the risk management processes mentioned in the medical device standard ISO 14971. Users can use this system to reduce the risk of medical devices.

At the end of this study, an evaluation of QMS architecture using ATAM was conducted, and the third and fourth research question were answered.

7.2 Contributions

This research have done a systematic literature review showing the relationship between medical device standards and architecture documents and the relationship between recent reference medical device architectures and architecture documents. Also, the design and evaluation of QMS architecture, the development of QMS have led to the contributions.

7.2.1 Connection of Medical Standards and Architectural Documents

The requirements of the medical device standards (ISO 14971, ISO 13485, IEC 62366 and IEC 62304) were classified to link them to the architecture documents (ISO 25010 and 4+1 View model). A mapping table was then made for this classification. These mapping tables show the relationship between the medical device standards and the software architecture aspects. They support the software architecture based on medical device standards.

7.2.2 Connection of Medical Standards and Reference Medical Device Architectures

The study searched for recent reference medical device architectures and studied their support for software architecture documents (ISO 25010 and 4+1 View model), and we presented this support in a mapping table.

7.2.3 A Novel Architecture of QMS

The entire architectural design process followed the medical device risk management standard ISO 14971. Architectural significant requirements including risk management requirements, constraints and quality attributes were identified through multiple meetings with stakeholders. Then ADD architecture design methodology was used to present each architectural driver in two iterations using several architectures. This architecture was used in the development of the QMS for this study to enable the QMS to better support risk management for manufacturers.

7.2.4 A Quality Management System

A quality management system was developed based on QMS architecture. This system simulates the risk management process mentioned in ISO 14971 and allows users to use this QMS for risk analysis, risk evaluation and risk control. This makes it easier for manufacturers to minimize the risk of medical devices.

7.3 Limitation

7.3.1 Relationship Between Reference Medical Architecture

In Chapter. 2, medical device standards in relation to reference medical architectures were presented. However, we only found five medical architectures, which is not enough. In addition, the medical architectures we found are not well supported by the software architecture documentation. The best-performing reference medical architecture had only 40% support for development view. Also, these medical architectures are used in the IoT domain, i.e. to develop software systems based on medical devices. Then one of our final outcomes in this study is a software system for risk management that does not use the reference medical architectures we found.

7.3.2 Architecture of QMS

In Chapter 4, we displayed the architecture of QMS designed using the ADD method. With no mature software for reference and no experienced software architects to guide us in the full spectrum, we have made some somewhat subjective architectural decisions. These decisions may not have been very comprehensive and well thought out. It was the first time we used the ADD approach to design the architecture, and there may be some inappropriate use of the approach.

7.3.3 Implementation of QMS

In this research, we have developed a quality management system. Due to time constraints, our system was developed in the shortest possible time. Therefore, the system has some shortcomings in terms of user experience and security. In terms of user experience, The individual interfaces of QMS are implemented in the simplest and most time-saving way. In terms of security, There is no information protection mechanism for the system to operate and read important information. Also, the system was not tested systematically. These aspects will be improved in the future work.

7.4 Future works

This section shows two possible directions for future research, which are discussed below.

7.4.1 The next step in the development of QMS

The quality management system developed in this study only implements the management of risk procedure and FMEA table at present, and there may be more needs to follow. In addition, the QMS at this stage is only applicable to the website, and in the future, we will develop a QMS that can be adapted to the cell phone or computer side to provide manufacturers with a more convenient risk management tool. Finally, in the early stage of QMS development, the project manager wanted us to use domain-specific language to develop a QMS that would be more user-friendly for non-developers to edit the UI interface. In the future, subsequent developers familiar with the domain-specific language could do part of the work.

7.4.2 IoT Software Based on Medical Standards

The main result of this study is a QMS architecture and QMS, but QMS is only for medical device manufacturers to provide device risk management. In the future perhaps we can extend the reference medical architectures found in RQ2 by implementing requirements in medical devices to develop medical device IoTs that meet medical device standards. In addition, in Chapter 2, we just found five reference medical architectures and these five architectures are not well supported by the software architecture documentation (ISO 25010 and 4+1 Views Model). Therefore, for the future development of medical device software, medical reference architectures that are well supported by software architectural documents will be collected. These collected architectures will be extended by adding the requirements of medical device standards. This will enable the development of medical device software based on the extended architectures to have a better performance in terms of security. In addition, specific examples will be given to explain the role of software architecture documentation and medical device standards in research in the field of medical IoT software.

7.4.3 Improving the Risk Management Process in QMS

This study focused on risk management, and we did not implement the post-production process in the QMS. In future QMS developments, new functional and non-functional requirements should be proposed to support the post-production process to make the QMS better and allow further risk reduction for medical devices.

7.4.4 Measuring Quality Attributes

This study mentions three quality attributes including Learnability, Modifiability and User error protection as functional requirements for architecture design to improve the quality of the architecture. However, we have not measured these quality attributes. In

future work ISO 25023 will be used as a reference standard for the measurement of quality attributes.

7.4.5 Providing Better Support on Using ATAM

Due to the impact of covid-19, we did not have enough time to bring the quality of the paper up to a higher level, and one of the most important improvements needed would have been the use of ATAM in the architecture assessment chapter. The nine steps of ATAM will be described in detail in future work.

7.4.6 Extending Systematic Literature Review

The SLR chapter of this study was completed in the early stages of this research work, due to the own reasons of researcher for not making a good plan of SLR, which also included the search of relevant reference papers and the collation and use of the papers searched. One of the biggest problems was the relatively small number of reference papers searched. In the future, we will use improved search strings to obtain more references and collect the possible evidence from them to answer research questions.

7.4.7 Providing Systematic Explain on Quality Attribute Workshop

In the chapter on architectural design, three tables of quality scenarios are shown. These three quality scenarios are derived by means of a quality attribute workshop (QAW) with different stakeholders, but we do not give details of the QAW. The QAW will be presented in detail in future work.

References

Aguwa, C. C., Monplaisir, L., Sylajakumari, P. A. & Muni, R. K. (2010). Integrated fuzzy-based modular architecture for medical device design and development. *Journal of Medical Devices*, *4*(3).

Alturki, A., Gable, G. G. & Bandara, W. (2013). The design science research roadmap: in progress evaluation. *PACIS 2013 proceedings*.

Bachmann, F. & Bass, L. (2001). Introduction to the attribute driven design method. In *Software engineering, international conference on* (pp. 0745–0745).

Barbacci, M. R., Ellison, R., Lattanze, A. J., Stafford, J. A. & Weinstock, C. B. (2003). *Quality attribute workshops (qaws)* (Tech. Rep.). CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST.

Baskerville, R. L. (1999). Investigating information systems with action research. *Communications of the association for information systems*, *2*(1), 19.

Bass, L., Clements, P. & Kazman, R. (2003). *Software architecture in practice*. Addison-Wesley Professional.

Bass, L., Klein, M. & Bachmann, F. (2001). Quality attribute design primitives and the attribute driven design method. In *International workshop on software product-family engineering* (pp. 169–186).

Bras Da Costa, S., Beuscart-Zéphir, M.-C., Christian Bastien, J.-M. & Pelayo, S. (2015). Usability and safety of software medical devices: need for multidisciplinary expertise to apply the iec 62366: 2007.

Chen, L., Babar, M. A. & Nuseibeh, B. (2012). Characterizing architecturally significant requirements. *IEEE software*, *30*(2), 38–45.

Corns, S., Thukral, A. & Thukral, V. (2014). 5.1. 1 parametric analysis through a model-based reference architecture for medical device development. In *Incose international symposium* (Vol. 24, pp. 406–417).

Dresch, A., Lacerda, D. P. & Antunes, J. A. V. (2015). Design science research. In *Design science research* (pp. 67–102). Springer.

Easterbrook, S., Singer, J., Storey, M.-A. & Damian, D. (2008). Selecting empirical methods for software engineering research. In *Guide to advanced empirical software engineering* (pp. 285–311). Springer.

Fulton, R. & Vandermolen, R. (2017). Chapter 4: Requirements - writing requirements. *Airborne Electronic Hardware Design Assurance: A Practitioner's Guide to RTCA/DO-254*, 89–93.

Gable, G. G. (1994). Integrating case study and survey research methods: an example

in information systems. *European journal of information systems*, *3*(2), 112–126.

Harp, S., Carpenter, T. & Hatcliff, J. (2018). A reference architecture for secure medical devices. *Biomedical instrumentation & technology*, *52*(5), 357–365.

Hegde, V. (2011). Case study—risk management for medical devices (based on iso 14971). In *2011 proceedings-annual reliability and maintainability symposium* (pp. 1–6).

Hevner, A., March, S. T., Park, J., Ram, S. et al. (2004). Design science research in information systems. *MIS quarterly*, *28*(1), 75–105.

IEC62304. (2015). *Iec 62304 – medical device software – software life cycle processes* (ISO). International Organization for Standardization.

IEC62366-1. (2015). *Iec 62366-1:2015 medical devices — part 1: Application of usability engineering to medical devices* (ISO). International Organization for Standardization.

ISO13485. (2016). *Iso 13485:2016 medical devices — quality management systems — requirements for regulatory purposes* (ISO). International Organization for Standardization.

ISO14971. (2019). *Iso 14971:2019 medical devices — application of risk management to medical devices* (ISO). International Organization for Standardization.

Kasparick, M., Schmitz, M., Andersen, B., Rockstroh, M., Franke, S., Schlichting, S., … Timmermann, D. (2018). Or. net: a service-oriented architecture for safe and dynamic medical device interoperability. *Biomedical Engineering/Biomedizinische Technik*, *63*(1), 11–30.

Kazman, R., Klein, M., Barbacci, M., Longstaff, T., Lipson, H. & Carriere, J. (1998). The architecture tradeoff analysis method. In *Proceedings. fourth ieee international conference on engineering of complex computer systems (cat. no. 98ex193)* (pp. 68–78).

Kazman, R., Klein, M. & Clements, P. (2000). *Atam: Method for architecture evaluation* (Tech. Rep.). Carnegie-Mellon Univ Pittsburgh PA Software Engineering Inst.

Kitchenham, B. (2004). Procedures for performing systematic reviews. *Keele, UK, Keele University*, *33*(2004), 1–26.

Kruchten, P. B. (1995). The 4+ 1 view model of architecture. *IEEE software*, *12*(6), 42–50.

Laukkarinen, T., Kuusinen, K. & Mikkonen, T. (2017). Devops in regulated software development: case medical devices. In *2017 ieee/acm 39th international conference on software engineering: New ideas and emerging technologies results track (icse-nier)* (pp. 15–18).

Lincoln, J. E. (2009). Product risk management under iso 14971: 2007. *Journal of Validation Technology*, *15*(4), 10.

Liu, J. W., Wang, B., Liao, H., Huang, C., Shih, C.-S., Kuo, T. & Pang, A. (2005). Reference architecture of intelligent appliances for the elderly. In *18th international conference on systems engineering (icseng'05)* (pp. 447–455).

Lundberg, L., Bosch, J., Häggander, D. & Bengtsson, P.-O. (1999). Quality attributes in software architecture design. In *Proceedings of the iasted 3rd international conference on software engineering and applications* (pp. 353–362).

Malhotra, R. & Chug, A. (2016). Software maintainability: Systematic literature review and current trends. *International Journal of Software Engineering and Knowledge Engineering*, *26*(08), 1221–1253.

Meng, S., Pan, Z., Li, W., Xie, S., Liu, C., He, K. & Yang, H. (2010). The "4+ 1 "view model on safe home system architecture. In *2010 ieee international conference on software engineering and service sciences* (pp. 352–355).

Noor, K. B. M. (2008). Case study: A strategic research methodology. *American journal of applied sciences*, *5*(11), 1602–1604.

Nord, R., Wood, W. & Clements, P. (n.d.). *Integrating the quality attribute workshop and the attribute-driven design method* (Tech. Rep.). technical report, CMU/SEI-2004-TN-017.

Purwadi, A., Haroen, Y., Ali, F. Y., Heryana, N., Nurafiat, D. & Assegaf, A. (2011). Prototype development of a low cost data logger for pv based led street lighting system. In *Proceedings of the 2011 international conference on electrical engineering and informatics* (pp. 1–5).

Raharja, I. M. S. & Siahaan, D. O. (2019). Classification of non-functional requirements using fuzzy similarity knn based on iso/iec 25010. In *2019 12th international conference on information & communication technology and system (icts)* (pp. 264–269).

Regan, G., Mc Caffery, F., Mc Daid, K. & Flood, D. (2013). Medical device standards' requirements for traceability during the software development lifecycle and implementation of a traceability assessment model. *Computer Standards & Interfaces*, *36*(1), 3–9.

Schensul, S. L., Schensul, J. J. & LeCompte, M. D. (1999). *Essential ethnographic methods: Observations, interviews, and questionnaires* (Vol. 2). Rowman Altamira.

Simon, H. A. (1996). *The sciences of the artificial 3rd ed.* MIT Press Cambridge.

Viriansky, Z. Y. & Shaposhnikov, S. O. (2019). Quality assurance of quality management systems. In *2019 international conference" quality management, transport and information security, information technologies"(it&qm&is)* (pp. 323–325).

Weber, R., of Australia, A. A. & Zealand, N. (1997). *Ontological foundations of information systems*.

Wojcik, R., Bachmann, F., Bass, L., Clements, P., Merson, P., Nord, R. & Wood, B. (2006). *Attribute-driven design (add), version 2.0* (Tech. Rep.). CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST.

Zhang, P. & Xi, J. (2008). Attribute-driven design of mdx compiler. In *2008 international conference on computer science and software engineering* (Vol. 2, pp. 508–511).