Software development and risk management in the safety critical medical device domain

Lindholm, Christin

2009

Software Development and Risk Management in the Safety Critical Medical Device Domain

Christin Lindholm

Licentiate Thesis, 2009

Department of Computer Science
Lund University
Faculty of Engineering

ISSN 1652-4691
Licentiate Thesis 9, 2009
LU-CS-LIC:2009-1

Department of Computer Science
Faculty of Engineering
Lund University
Box 118
SE-221 00 Lund
Sweden

Email: christin.lindholm@cs.lth.se

To Henrik, Niklas, Johan and Hans

Abstract

The healthcare sector is one of the fastest growing economic sectors of today. The medical device domain is one part of that sector. An increasing part of functionality in medical devices and systems is implemented in software and many features should not be possible to implement without software.

The use of medical software is an inherent risk to the patient and the outcome of a failure can vary from death to almost no effect at all. Risks and risk management is closely connected to medical device domain and it is crucial to all medical device companies to have a good risk management process. It is also stated in law that the companies developing medical devices must have a risk management process.

One part of the research in this thesis focuses on the current state of practice in the medical device domain. As a result of this research, the need for high quality software in this domain has been identified and also the needs for new techniques, methods and processes to further improve software quality in the medical device domain. The results have been used to derive a set of requirements on new processes, methods and techniques in the area, to be used by researchers as a guide in the development of more adapted processes, methods and techniques for software development in the medical devices domain.

The other part of the research in this thesis focuses on risk and is based on two experiments. A number of decisions regarding risks are taken during software project risk management and it is the people involved that make the decisions. Different people's opinions about the importance of identified risks are investigated in an experiment and it is concluded that different participants have different opinions about how serious risks are, concerning faults remaining after testing. Probably it is possible to generalise this and conclude that in the software engineering process different people are more or less risk seeking.

From the second experiment it could be concluded that multiple roles and thereby different experiences will affect the risk identification process. Involving multiple roles will result in a more complete set of identified risks than if only one role is included.

Acknowledgements

First of all I would like to thank my supervisor Dr. Martin Höst for his excellent support and guidance during the work of this thesis and tanks also to my assisting supervisor Prof. Per Runeson.

I wish to thank Gyllenstienska Krapperupsstiftelsen, who has partly funded the work presented in this thesis. I am very grateful for the funding and the possibilities that have given me.

Many thanks to all my past and present colleagues at LTH Ingenjörshögskolan, Campus Helsingborg, the Software Engineering Research Group and the Department of Computer Science, for your support and encouragement. My thanks go also to all participants in the interviews and experiments.

Special thanks to my family and all my friends. I am grateful to Golan and Arne, my parents for all their love and for being there for me all my life. I would also like to bring my very special thanks to my lovely children Henrik and Niklas who give me so much joy in life and my husband Hans. Gunnel, thank you for your support with the children and that you are always able to stand in with short notice.

Finally I would like to thank Stoika, for all our interesting discussions, your inspiration, support and for always being there for me, and Birgitta for our kitchen discussions and

encouragement. Without you two this had not been possible, you will always have a special place in my heart.

Tank you all!

"In me the tiger sniffs the rose".
(Siegfried Sassoon)

Contents

Part A: Introduction

Part B: State of Practise in Software Development in Medical Devices

Paper I: State of the Practice in Software Development for Medical Device Production

Paper II: Development of Software for Safety Critical Medical Devices – an Interview-Based Survey of State of Practice

Part C: Risk Management in the Medical Devices Domain

Paper III: Different Conceptions in Software Project Risk Assessment

Paper IV: Risk Identification by Physicians and Developers - Differences Investigated in a Controlled Experiment

Part A

Introduction

Introduction

1 Background

Medical devices are instruments or material used on human beings to diagnose or treat diseases and other conditions. Software can be an integrated part of a medical device or software can affect the use of a medical device in some way. In the area of software and medical devices, the technique and operator are interacting and create an entirety. The software developers, the marketers, the managers, the users i.e. the physicians and nurses and the patients that are going to be treated and helped with the use of the different medical devices on the market are the operators. All the individuals creating the entirety are driven and influenced by personal knowledge, views, prosperity, power, manners and habits, written and unwritten laws and moral and ethical values.

All the individuals in the chain of developing and marketing a medical device are important. The medical device development, and the medical device are important but the medical device itself is of no use without the presence and knowledge of the user. It is important to use the device within all the functionality

It provides but it is also important to know that the device itself does not exceed the human eye in the interaction with the patient.

Developers in their environment influenced by techniques and special development processes develop the medical devices and the released medical devices are then used in a new environment with new people with other experiences and values. On the medical device's way from development to market, the device is also affected by business and marketing decisions. It should always be taken in account what the incentives are for the chain of people involved, what drives the producers of the medical devices and what drives the users.

Risk management, a general procedure for resolving risk, is an area strongly connected to the development and use of medical devices and it is important that the risk for patients and medical staff is as low as possible. The dilemma can be the trade-off between maximal treatment effect and the risk of injury and side effects, to decide when the risk is low enough and to know when all the important risks have been considered and handled. This is a dilemma dealt every day in the medical device domain.

The main goal of the research presented in this thesis is to understand and describe the development processes and quality assurance techniques that characterise the development of safety critical medical devices and how risk and risk management interact in this area. The goals of the research and future work are to find better support and adapted processes and techniques for software development in the medical device area, to bridge over the existing gap between the "different worlds", to understand the underlying mechanisms and how human behaviour will affect decisions and actions.

1.1 The medical device domain and software development

The healthcare sector is one of the fastest growing economic sectors of today [17] and the medical device domain is one part

of that sector. Medical devices and systems have been developed over many years but these types of products are now containing and based on more and more software. An increasing part of functionality is implemented in software and many features should not be possible to implement without software.

Software that runs medical devices or in some way are used together with medical devices are automatically classified in the same safety classification as the rest of the medical device and has to follow the same laws and regulations as the rest of the product [8]. These laws and regulations affect the medical devices developers' way of working.

Within software development there is a lot of possibilities but also difficulties. When developing software it is possible to make substantial changes late in the development process, which can be beneficial but can also cause serious incidents. Another matter that makes the software development difficult is the complexity of the software [6]. The complexity makes it hard for the software developers to develop fault free software and it also demands that the developers understands the major part of the product to be able to produce good work. Software is abstract and intangible which makes it difficult for persons that are not directly connected to the software development to perceive the quality of the software. The quality of the software in a product is very important and there is a special need for high quality in applications that are safety critical. Most medical devices are safety critical at different levels and in the medical device area a lot of work is made in the quality assurance area through using standards and techniques. Laws and standards that affect the quality assurance work are for example CE labelling and special laws and regulations from the Commission of the European Communities [8] and U.S. Food and Drug Administration (FDA) [40]. However not much documented research has been carried out on how these standards and techniques are used in the medical device domain. Because of

this a need to survey the standards and techniques used in connection with software development in the medical device area was identified.

1.2 Risk management

In the medical device domain the smallest fault or mistake can mean the difference between life and death. The use of medical software is an inherent risk to the patient and the outcome of a failure can vary from death to almost no effect at all.

Risks and risk management are closely connected to the medical device domain as well as other safety critical domains. In the development and use of safety critical systems there are different kind of risks to identify and handle in a correct way. It is crucial for all types of project planning and management to have a good risk management process but for all organisations that develop medical devices it is also regulated by law [8] that they must have a risk management process. However it is the company itself that have to design and implement this process and take the responsibility for the execution.

Risk identification is an important part of the risk management process especially in the development of medical software. It is important to get an as complete set of identified risks as possible. With this in mind we decided to investigate involving multiple roles in the risk identification process to investigate if it would result in a more complete set of risks.

In the risk management process a large number of decisions are made, for example decisions on how important an identified risk is. These decisions are affected by the participants risk tendency, if the participant is risk seeking or risk averse. The possibility of measuring different persons′ risk tendency are of interest as a part of the research about developing decision-support. The awareness of different persons′ risk tendency can be a way of improving the risk management process.

1.3 Outline of the thesis

The work in this thesis is divided and presented in three different parts. The first part is an introductory part, the second part focuses on state of practice in the medical device domain and the third part focuses on the risk management area:

- **Part A**. This is an introductory part that describes the research context and results. The, first section in Part A present the thesis outline and Section 2 contains the research goals of this thesis. Section 3 continues with a description of related work that put the research into context. This is followed by the research methodology and some general threats to validity in Section 4. The main contributions of this thesis are then presented in Section 5. followed by future works in Section 6.
- **Part B**. The second part of this thesis is based on one technical report (Paper I) and one research paper (Paper II) and contributes to the understanding of the state of practice in the medical device domain. Paper I describes the state of practice regarding software development in the medical device domain based on a international web survey followed by Paper II, that goes more in to depth regarding the state of practice in Sweden. The results presented in Paper II have also been used to derive requirements that can serve as guidance to researchers aiming at improving software processes, methods and techniques in the medical device domain. These requirements are presented in the end of Paper II.
- **Part C.** The third part of this thesis focuses on risk and includes two papers. The first paper in Part C, Paper III investigates in a controlled experiment different people's opinions about the importance of identified risks. The investigation is made by the use of utility functions. Paper IV investigates if there are any difference between users of and developers concerning risk identification.

The paper describes an experiment where software developers, physicians and medical device developers analysed the same risk scenario and reported the risks they found.

Part B of this thesis contains the following papers and reports:

PAPER I:
State of the Practice in Software Development for Medical Device Production
Christian Denger, Raimund L. Feldman, Martin Höst, Christin Lindholm and Forrest Shull
IESE-Report No. 071.07/E, Fraunhofer Institute for Experimental Software Engineering, February 2007.

Parts of Paper I have been presented from different point of views in two publications, not included in this thesis:

Software Engineering Techniques in Medical Device Development
Raimund L. Feldman, Forrest Shull, Christian Denger, Martin Höst and Christin Lindholm
In workshop on High Confidence Medical Devices, Software and Systems (HCMDSS) and Medical Device Plug-and Play (MD PnP), Boston, USA, June 2007, pp. 46-54.

A Snapshot of the State of Practice in Software Development for Medical Devices
Christian Denger, Raimund L. Feldman, Martin Höst, Christin Lindholm and Forrest Shull
In proceedings of International Symposium on Empirical Software Engineering and Measurement (ESEM), Madrid, Spain, September 2007, pp. 485-487.

PAPER II:
Development of Software for Safety Critical Medical Devices – an Interview-Based Survey of State of Practice
Christin Lindholm and Martin Höst
In proceedings of the eighth Conference on Software Engineering Research and Practice in Sweden (SERPS´08), Karlskrona, Sweden, November 2008.

Part C of this thesis contains the following papers:

PAPER III:
Different Conceptions in Software Project Risk Assessment
Martin Höst and Christin Lindholm
In proceedings of the Software Engineering Track at the 22:nd Annual ACM Symposium on Applied Computing (SAC), Seoul, *Korea,* March 2007, pp. 1422-1426.

PAPER IV:
Risk Identification by Physicians and Developers – Differences Investigated in a Controlled Experiment
Christin Lindholm and Martin Höst
Accepted for publication in proceeding of the ICSE 2009 Workshop on Software Engineering in Healthcare, Vancouver, Canada, May 2009.

2 Research focus

The main goal of the research presented in this thesis is to survey and understand the development processes and quality assurance techniques that characterise the development of safety critical systems in the medical device domain. Risk management is an important part of quality assurance and identified as a very important area for the medical device domain.
The overall research goal has been divided into the following main research aspects:

- What is the state of practice of the medical device domain with respect to the software development processes and software quality assurance techniques?
- What role does the software have in the medical device domain?
- How do organisations in the medical device domain guarantee the quality of the software in the medical devices?
- How can different people′s risk tendency be defined in an adequate way and how can the result be used to support decision in a risks management process?
- What difference can be established between the users of a system and the developers of a system according to risk identification?

The research in this thesis is guided by a vision to understand software development and risk management in the medical device domain. The intension is that this should lead to further research on specific support to software developers in the medical device domain and other domains.

Since there are little published before about state of practice in software development for medical device production a need for explorative studies to survey the current state of practise was

identified. The three first research aspects are therefore of survey nature and correspond to Part B of this thesis.

It is clearly indicated by related work that further research is needed in the medical device area. The related work points out, that software and its complexity in the medical device domain among with the accidents depending on software failures, increases.

Risk management is crucial to the medical device domain but risk management in this area seldom addresses the risks for software specifically. The two last research aspects presented above are ways to find different angles to support the area of risk and correspond to Part C of this thesis.

3 Background and related work

This chapter describes work related to software development in medical devices and systems, with a special focus on safety critical medical devices and systems. Safety can be defined as the freedom from exposure of danger or exemption from injury or loss [5]. According to Bowen and Stavridou [5] “safety in safety-critical computer systems is something that has to be designed in the system and danger must be designed out”. Safety cannot be considered as an add-on feature after the system has been developed. Safety is according to Sommerville [38] “the ability of a system to operate without catastrophic failure” and software that is safety critical falls into two classes: Class 1. Primary safety critical software where software is embedded as a controller in a system. An example of problem with primary safety is malfunction of software that causes malfunction of hardware. Class 2. Secondary safety critical software where software can indirectly result in injury. An example of problems with secondary safety is a medical database containing incorrect information about for example drug prescription.

This chapter is divided in three parts; Section 3.1 focuses on the medical device domain, Section 3.2 focuses on risk management and in the last Section 3.3 future challenges are discussed.

3.1 The medical device domain

3.1.1 The medical device domain is a complex domain

Garde and Knaup [12] state that health care is a complex domain for many reasons. There are a lot of difficulties associated with the domain that makes it complex. Garde and Knaup have studied reports on difficulties raised when application systems have been introduced in health care. Based on these reports they have identified several characteristics from the medical domain. Some of the characteristics described are for example, that the product of health care, meaning the treated patient cannot be categorised in a package. The reason why the patient cannot be categorised is that the patient for all practical purposes has an unlimited set of characteristics, which constantly change and interact. Decisions in the health care domain are sometimes according to Garde and Knaup based on little or unreliable evidence and circumstances change rapidly, which leads to the need of quick adjustments in the planning of actions. Another characteristic mentioned is that the majority of stakeholders in health care are non-technical professionals for example physicians, nurses, administrators etc and some of them are even avers against computers and IT.

The multitude of medical standards and medical terminology is another complicating factor that contributes to complexity. In Robert N. Charette´s article “Dying for data” [7] where Michael Rozen, vice chairman of the IEEE-U.S. Medical Technology Policy Committee, states that there are 126 ways to say “high blood pressure” and that agreeing on medical terminology is a large issue. In the U.S. the Healthcare Information Standards

Panel identified an initial set of 90 medical and technology standards, out of an original list of about 600. In the panel there are more than 190 organisations participating, representing customers, health care providers, government agencies and standard development organisations, which means that this is not an easy way to come to consensus on medical standards.

3.1.2 Software in medical devices

Software and embedded systems controlled and managed by software play an increasingly important role for the medical device industry and medical devices and systems play a more and more important role in health care [4, 27, 28]. Medical care is one of the traditional areas considered as safety critical. Safety critical systems are defined by Knight [23] as "those systems whose failure could result in loss of life, significant property damage or damage to the environment".

Embedded systems with special purpose computer systems have increasingly become predominant in a rage of safety critical applications for example in medicine, nuclear power plants, aviation and aerospace industries [23]. These embedded systems have functions that are implemented, controlled and managed by dedicated software.

There are many different safety critical systems in health care containing software for example defibrillators, dialysis machines, surgical devices, pacemakers and they have to provide such quality that they can be relied on because such applications and systems can endanger human lives. According to Hewett and Seker [15] other safety critical industries as well as medical device industries mandate certification for the code and its development process to assure quality of the system. The certification process requires much effort and the cost for developing a safety critical software system. Nilsen [29] states that it is much higher than the cost of developing non-critical

systems of the same size. Knight [22] discussed in 1990 the use of traditional methods (i.e. verification) to ensure quality in medical systems. Software was functioning in the used test cases and the use of formal verifications and mathematical proofs could be used to certify the absence of deadlocks and infinite loops, but the difficulty with medical systems is that they are very complex according to Knight. He means that because the medical systems are complex, embedded, and operates in real-time, the correctness determination in testing is extremely difficult. The output cannot be simply and quickly checked because there are no practical techniques for that available. Life testing is a way of testing and means that a system run in its operational environment or in a simulated similar environment, and that the failures of the system are observed over time. This type of testing requires too much elapsed time to reach a high level of confidence. Knight's conclusion is therefore that reliance on safety critical computer-based systems should only be undertaken with the greatest care.

The Therac-25 [26] is a well-known incident where a software fault led to that several people died or was seriously injured by massive radiation overdoses. The Therac-25 incident is an immediate example of a system harming patients. There are also those incidents that are less immediate harmful, inaccurate calculation of Downs-syndrome risk mentioned by Garde and Knaup [12] is an example. Faults that are not immediately harmful more often remain undiscovered and are therefore not reported in literature.

There are several mechanisms that can go wrong when it comes to software in medical devices. In the technical report Software Product Liability [1] several examples of software faults that resulted in recalls of medical equipment are presented, for example incorrect match of patient and data, faulty programming causing pacer telemetry errors, incorrect software design causing lockup of cardiac monitor, incorrect calculations, algorithm error

causing low blood pressure readings. Wallace and Kuhn [43, 44] have also studied software related failures of medical devices. They have had access to U.S. Food and Drug Administration database of medical devices failure that contains problems found in medical devices recalled by their manufactures either in final testing, installation or actual use in 1983 – 1997. The authors have paid a special interest to the medical devices containing software, for example insulin pumps, ultra sound imaging systems, pacemakers etc. One observation made is that the software related recalls have a higher percentage in later years (e.g. from 6% to 10-11%) and one possible explanation could be the rapid increase of software in medical devices. In a more recent article from 2006 Lee et al [24] state that the number of medical devices that have been recalled due to software and hardware problems is increasing to an alarming rate.

Wallace and Kuhn [43, 42] reduced the problem description for every recall to a symptom of failure and the most frequent found symptom in recalls were “function” (29 %), usually a single calculation or activity causing the failure. The authors also looked at the different types of faults and found that logic and calculations faults were most common and logic faults with its 43 % were the absolute most dominate type. The conclusion drawn by the authors from this is that it is important to have methods and techniques to prevent and detect logic and calculation faults for example inspections, traceability analysis, and different kinds of testing. Wallace and Kuhn also found that the nature of several faults indicates that known practices may not been used at all or may be been misused.

3.1.3 Medical devices regulated in law and regulations

There are lots of requirements to consider when medical devices are developed. The development of medical devices including the software is regulated by various standards, laws, regulations and

recommendations. In general, the standards describe software lifecycle process models that shall be implemented by the manufacturers and these processes are assumed to increase the organisation's capacity to develop high quality and safe medical device software. Regarding the software engineering techniques, the standards are quite vague what techniques to use during the different phases in the software development process and to what extent.

By the Commission of the European Communities [8] the term "medical device" is defined in the law about medical devices with the following definition: "Medical device means any instrument, apparatus, appliance, material or other article, whether used alone or in combination, including the software necessary for its proper application intended by the manufacturer to be used for human beings for the purpose of:

- diagnosis, prevention, monitoring, treatment or alleviation of disease,
- diagnosis, monitoring, treatment, alleviation of or compensation for an injury or handicap, investigation, replacement or modification of the anatomy or of a physiological process
- control of conception (birth control, solve infertility, miscarriage etc)"

It is important to notice that it is the manufacturer's purpose and the operation of the product that decides if the product is classified as a medical device, not the designer or the user.

The intention of this chapter is further to give a brief description of the laws and regulations in the U.S. compared with the laws and regulations in Europe including Sweden. The U.S. has a lot of companies that develop medical devices and is a trendsetting nation in the medical device area, which makes their laws and regulations interesting to compare with. Medical devices for the European market are regulated through European Union (EU) legislation. On the 14 of June 1998 the Council

Directive 93/42/EEC concerning medical devices (MDD[1]) [8] became mandatory. Every member state in the EU must adopt and publish laws, regulations and administrative provisions to implement the directive. There are some variations in national requirements, most of these concerns the need to notify the Competent Authorities, for example in Sweden the Medical Products Agency (MPA), when medical devices are placed on the market in their countries. There are different laws, regulations and a duplication of registration procedures for a medical device placed on the US market and the European market even if it is the same medical device. The demands on medical devices for the Swedish market are covered in the law (1993:584) [39] about medical devices and the regulations from the Medical Products Agency issued according to this law. The Swedish constitutions for medical devices are adapted to the common safety and security demands produced by EU. A medical device is considered suitable for the Swedish market, when used as intended, if it achieves the performance intended by the manufacturer and meets high standards for the protection of life, personal safety and health of patients and others. The manufacturers have to supply labels and instructions for use (users manual, display, voice etc) written in Swedish. This is irrespective of the device being used by a patient or by trained staff or if the device is used in a hospital or in a private clinic. Service manuals however can be in English.

So a company in Sweden who develops or manufactures medical devices for the Swedish, European and U.S. market has many different laws and regulation to adjust to, as seen in Figure 1.

[1] Medical Device Directive

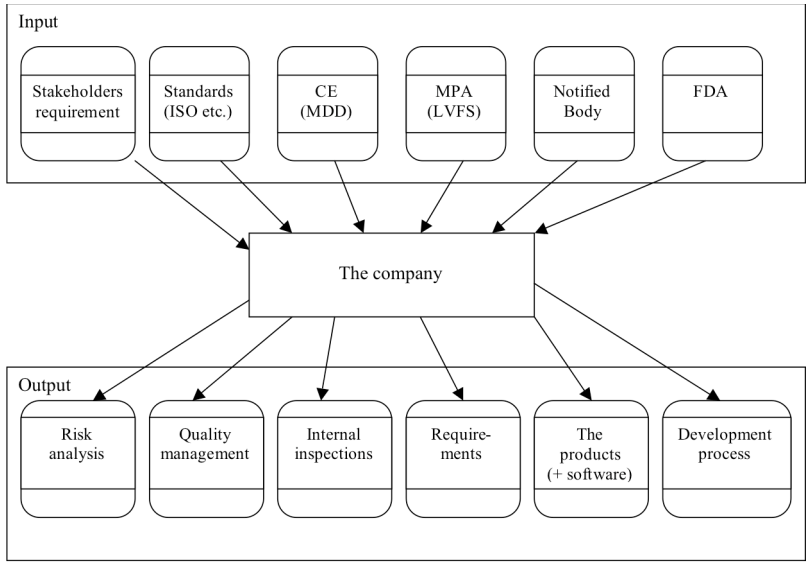


Figure 1. A company's input and output

All companies in the medical device domain have to study and comply with all the laws, regulations, directives and general advices for the market they are interested in and they have to produce several different documents and follow several different processes to fulfil the laws and regulations.

According the Medical Device Directive [8], medical devices in EU are divided into four classes Class I, IIa, IIb and III. Table 1 gives a short overview over the four classes and they are presented more in-depth further down. All medical devices on the European market are classified in one of these four classes based on the level of control necessary to assure safety and effectiveness. The medical devices with low risk potential are registered in Class I and the medical devices with the highest risk potential for example defibrillators and pacemakers is registered in Class III. All medical devices must be registered and approved by a national authority. In Sweden the national authority is the Medical Products Agency (MPA) and they are responsible for

regulation and surveillance of the development, manufacturing and sale of drugs and other medicinal products. The MPA is a government body under the Ministry of Health and Social Affairs (Läkemedelsverket). One of their main tasks is to register and have surveillance and control of medical devices and their manufactures in Sweden. The manufacturers themselves classify the medical device. For medical devices classified in Class I the manufacturers themselves assess if they fulfil laws and regulations. The manufacturing process however shall be controlled by a third part, often a Notified Body (NB). For medical devices in Class IIa a limited third part assessment is required were certain aspects are assessed. For the medical devices with high risk potential classified in Class IIb and Class III it is required a full third part assessment.

Table 1. Medical device classification

Class	**Risk**	**MPA**	**Self assessment**	**NB manu-facturing**	**Limited NB**	**NB**
Class I	Low risk potential	X	X	X		
Class IIa	Moderate risk potential	X			X	
Class IIb	High risk potential	X				X
Class III	Highest risk potential	X				X

The software that runs a medical device or affects the use of a device for example surveillance the medical device automatically belongs to the same class as the device. The classification is build up on the risks the human body can be exposed to due to the design, the use or the mode of manufacture of the medical device. It is assigned to the manufacturers, based on the

regulations [31] to establish in which class the medical device belongs and after that establish which procedure to apply to ensure that all the demands in the regulations are met. The manufacturer carries out the classification of the devices, possibly in cooperation with a Notified Body (third party assessment). The four different classes are:

- Class I: Registration at the MPA**.** The manufacturer of medical devices in Class I can themselves assess if the devices fulfil all the valid demands (i.e. fulfil laws and regulations) and a Notified Body shall check the manufacturing process. The manufacture, according to the regulations has to establish a technical file and an assurance that there is full compliance with the demands. This documentation must be preserved at least five years after the production of the device have been closed down. If the manufacture has a registered place of business in Sweden the manufacture has to register their medical devices at MPA. MPA will issue confirmation of registration in the form of a Certificate of Registration that allows the device/s to be placed on the market. The registration will convey an annual fee. Class I is comprised of the least dangerous devices and is the largest group.
- Class IIa: Limited third party assessment is required; Class IIa requires assessment by a Notified Body for certain aspects. This can be done in two ways depending on what the manufacture choose. The first way is that the manufacturer declares that the device is in conformity with the essential requirements. After that the Notified Body performs a type-examination of products and/or batches or certifies the quality assurance system for manufacturing and/or final inspection. The second way is that the Notified Body certifies that the manufacturer's has done their work, an application for registration of the

company and the devices are sent to the Medical Products Agency.

- Class IIb and Class III: Third party assessment required. Devices in Class IIb and Class III have a higher risk potential and shall always be assessed by a Notified Body. Even here the manufacture can choose way either the Notified Body certifies the manufacturer's entire quality assurance system or the Notified Body performs a type-examination of the device and/or batches or certifies the quality assurance system for manufacturing and/or final inspection. For all medical devices is it required that the assessment is to be documented in the form of technical files, declarations, explanations, type certificates etc. This documentation shall be written in one of the official languages of the EEA (European Economic Area) or in a language accepted by the Notified Body in question. Class III is set a side for the most critical devices for which explicit prior authorisation with regard to conformity is required for them to be placed on the market. The Medical Products Agency is able to combine injunctions and ban with fine if the law and regulations are violated. The manufacture that breaks the law or MPA's regulations gets an order to pay a fine or can be sentenced to at most one years' imprisonment.

In laws and standards are very few of the requirements direct requirements on software but a number of requirements are indirect applicable. In Sweden there is an ongoing work to change the law [39] and it is stated in the referral to the Council on Legislation [18] that an important change is the clarification that software is included in what can be a medical device and also detached software can be a medical device. It shall be added that for software regarded, as a medical device shall "be validated according to state of the art in the domain and taken into consideration the principles for development lifecycle, risk

management, validation and control" [18]. The changed law is planed to take legal force in March 2010.

In the US the regulatory body of the U.S. Food and Drug Administration (FDA) must approve medical devices. A medical device has to go through one or two evaluation processes, premarket notification (510(k)) or premarket approval (PMA) [40]. Requirements in U.S. law [41] is also vague for example, in the law software validation is addressed "confirmation by examination and provision of objective evidence that software specifications conform to user needs and intended uses, and that the particular requirements implemented through software can be consistently fulfilled".

The classification in the U.S. differs and they have three different classes, named Class I, Class II and Class III. FDA has established classification for approximately, 1,700 different types of devices and grouped them into 16 medical specialities referred to as panels. Based on the level of control necessary to assure safety and effectiveness the device is assigned to one of the three regulatory classes. The three FDA classes and their level of control are:

FDA Class I requires General Controls,

FDA Class II requires General controls and Special Controls

FDA Class III requires General Controls and Premarket Approval (PMA)

General controls are the baseline requirements of the Federal Food, Drug and Cosmetic Act [41] that apply to all medical devices. The company has to register their establishment and their device with FDA, comply with the labelling regulation, design and produce devices under good manufacturing practices (GMP) and submit a premarket notification (510(k)) to FDA [40]. The GMP regulation contains general quality assurance or quality system requirements in areas of concern to all manufacturers of complete devices. A premarket notification is marketing application to demonstrate that the medical device is a

safe and as effective or substantially equivalent to a legally marketed device that was or are currently on the U.S. market. The manufacturer cannot market the device unless the firm receives a marketing clearance letter from FDA. If the medical device is classified in Class III it must have a premarket approval (PMA) from the FDA and PMA is the process described by FDA to evaluate the safety and effectiveness of FDA Class III devices. FDA Class III is the most stringent regulatory category for medical devices and usually contains devices that support or sustain human life, are of substantial importance in preventing impairment of human health or which present a potential, unreasonable risk of illness or injury.

To short summarise, the medical device domain is a complex domain due to for example constant changes, many different types of persons are involved and there are multitude of medical standards and terminology. Software in medical devices and systems increases and are an important part of the medicial device domain that brings different issuses to address. Medical devices, the software included are regulated in many standards and laws and the companies have to adjust to all of this standards and laws depending on the market the devices and systems are market on.

3.2 Risk management

3.2.1 Risk management in the medical device domain

Many safety critical development projects as well as other development projects that contain large amount of software have certain risks that can pose threats to the development project, to the product or to the organisation. In this context three categories of risks are defined by Sommerville [38]:

- Project risks
- Product risks
- Business risks

Project risks are risks that affect the projects resources, planning and scheduling, product risks are risks that affect quality or performance of the developed software and finally business risks are risks that affects the organisation that are developing or obtaining the software. It is important to identify different risks to secure the quality of the product, for example identifying product risks such as the number of persistent faults in the product.

Risk can be classified into categories to give a better understanding of the nature of the risk as shown in Figure 2 inspired by Hall [13]. In Figure 2 the term "Management" refers to project and management process risk, "Technical" refers to product and technical process risks, "Project" to a major risk category including customer relationships, "Process" includes tools to produce a product and "Product" includes intermediate work products.

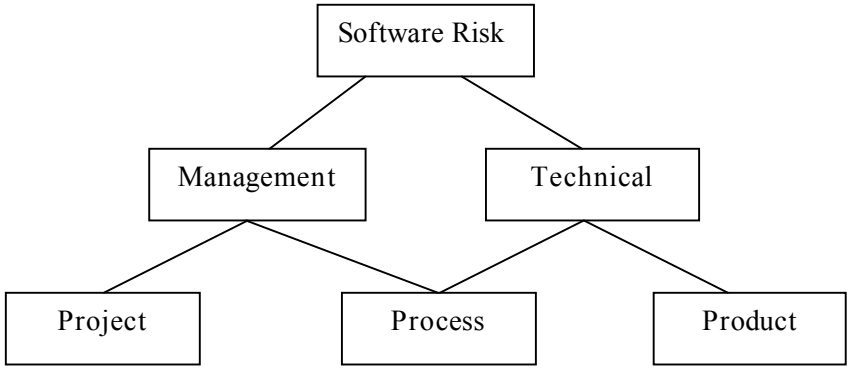


Figure 2. Software risk classification

The consequences of faults, error and mistakes that could occur in our daily life might not be so severe but when it comes to the health care domain the smallest mistake can led to

unforeseen consequences. The risks dealt with in health care are both, direct risks and indirect risks. A risk is according to Fairley [11] "the probability of incurring a loss or enduring a negative impact". It is crucial that medical devices will not malfunction in a way that they harm the patient and if unsafe medical devices that can malfunction are used it would pose direct risks to the patient. An indirect risk to the patient on the other hand, is if incorrect data from a medical device is used for diagnosis that in turn will lead to incorrect treatment. Clearly risks occur in the medical devices domain and have to be dealt with. The risks that developers of medical devices must address are to patients, users, third parties; for example service technicians, and the environment [30].

Risk management is today a general procedure for resolving risks, which means when it is applied to any instance, the possible consequences are all acceptable. An acceptable risk means that it is possible to live with the worst-case outcome [13]. Risk management is according to Doernemann [9] highly accepted in safety critical industries as for example in healthcare but more and more branches see the value or establishing risk management processes. Risk management is defined by Failey [11] as "an organised process for identifying and handling risk factors; includes initial identification and handling of risk factors as well as continuous risk management" and is often carried out in a number of steps: risk identification, risk analysis, risk planning, and risk monitoring as shown in Figure 3 inspired by Sommerville [38]

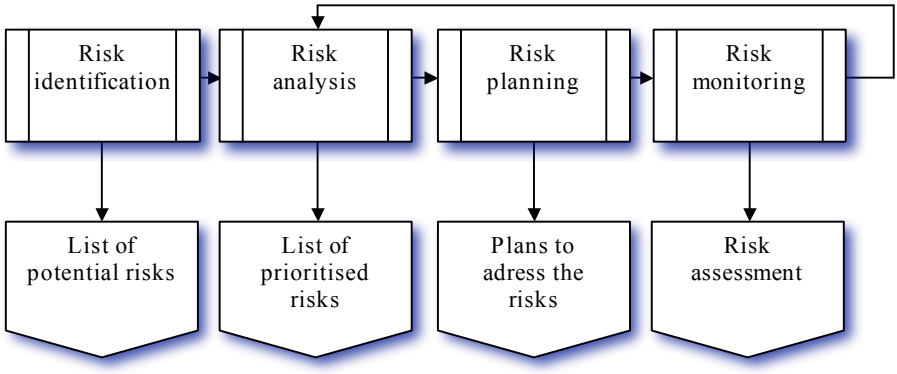


Figure. 3 Risk management process

During risk identification, brainstorming techniques and checklists are used to identify possible project, product and business risks. The risks are then prioritised with respect to their probability of actually occurring and their potential impact. The next step includes making plans to address the risk either to avoid them or to minimise the effects if they will occur and in the monitoring step the risks will constantly be assessed and the plans updated when more information about the risks becomes available.

According to Fairley [10], risk management is seldom applied as an explicit project management activity. The reason he gives is that there are very few guidelines available that offer a step-by-step approach to managing risks.

Rakitin [30], states that it lies on the medical device companies to show that their software is safe and efficient. He means that for the companies to meet these responsibilities it is required by the companies to have expertise in effective risk management practices, to be familiar with software safety and to be able to adopt risk management mind-set. According to Rakitin [30] the identification of safety critical software components and data should be part of the risk management plan. He also means that risk management is an activity that requires active involvement

of a multidisciplinary team of design engineers, clinicians, service personnel and quality and regulatory staff.

According to Boehm [3], risk management is not a “cookbook approach”. A great measure of human judgment is required to be able to handle all the complex people-oriented and technology-driven success factors in projects.

3.2.2 Risk management according to law

All medical device companies have to have risk management according to law; how strict and detailed it have to be depends on the product. A defibrillator has to have a much more strict risk management than for example an electrical wheelchair. ISO 14971 (www.iso.org) is a standard for medical devices where a majority of risk management terms is defined and a framework for an effective risk management process is given. Often, software is used to mitigate hazards caused by the other components, for example detect if a temperature is not in the correct range. According to ISO 14971 the companies must be able to show documented evidence that the software mitigations are effective and it also demands that the companies have risk management after the products are out on the market. Rakitin [30] means that the focus shall be put on how serious the consequences will be if a risk occurs and not so much on how likely it is for the risk to occur.

Software in medical devices is regulated by laws and regulations in the same way as medical devices in general. According to Leif, [25] there are different key elements the FDA ad ISO focus on. The three areas are:

- Software requirements
- Traceability though design, code and testing
- Formalised testing

Hazard and safety are the main focus fore all of the three areas mentioned above.

3.3 Challenges in the future

What will the future bring for software in medical devices and systems? Schrenker [34] discuss this and states that software has contributed to improved and more advanced health care up until now. To continue these improvements demands cooperation between manufactures, distributors and users, even to a greater extent than today. Schrenker regards that there is a need for focusing more on the user interface to reduce failures regardless of it is an embedded system or not. He also means that the software developers' reputation in the domain is not so good as it should be, but since the state of the practise has improved in the domain in recent years, it should be possible to improve the reputation.

The conference 2005 High-Confidence Medical Device Software and Systems (HCMDSS) [22] were held to discuss and to develop a roadmap for overcoming crucial issues and challenges regarding software in medical devices and systems. The participants identified six issues as critical for the future of high-confidence medical devices, for example software need to be developed with rigorous software development methods to ensure reliability and to protect the public health. The question is exactly how to accomplish that particularly because devices and systems are becoming increasingly complex and interconnected. Validation and certification processes need to be improved or changed because the complexity increases and more and more embedded systems are used which results in more stressed validation and certification processes. This trend then results in higher development costs for manufacturers and longer time to market. During the workshop, seven research directions were also agreed on. These research directions will be a help to make significant progress toward realising the outlined

challenges. “Requirements and metrics for certifiable assurance and safety” is one of the seven research directions. It means developing rigorous requirements for clinical and design purpose and metrics for certifiable assurance, in particular formalising requirements that will enable precise and transparent translation of natural-language clinical requirements to quantified engineering requirements. In testing this would mean developing frameworks for generating scenarios from clinical requirements. To succeed within the research directions it is necessary with collaborative effort involving academics and professionals working together with the support from governments.

4 Research methodology

This chapter gives an overview over the research methodology used in this thesis. A large part of the software research in this thesis is focused on creating an understanding of the way of working and the characteristics of software development in the medical device domain. The findings lay out the foundation for creating and evaluating new methods, processes and techniques.

The paradigm of behavioural-science seeks to find “what is true?” and the design-science paradigm seeks to create “what is effective? Hevner et al argue [14] that to ensure relevance and effectiveness of information system research both paradigms are needed. According to Silver [36] information systems research lies in the intersection of people, organisation and technology. The characteristics of software engineering objects of study are different to some extent from information system [33]. The study objects in software engineering are private corporations or public agencies that are developing software rather than using software. The study objects are also project oriented and the work is advanced engineering work conducted by highly educated persons than line oriented routine work.

The research in this thesis is software engineering research not information systems research. However the research of software engineering and software engineering in the medical device area lies also in the intersection of people, organisation and technology and both questions, “what is true?” and “what is effective?” are needed in the research.

4.1 Methodology

For the papers presented in this thesis, empirical research methods have been used. Empirical research is seeking to explore, describe and explain different phenomena through collecting and using evidence based on observations or experiences. The evidence is obtained through for example interviews, surveys or experimentation [37]. According to Seaman [35] most empirical software engineering studies combine qualitative and quantitative methods and data.

Research can according to Robson [32] have two main types of research design, fixed design or flexible design. Fixed designs, also called quantitative designs relying on quantitative data are either descriptive or experimental, and are highly pre-specified and prepared. Fixed design are often concerned with comparing two or more groups, and a theory is required in order to define what to search for [32]. Flexible designs, also called qualitative designs are concerned with studying objects in their natural setting and describe issues of the real world. The intention with the design is that it should progress based on the more knowledge the researcher gain during the study. Flexible designs are less pre-specified and rely more on qualitative data. In a qualitative study [35] where, for example, interviews are used it is important that the interview is flexible enough to allow for unforeseen information to be recorded and all information shall be regarded useful, even if the usefulness of a specific data is not known until long after it was collected.

A design cannot be both fixed and flexible at the same time but a design could have flexible phases followed by fixed phases; the other way around is very rare. Flexible designs can include the collection of quantitative data however, even if fixed designs can include qualitative data it rarely does [32].

The research in this thesis uses two types of research methods; surveys and experiments.

The purpose of surveys is defined by Wohlin et al [45] and the purpose is "to understand, describe, explain or explore the population". It is difficult to give a concise definition of survey research but a survey has often three typical central features according to Robson [32]:

1. Fixed, quantitative design is used.
2. From a relatively large number of subjects a small amount of data in a standardised form is collected.
3. A representative sample of individuals from known populations is selected.

These three central features capture a large part of surveys but there are surveys where considerable amounts of data are collected from each individual but the individual do not represent themselves but rather a company or organisation.

However even if surveys often are referred to as a fixed design, Robson [32] also argue for that surveys can be based on either flexible or fixed design depending on the degree of pre-specification. In typical fixed design the data collection is made by questionnaires with closed questions and in typical flexible design the data collection is made through interviews with open-end questions. Both fixed and flexible survey designs are used in this thesis.

The research in the second part of this thesis, the part about the state of practice in the medical device domain is based on the use of surveys in two different ways. Paper I is based on fixed design and carried out through a web-based questionnaire and

Paper II containing in-depth interviews is based on flexible design with open-ended (semi-structured) interview questions.

The fixed design with the use of a web-based questionnaire was chosen because it is an easy way to retrieve information from a large set of people in different countries. It allows anonymity and can provide large amount of information to a low cost in a short period of time. The design was typically fixed with the closed questions, where it is possible to know that the questions mean the same to the different respondents. The design in the study described in Paper II is flexible since the number of participants was limited and it allowed interviews with open-end questions where concepts and terms could be explained to the interviewees. The open-ended questions allowed unforeseen information to be recorded and the flexibility allowed questions to be added, removed or changed. Another reason for using flexible design is that the researchers gained more and more knowledge in the area, so the used design had to allow progress based on the more gained knowledge.

A commonly used technique for preparing qualitative data to be analysed quantitatively is coding, were value for quantitative variables extracted from qualitative data in order to do some quantitative or statistical analysis. This process was used in Paper II were interviews were used to collect the qualitative data and then same statistical analysis was made. The distinction between qualitative and quantitative data is not if it subjective or objective, it is how the information is represented [35]. Quantitative data is represented as numbers or other discrete categories and qualitative data is information expressed by words or pictures. Using qualitative methods increases the amount of information contained in the collected data since qualitative data is richer that quantitative data [35].

Experiments are used as research method for Part C of this thesis. Experiments are conducted when the researcher wants control over situation with systematic manipulation of the

behaviour of the studied phenomena [46]. Experiments [32] are of fixed design type and are focused studies with a few variables to handle. The experimentation is a research strategy that involves manipulation of one or more independent variables by the researcher, the measurements of the effects of manipulation on one or more other dependent variable and control over all other variables.

The research in Part C of this thesis is based on two different experiments. Paper III is based on an experiment were students acted as subjects. The use of students as subjects can be questioned. Höst et al [19] conclude in a comparative study of students and professionals in lead-time impact assessment that "there are only minor differences between the conception of students and professionals and there is no significant difference between the correctness of students and professionals". However it cannot be concluded with large validity that the students that participated in the experiment presented in paper III are representative of professional practitioners. So the result from this experiment should primarily serve as basis for continued experiments in the area. Another factor regarding the participants in controlled experiments is the incentives for participants in the experiment [20]. Höst et al [20] argue that the validity of a study is affected by the motivation of the participants and they introduce a way of trying to capture the motivation by looking at the experiment situation where the subjects are participants. In the experiment described in Paper III the intention was to take that into count and motivate the students as subjects, by having a seminar about risks and by designing the experiment to be representative for engineering work and linked to the project course the students attended at that time. The students being in their second year at the university are classified as "E1: Undergraduate student with less than 3 months recent industrial experience" according to Höst et

al [19]. Recent industrial experience means that it is experience received less that two years ago.

The second experiment is an experiment in real context also called quasi experiment. Quasi experiments are experiments when units are non-randomly assigned to experimental groups [20]. Kampenes et al [21] conclude that quasi-experimentation is useful in many settings in software engineering. Quasi experiments according to Basili [2] tend to involve qualitative analysis components and that quasi experiments easily can be done in vivo with experts and that this experiments easily deals with large projects. The subjects used in the quasi experiment described in Paper IV are three different categories of professional practitioners, software developers, medical device developers and physicians. The subjects were non-randomly selected.

4.2 Data collection methods

Data collection can be done with several different methods depending on the type of empirical study. In the survey presented in Paper I and in the quasi experiment presented in Paper IV questionnaires were used to collect data. The way the questionnaires were administrated is so called self-completion [38], which refers to that the respondents fill in the answers themselves.

The purpose behind the use of interviews in empirical studies is often to collect data about phenomena not suitable for quantitative measures [16]. Hove and Anda [16] state that it is important that the interviewees feel comfortable during the interview so that they are willing to share their experiences. This was taken in mind when the data was colleted though face-to-face interviews in the study described in Paper II.

Interviews can be classified in different types depending on how well structured they are. If the interviews are very well structured they are classified as fully-structured, over semi-

structured and then on the other hand there is unstructured interviews [32]. Semi-structured interviews combine specific questions, to get foreseen information and open-end questions to elicit unexpected information [16]. The interview performed in Paper II is a semi-structured interview with predetermined questions and open-ended questions. When the interview is semi-structured it is possible for the researcher to explain concepts and terms and it is also allows for changing the order of the questions [32]. In the experiment described in Paper III a special designed software tool was used to collect the data.

4.3 Validity

It is important to address validity even if the research study has been conducted with reliable and structured methods and techniques the results should always evaluated and questioned.

Validity can according to Yin [47] be classified in construct validity, internal validity, external validity and reliability.
Construct validity is affected by how well the collected operational measures represent the concept studied by the researcher. Internal validity is affected by factors that are outside the control of the researcher but affect the measures. External validity concerns the problem of how general findings are with respect to the subject population and beyond the immediate study. Finally, reliability also affects validity. How reliable a study is depends on how well the described procedures are followed and documented, so that the study can be repeated in the same way over again.

There are several possible threats to validity in studies. Threats to construct validity can be participants' bias. There is a risk that the participants misunderstand and interpret terms, concepts or questions differently. In the studies described in this thesis, the risk of misunderstanding have been reduced by only allowing responses on such a level of detail that subjectivity

from the participants and need for interpretation are minimised. During the interviews presented in Paper II the interviewer explained the terms and concepts during the interview with the aim to try to avoid any misunderstanding. Another participant's bias can be that the participant gives a too positive picture of the situation. In order to lower this risk in the survey study presented in Paper I, the participants were allowed to be anonymous. However here is a risk that the participants exaggerate the negative sides in pure frustration over different situations and this, as it has been done, must be taken in consideration when interpreting the results.

The threats to internal validity have been minimised by only analysing relations between factors and not draw any conclusions of causal direction in the studies.

External validity primarily relates to how general the results of the study are but another threat can be that the participants are not representative of the target population. In the studies in Paper I, II and IV professionals from the software and medical sectors have been used but in the study in Paper III it is students that have been used as participants in the experiment. This is a threat and this experiment should be repeated again with participants working as software engineers.

Concerning replication, controlled experiments in a laboratory can be performed as direct replications and external replications can be done where the replications is conducted in for example different environments and then compared to the original study. In a flexible design study however the context of the study always changing, which makes it impossible to recreate. There are particular threats to qualitative research in the areas of description, interpretation and theory. After an interview the researcher must provide a valid description. A way of assuring such validity is to record the interviews as was done in the interview study described in Paper II. The interpretation of the answers or discussions should not be affected by researcher's

bias. The studies presented in this thesis have always been analysed by more than one researcher with the intension to try to avoid this threat. Based on the findings the researchers have considered alternative interpretations or explanations for the results and searched for material in the study that does not fit in the theory of the study.

Conclusion validity [47] should also be taken in count. Conclusion validity is related to the possibility to draw correct conclusions about relations between the dependent and independent variables. A typical threat can be the use of wrong statistical tests. With this in mind the statistical tests were chosen for the studies with great care and for example in the controlled experiment presented in Paper IV the analysis was also done with non-parametric test.

A main concern in the work with the four papers in this thesis to identify and reduce all the validity threats as much as possible.

The validity is as mentioned also affected of reliability. It is important that procedures are followed well and carefully documented to get a reliable study that can be repeated over again. Procedures and changes in the studies presented in this thesis have been monitored and documented over time.

A more detailed presentation of research methods, data analysis and validity threats for each paper are presented in respective paper in Part B and C of this thesis.

5 Contributions

The main contribution of the thesis is summarised in this section. The individual studies have provided results in two areas; the understanding of the state of practice in the medical device domain, and risk management. Two areas are also

interconnected since risk management is an important process in the quality management process of the medical device domain.

Companies have been developing medical devices and systems for many years but the tendency is that these products contain more and more software. The research of software engineering in the medical device domain is relatively new and not so comprehensive. Therefore there is a need for increased understanding of the current state of practise of software development in the medical device domain. It is important to understand the companies' goals and concerns, what development processes and quality assurance techniques they use and how they address law and standards in this area.

The papers in Part B of this thesis present the current state of practice for software development in the medical device domain. Software has a high impact on product quality. The results in Paper I indicate that safety critical functions in medical devices often are realised by software. The research in this thesis has identified the need for high quality software in this domain and also the needs of new techniques, methods and processes to further improve software quality in the medical device domain. The findings indicate that software development processes in different activities seem to be less formal than expected and that software related standards have low impact in the current state of practise when developing software. The intention is that the results should be used as a guide to find more adapted processes, methods and techniques for software development in the medical devices domain. The results have been used to derive a set of requirements on new processes, methods and techniques in the area. The derived requirements are presented in detail in Paper II and can serve as guidance to researchers aiming to improve and develop processes, methods and techniques in the medical devices domain.

The results in Part B indicate that risk analysis is performed on a regular basis by the majority of the companies but the risk

analysis seems to focus more on the overall product and is less frequently applied on the software in detail. The research shows that established and systematic techniques to analyse risks of the development and the products is not as frequently used as could be expected. No indication was given in the research of why the established and systematic techniques of risk analysis are not frequently used; this will be a question for further research.

A number of decisions are taken during software project risk management. In Paper III it is concluded that different participants have different opinions about how serious risks are concerning faults remaining after testing are. Probably it is possible to generalise this and conclude that people, acting in software engineering processes are more or less risk seeking, something that is important to know in a risk management process. The measurements of project participants risk tendency could be useful in the risk management process and the research shows that there are methods (e.g. the Trade-off method [42]) for assessing the level of risk tendency available.

Risk identification is an important part of the risk management process. It is desirable to obtain an as complete set of identified risks as possible. In the controlled experiment presented in Paper IV is it concluded that multiple roles; software developers, medical device developers and physicians and thereby different experiences will affect the risk identification process. Involving multiple roles will result in a more complete set of identified risks than if only one role is included in the risk identification process. It can be concluded that for systems in the medical device domain it is important to include participants from different professional groups in the risk identification process. The users of the systems must also be included it is not sufficient to only include the developing organisation in the process.

6 Future works

This section describes how the research can be continued in the future. Good software development processes, methods and techniques can lead to high quality products and the overall goal for future research is aiming to further improve software quality in the medical devise domain.

The main goal of the research presented in Part B of this thesis is to create general knowledge and understanding of software development in the medical device domain and also to form a source for further research. The research has lead to that a number of software development issues have been identified and future research will be to investigate these issues in-depth. One of these issues is, the increasing software complexity that affects the validation and certifications processes and make them more time and effort consuming. Another issue are that the development processes and risk management processes are not adapted for software development. The focus will also be to address identified issues by developing and improving more adapted processes, techniques and methods for software development to the companies in the medical device domain. In Paper II a set of requirements on new processes, methods and techniques are presented. These requirements are intended to be used as a type of checklist, question list, or guidance in for future research and development.

The next phase of this research in the risk management area should be in cooperation with a company in the medical device domain. Since research indicates that the risk analysis in the medical device domain seems to focus more on the overall product than on the software in detail a proper study in the environments of the companies should be done. The results from such a study should then be used to support and improve the company's risk analysis process with added focus on software. The first step in this phase could be to do the experiment

described in Paper IV again but this time combined with a checklist. Another angle could be to involve a risk expert or a group of risk experts in the experiment.

Another possible continuation for the research in the risk management area is to use the Trade-off method described in Paper III and develop a tool for support of the decision-making process in the risk management area and evaluate the tool. A possibility would also be to combine the experiment in paper III with the experiment in paper IV to investigate if participants that are classified as risk averse find more risks than those who are risk averse or vice versa.

For research in the medical device domain regarding development or improvement of new software processes, methods and techniques there are some concerns to consider as researcher. The processes, methods and techniques must:

- be able to fulfil several different laws and standards. Medical devices that are marketed in several countries have to follow the different laws in these different countries. Medical device companies are also often certified according to several different standards and have to adjust to them.
- contain documentation that is easy to do and support. All quality assurance activities in a medical device company must be documented according to law.
- focus on safety and effectiveness. Safety and efficiency are key areas for the medical device domain.
- include design and process control. According to law the medical device companies must have design control that are an interrelated set of practise and procedures that are incorporated into the design and development process. According to standards the companies must monitor, measure and analyse processes.

- secure customers requirements.
  According to standards, it is the medical devices companies' top management that have the responsibility to secure the customers′ requirements.
- support traceability.
  According to law, all quality activities must be able to trace during the whole development process.
- be easy to inspect by third part.
  Most of the medical device companies have their quality systems inspected several times a year by third part.

These concerns mentioned above shall as much as possible affect the future research on development or improvement of new software processes, methods and techniques.

References

[1] Armour, J., Humphrey, W. S. “Software product liability”, Technical report, CMU/SEI-93-TR-1, ESC-TR-93-190, 1993.

[2] Basili, V. R. “The Role of Experimentation in Software Engineering: Past, Current and Future”, In proceedings of the 18th International Conference on Software Engineering, 1996, pp 442-449.

[3] Boehm, B. W. “Software Risk Management: Principles and Practices”, *IEEE Software,* vol. 8, no.1, 1991, pp 32-41.

[4] Bovee, M. W., Paul, D. L., Nelson, K. M. “A framework for assessing the use of third-party software quality assurance standards to meet FDA medical device software process control guidelines”, *IEEE Transactions on engineering management,* vol.48, no 4, 2001, pp. 465-478.

[5] Bowen, J., Stavridou, V. ”Safety-critical systems, formal methods and standards”, *Software Engineering Journal,* vol. 8, no. 4, 1993, pp. 189-209.

[6] Brooks, F. P., Jr., “No Silver Bullet Essence and Accidents of Software Engineering”, *IEEE Computer,* vol. 20, nr. 4, 1987, pp. 10-19.

[7] Charette, R. N. “Dying for data”*, IEEE Spectrum,* vol. 43, 2006, pp. 22-27.

[8] Commission of the European Communities, Council Directive 93/42/EEC of 14 June 1993 concerning medical devices.

[9] Doernemann, H. “Tool-based risk management made practical”, In proceedings of the IEEE Joint International Conference on Requirements Engineering (RE´02), 2002, p. 192.

[10] Fairley, R. “Risk Management for Software Projects”, *IEEE Software,* vol. 11, no. 3, 1994, pp 57-67.

[11] Fairley, R. E. “Software Risk Management”. *IEEE Software,* May/June 2005, p. 101.

[12] Garde, S., Knaup, P. "Requirements engineering in health care: the example of chemotherapy planning in paediatric oncology", *Requirements Engineering,* April 2006, pp. 265-27

[13] Hall, E. M. *Managing Risk Methods for Software Systems Development,* Addison-Wesley, 2003.

[14] Hevner, A. R., March, S. T., Park, J., Ram, S. "Design Science in Information Systems Research", *MIS Quarterly,* vol. 28, no. 1, 2004, pp 75-105.

[15] Hewett, R., Seker, R. "A risk assessment model of embedded software systems", In proceedings of the 29th Annual IEEE/NASA Software Engineering Workshop, 2005

[16] Hove, S. E., Anda, B. "Experiences from Conducting Semi-Structured Interviews in Empirical Software Engineering Research", In proceedings of the 11th IEEE International Software Metrics Symposium, 2005 (METRICS 2005), pp. 23-33.

[17] http://ec.europa.eu/information_society/activities/health/whatis_ehealth/index_en.htm, 2009-01-27.

[18] http://www.regeringen.se/sb/d/108/a/116966, 2009-02-23.

[19] Höst, M., Regnell, B., Wohlin, C. "Using Students as Subjects-A Comparative Study of Students ad Professionals in Lead Time Impact Assessment", *Empirical Software Engineering*, vol. 5 Issue 3, 2000, pp. 201- 214.

[20] Höst, M., Wohlin, C., Thelin, T. "Experimental Context Classification: Incentives and Experiences of Subjects", In proceedings of the 27th International Conference on Software Engineering, 2005, pp 470-478.

[21] Kampenes, V. B., Dybå, T., Hannay, J. E., Sjöberg, D. I. K. "A systematic review of quasi-experiment in software engineering, *Information and Software Technolog",* vol. 51, 2009, pp. 71-82

[22] Knight, J. C. "Issues of software reliability in medical systems", 3rd Annual IEEE Symposium on Compute-Based Medical Systems, 1990, pp. 153-160.

[23] Knight, J. C. "Safety Critical systems: Challenge and Directions", In proceedings of the 24th International Conference on Software Engineering, 2002. ICSE 2002, pp. 547-550.

[24] Lee, I. et al, "High-Confidence medical device software and systems", *IEEE Computer,* April 2006, pp 33-38.

[25] Leif, S. B., Leif, R. C. "Producing quality software according to medical regulations for devices", In proceedings of the 5th Annual IEEE Symposium on Compute-Based Medical Systems, *IEEE Computer Soc. Press,* 1992, pp 265-270.

[26] Leveson, N., Turner, C. "An investigation of the Therac-25 accidents", *IEEE Computer,* vol. 26, 1993, pp 18-41.

[27] Linberg, K. R. "Defining the role of software quality assurance in a medical device company", In proceeding of 6th Annual IEEE Symposium on Compute-Based Medical Systems, 1993, pp. 278-283.

[28] McCaffery, F., McFall, D., Donnelly, P., Wilkie F. G., Steritt, R. "A software process improvement lifecycle framework for the medical device industry", In proceeding of 12th IEEE International Conference and Workshops of the Engineering of Computer-Based Systems (ECBS´05), pp. 273-280.

[29] Nilsen, K. "Stringent certification requirements for safety-critical software", *Embedded Control Europ,* pp. 18-21.

[30] Rakitin, S. R. "Coping with Defective Software in Medical Devices", *IEEE Computer,* vol. 39, no. 4, 2006, pp. 40-45

[31] Regulations from Medical Products Agency, The Regulation LVFS 2003

[32] Robson, C. *Real world research*, second edition, Blackwell Publishers Ltd, Oxford, 2002.

[33] Runeson, P., Höst, M. "Guidelines for conducting and reporting case study research in software engineering", *Empirical*

Software Engineering, Springer, 2008, doi: 10.1007/s10664-008-9102-8.
[34] Schrenker, R. A. "Software engineering for future healthcare and clinical systems", *IEEE Computer,* vol. 39, no. 4, 2006, pp 26-32.

[35] Seaman, C. B. " Qualitative Methods in Empirical Studies of Software Engineering", *IEEE Transactions on Software Engineering*, vol. 25, no. 4, 1999, pp 557-572

[36] Silver, M. S., Markus, M. L., Beath, C. M. "The Information Technology Interaction Model: A Foundation of the MBA Core Course", *MIS Quarterly,* vol. 19, no. 3, 1995, pp. 361-390.

[37] Sjöberg, D. I. K., Dybå, T., Jörgensen, M. "The Future of Empirical Methods in Software Engineering Research", *IEEE Future of Software Engineering (FOSE´07),* 2007, pp 358-378.

[38] Sommerville, I. *Software Engineering*, 8:th edition, Addison Wesley, 2004.

[39] Swedish Code of Statutes (Svensk Författningssamling, SFS), The Act (1993:584) Medical Devices.

[40] U.S. Food and Drug Administration, 1995. Premarket Notification [510 (k)], Regulatory Requirements for Medical Devices, HHS Publication, FDA 95-4158.

[41] U.S. Food and Drug Administration, Federal Food, Drug and Cosmetic Act section 201(h).

[42] Wakker, P., Deneffe, D. "Eliciting von Neumann-Morgenstern Utilities when Probabilities are Distorted or Unknown, *Management Science vol. 42, no.8,* 1996, pp. 1131-1150.

[43] Wallace, D. R., Kuhn, R. "Lessons from 342 medical devices failures", In proceedings of the 4th IEEE International symposium of high-assurance systems engineering, 1999, pp 123-131.

[44] Wallace, D. R., Kuhn, R. "Failure modes in medical device software: an analysis of 15 years of recall data". *International Journal of Reliability Quality and Safety,* vol. 8, no 4, 2001

[45] Wohlin, C., Höst, M., Henningsson, K. “Empirical Research Methods in Software Engineering”, In Lecture Notes in Computer Science: Empirical Methods and Studies in Software Engineering: Experiences from ESERNET, edited by A.I. Wang and R. Conradi, Springer Verlag, 2003.

[46] Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., Wesslén, A. *Experimentation in Software Engineering: An Introduction,* Kluwer Academic Publishers, 1999.

[47] Yin, R. K. *Case study research design and methods,* 3rd edition, Sage, Thousand Oaks, Californian, 2003.

Part B

State of Practise in Software Development in Medical Devices

Paper I

State of the Practice in Software Development for Medical Device Production

Christian Denger, Raimund L. Feldmann, Martin Höst, Christin Lindholm, Forrest Shull

IESE-Report No. 071.07/E, Fraunhofer Institute for Experimental Software Engineering, February 2007.

Abstract

Today, many medical devices could not fulfil their intended use without the software embedded within them, which implements a variety of functions and features. Surveys of trends in the medical device industry indicate that software is one of the most decisive factors for producing innovative products with new capabilities, and predict that the importance of software will only further increase in the future. The increasing importance of software is reflected by a number of standards, which have been developed to codify the types of good engineering practices that are necessary to minimize the safety risks of such software. This report describes the results of an international survey that was performed in 2006 to understand the current practice for software development in the medical device domain. Our objective was to understand the current goals and concerns of companies in this market and how they choose practices to address the issues that they see. We were interested in understanding the extent to which the standards that have been developed have been recognized and instantiated by industry.

Furthermore, we were interested in eliciting the most important challenges with respect to developing embedded software for medical purposes. These results can be used by researchers and practitioners to get an overview of the level of usage of various processes and tools in the domain as well as to understand useful targets for developing new techniques and methods aimed at further improving software quality in the medical sector. More than 90 companies from Europe and the USA participated in the survey, ranging in size from SMEs with less than 20 employees to global players with several thousand developers. From these 57 companies could be used for a detailed analysis. To our knowledge this is the first survey of this size performed in the medical device domain that explicitly focused on the topic software engineering. The results indicate that software is an integral part of medical devices realizing safety critical functions. In most companies, software is mainly developed by non-computer scientists. The requirements engineering step and the architecture phase are perceived as the most challenging ones in software development. Mainly informal languages are applied in different work products and tools are rarely used to support the different development activities.

1 Introduction

1.1 The medical device domain

The medical device domain today is a strong and growing market [BDI05, Arte05]. As many western countries are experiencing increasingly elderly populations, achieving efficient health care systems is becoming recognized as one of the most important future challenges in many countries. An important component of such health care systems are the need for affordable but also innovative medical services. Consequently, many experts see the medical device market as strongly increasing in the near future. In Germany, for example, medical device producers increased their revenue by 9% to 14.8 billion Euro [BDI05] in the year 2004/2005. However, to meet these needs, medical devices also need to produce new capabilities, leading to extremely short innovation cycles in this market, especially in the context of electrical medical equipment. Studies show that companies make 50% of their revenues with electronic products that are less than 3 years old [BMB05].

In order to respond to this pressure for high innovation and to fulfil the increasing demands of new features and functionalities, software has become an integral part of many medical devices. Today, many medical devices could not fulfil their intended use without the software embedded within them, which implements a variety of functions and features. Surveys of trends in the medical device industry (e.g., [Adva04], [ITEA05], [BDI05]) indicate that software is of the most decisive factors for producing innovative products with new capabilities, and predict that the importance of software will only further increase in the future [BMB05], [BDI05]. Studies also predict that the research and development (R&D) investment in software in this market will increase to 33% of the overall budget by 2015 [ITEA05].

Due to the increasing importance of software for the overall quality of the product, the global harmonization task force defines that embedded software itself qualifies as a medical device (e.g., in [IEC06]):

"A medical device is any instrument, apparatus, implement, machine, appliance, implant, in vitro reagent or calibrator, software, material or other similar or related article, intended by the manufacturer to be used, alone or in combination, for human beings for one or more of the specific purpose(s) of

- diagnosis, prevention, monitoring, treatment or alleviation of disease,
- diagnosis, monitoring, treatment, alleviation of or compensation for an injury,
- investigation, replacement, modification, or support of the anatomy or of a physiological process,
- supporting or sustaining life,
- control of conception,
- disinfection of medical devices
- providing information for medical purposes by means of in vitro examination of specimens derived from the human body,

and which does not achieve its primary intended action in or on the human body by pharmacological, immunological or metabolic means, but which maybe assisted in its function by such means."

In this context, both the embedded software in medical devices and stand-alone software products that fulfils medical needs must adhere to the same regulations and restrictions as any other medical device. In other words, software and the software development process must ensure (see also [HCM06]):

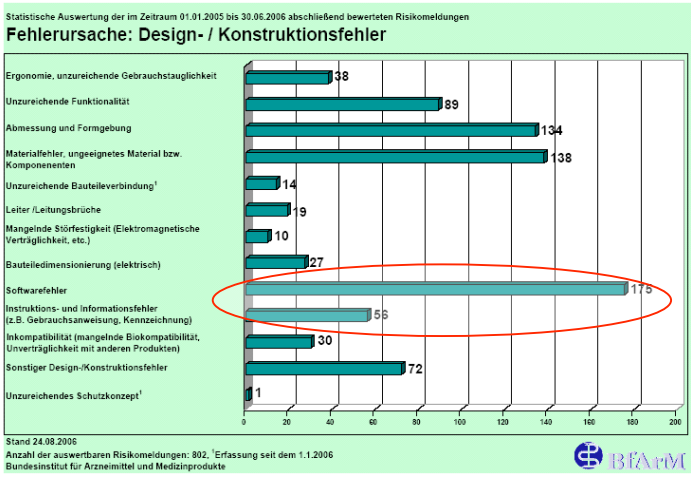
- High reliability of the device

- Safety of the device (no harm to patients, users, physicians, or the environment)
- Zero-Defect products (i.e., software is not the cause of any harm)
- Ease of use by different user groups
- Certifiable product and development processes

1.2 Motivation for this survey

As the role of software in the medical device domain increases in importance, so do the failures due to software defects. An analysis of medical device recalls by the FDA in 1996 [WK01] found that software was increasingly responsible for product recalls: In 1996, 10% of product recalls were caused by software-related issues. This was up from 6% in the years1983 – 1991. The German institute for pharmaceutical and medical products (see BfArM, Bundesinstitut für Arzneimittel und Medizinprodukte) continuously collects and analyzes notes of medical device companies according risks related to their products. In Figure 1 the causes of medical device risks are listed. Software is the top cause for risks related to construction and deign defected of medical device products. The analysis from June 2006 shows that 21% of the risks are caused by software (see also http://www.bfarm.de). This is an increasing trend as in November 2005 an earlier version of the analysis showed that software was with17% only the second most cause for risks related to construction and design defects. Software defects can stem from various causes, including (to name just a few):

- Incomplete / incorrect requirements
- Lack of formal requirements descriptions
- Missing traceability from requirements to design to code to test to user documentation
- Insufficient or wrongly-focused validation and verification activities



Software is the most common source of construction and design defects of medical devices

Figure 1: Failures of medical devices due to design errors.

To overcome such issues, the development of software is regulated by various standards, laws and recommendations (e.g., IEC 62304, IEC 60601-1-4, FDA-Software Validation principles, FDA-Premarket regulations for medical devices including software, FDA-Quality System; see [CDRH02], [ISO00], [IEC00] for details). In general, these standards describe software life-cycle models that should be implemented by manufacturers. The overall objective is the definition of general process steps and development work-products. Adhering to the regulations and following the specified processes increases an organization's ability to produce high quality and safe medical device software. However, in many cases the standards are quite vague regarding the concrete software engineering techniques that should be used in different development steps. Thus, there is in practice a high degree of freedom in instantiating the processes.

Given this context, and the general lack of empirical knowledge about the state of the practice regarding medical device software development, we designed an international survey. Our objective was to understand the current goals and concerns of companies

in this market and how they choose practices to address the issues that they see. We were interested in understanding the extent to which the standards that have been developed have been recognized and instantiated by industry. Furthermore, we were interested in eliciting the most important challenges with respect to developing embedded software for medical purposes. These results can be used by researchers and practitioners to get an overview of the level of usage of various processes and tools in the domain as well as to understand useful targets for developing new techniques and methods aimed at further improving software quality in the medical sector.

1.3 Outline of this report

The remainder of this report is structured as follows: Section 2 details the objectives of our survey. Section 3 describes our research methodology. Section 4 describes the detailed results, including descriptive statistics and correlation analyses. Section 5 discusses the survey findings and Section 6 concludes the paper with future work that can build upon these findings.

2 Objectives of the Survey

This survey was designed by software engineering researchers from three institutions in the United States and Europe (the Fraunhofer Institute for Experimental Software Engineering in Germany; the Fraunhofer Center in Maryland, USA; and Lund University, Sweden). The main objective of the survey was to characterize the state of the practice of software development in the specific context of *medical devices and medical information systems*, in order to understand how practices in this field were similar to or different from software practices in other domains. Since the medical device domain has specific quality goals and constraints we expected that there would therefore be a commensurate difference in the practices and processes that

were selected for building software under these conditions. By including organizations from both the US and Europe in this survey we hoped to understand how software development processes also differed in response to different regulatory and business environments.

In all cases, our overall objective was to *characterize* the state of the practice in this context. The survey collected no information that could be used to directly evaluate the effectiveness of the practices that are being applied, nor did it collect information that would compare respondents in this survey to those in another domain. We decomposed this high-level objective into three more specific research questions.

Research question 1: How can the medical device area in general be characterized with respect to *quality needs* and *development team organization*?

We were interested in characterizing the domain itself, its quality goals and constraints, independent of the software field. In order to truly understand the constraints on software it was necessary to understand in a larger sense the business and regulatory issues that companies found themselves operating under. (For example: How large are organizations as a whole; what types of devices does software form a part of; what type of role does software play in the device taken as a whole; what type of quality goals do the end-product devices have to satisfy?)

Said more precisely, our specific goals in this area were to:

- (Goal 1) **Analyze** the characteristics of the medical device domain **with respect to** the most important quality needs of devices and additional constraints for software development.
- (Goal 2) **Analyze** the organizational structure of companies **with respect to** the types of products created and the development constraints for software development

Research question 2: How is software engineering characterized during medical device production?

We were interested in understanding whether there were common approaches to software development in the medical device domain. For this reason the survey contained a number of questions that aimed at eliciting the way software teams are composed; the background of people on those teams; and the types of processes and practices applied by team members for a number of specific tasks.

Said more precisely, our specific goals in this area were to:

- (Goal 3) **Analyze** software development environments as well as software development and quality assurance processes **with respect to** the state-of-the-practice in this domain.
- (Goal 4) **Analyze** the interrelationships among software engineering processes and domain characteristics **with respect to** the state-of-the-practice in this domain.

Research question 3: What are the most recent challenges with respect to software engineering in medical device production?

In addition to understanding the state of the practice for software development, we also needed to gain an understanding of where those practices were working well and where the largest challenges still remained. For this reason, the survey contained another subset of questions that asked respondents to rank a number of potential issues that could occur at various stages of software development.

Said more precisely, our specific goals in this area were to:

- (Goal 5) **Analyze** the challenges in the domain **with respect to** patterns that can be found for organizations (or subsets of organizations) in this domain.

The specific questions that composed the survey were created through application of the Goal Question Metric (GQM) paradigm ([BCR94], [SB99]). This method resolves the goals

listed above into specific questions that address those goals. For each question a metric must be defined that is able to provide the necessary information and is feasible to collect.

3 Survey Methodology

3.1 Sample and target group

The target population for the study consists of organizations developing software for medical devices and medical information systems. This is a large population and it is impossible to carry out a study with the complete population where every organization is included. Therefore a sample of the population has to be chosen.

The sample was chosen based on the contacts that the authors had and were able to obtain in the initial steps of the survey. The survey questionnaire was available via a website, and the participants were invited by email. This means that a rather large set of potential participants could be invited. The contact information of potential participants was either available before the survey or obtained by consulting colleagues or attending conferences in the area. In addition, an invitation to participate was sent to all participants in the address database of large industry associations. (Since we were not allowed to access these databases directly, we do not know how many invitations were sent out in total.) As a response to the invitations 349 respondents looked at the starting page of the questionnaire. From these 113 started the questionnaire and were considered for analysis.

Answers, both valid and non-valid, were received from these 113 participants. Since a broad set of potential participants was invited a check was made for every answering participant whether they came from the domain of medical systems and whether the company was developing software. This resulted in 109 responses left for conducting the analysis.

Since it was possible to skip questions when answering the survey, some respondents did not provide answers to all survey questions. Thus we needed to decide whether there was a minimum threshold, below which too little information had been provided to be usefully included in the analysis. The survey ends with a set of characterization questions about the respondent's organization and the developers working there. Only the answers of those respondents who have answered at least one of these characterization questions are included in the survey. We believe that subjects, who reviewed all of the questions, even if they did not provide answers to all of them, are likely to provide the most valid data. In a second filtering step, subjects who have not answered any other question then the last characterization questions were removed. This left us with 57 valid responses, which were included in the analysis. In the following we call this set of companies "core data set" or "core subjects". The answers are from a variety of many different types of companies (as described later). Geographically, most of the respondents came from Germany (38), the USA (8), and Sweden (5).

3.2 Conducting the survey

The survey was carried out through a web-based questionnaire. Potential participants were given a link to the URL and asked to fill out the questionnaire. In order to motivate people to participate they were given the possibility to register their email in order to obtain the result of the survey. We also advertised that we would contribute $1 to the International Red Cross for every survey completed.

In order to increase the number of answers, reminders were sent to everyone who was invited, both respondents who had answered and respondents who had not answered the survey.

3.3 Analysis of data

One of the most important questions revolved around how companies who produce medical device obtain the software components (i.e., whether the software is developed in-house or obtained from outside). Of the 57 respondents, 20 companies use only software developed in-house in their products and 7 companies use only third party software in their products. 29 companies do both, develop their own software and integrate third-party software. 1 company stated that their products do not contain software. (Figure 2).

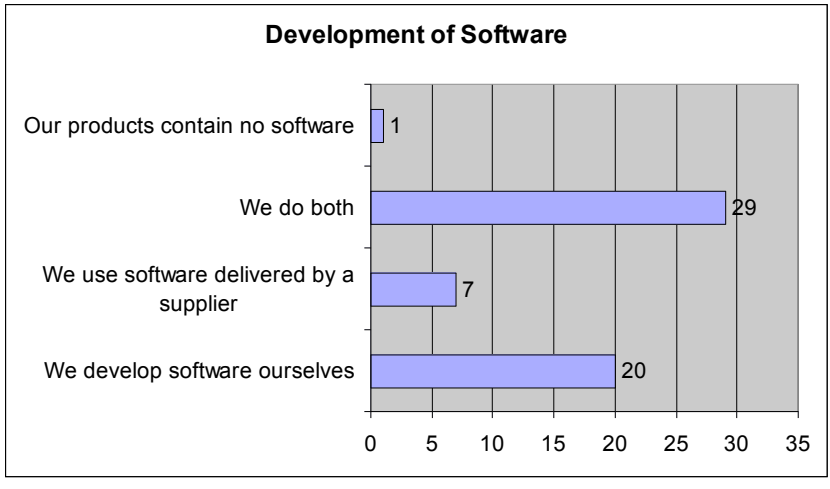


Figure 2: The source of the embedded software

Based on these data the analyses have been carried out with two data sets. First the analysis was carried out with the core data set consisting of the answers from the 57 people who had answered the characterization questions. These 57 data points have been used for the basic statistics (see Section 4.1). The way software is developed in the companies (see Figure 2) has implications for the amount of data that can be used in further analysis. We use 49 respondents for questions dealing with software development and 36 respondents for questions with respect to software from third parties.

After this, and only in case that we perceived this as relevant an extra analysis was carried out in order to evaluate the findings gathered from the analysis of the core data set. This analysis was carried out with an extended data set consisting of all available answers, i.e., the 113 subjects who started the questionnaire; irrespective of whether the respondent has completed the questionnaire or not. The extended data set was especially used for the analysis of correlations in the data sets (see Section 4.2). By doing this it was possible to verify whether the results of the core data set analyses held for the larger group. However, this extra analysis confirmed in all cases the results of the analysis of the core data set (i.e., it showed the same tendencies). Consequently, we do not describe the results of the extra analysis in this report.

The identified data set was analyzed with descriptive statistics in order to get a first understanding of the population and the answers from the subjects with respect to basic descriptive statistics. At this stage an initial analysis of clearly visible relationships between variables in the data set was also carried out.

As a next step relationships between variables were analyzed. Since most of the data is either on a nominal scale or on an ordinal scale with few possible values, statistical methods intended for this must be chosen. If the data indicates relationships one alternative statistical method to apply is the chi-2 test, e.g. [SC00]. Then the evaluation is treated as an experiment where the independent variable is represented by one of the variables and the dependent variable is represented by the other variable. For example, an objective may be to evaluate the relationship between how safety critical a company assumes that their product is and what software development process they use. Then the independent variable represents how safety critical the product is assumed to be and the dependent variable represent which process that is used.

4 Survey Results

This section presents a subset of the most important results from the survey. Section 4.1 describes basic statistics characterizing the state of practice regarding software engineering in medical device software development. Sections 4.2 to Section 4.4 contain analyses of the dependencies between different aspects of the various companies, the medical devices they produce, and the software development processes they use to do so. All statistics are taken from the core data set.

4.1 Basic descriptive statistics

The following sections provide an overview of the state of the practice regarding medical device software development.

4.1.1 Characterization of software development

Most of the companies (71%) participating in the survey are small and medium sized companies with 10 – 250 employees. The other 29% have more than 250 developers at the site participating in the survey. Within the companies the size of the software development departments or development teams varies. Figure 3 shows the distribution of the team-sizes. Almost 50% of the software development teams are smaller than 11 people. 18% of the companies have a development team that is larger than 50.

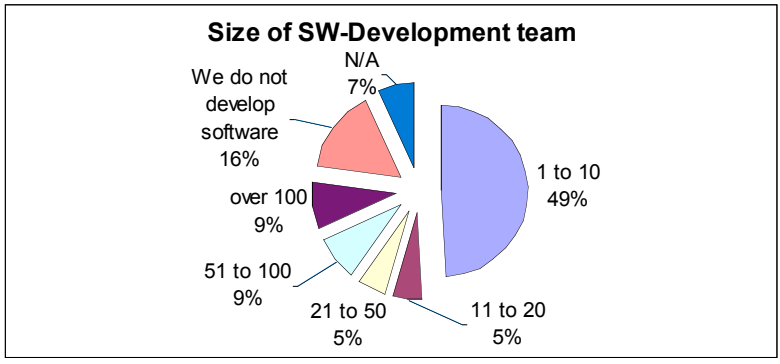


Figure 3: Size of the software development team

Figure 4 shows the educational background of the people developing the software of the medical device. The majority of the software developers have a background other than computer science. Almost two thirds (64%) of the companies answer that most of their software-developers stem from other disciplines, for example, electronics, medical sciences or electrical systems engineering. Consequently, in 36% of the cases the medical device software is mainly created by computer scientists.

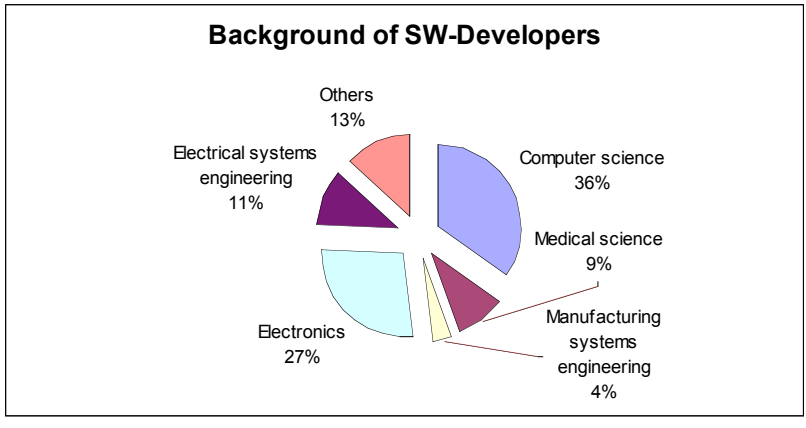


Figure 4: Educational Background of Software Developers

Figure 5 indicates the relevance of software in medical device products. It is evident that software is a decisive part in many medical devices: 98% of the companies rate software as either a

very important (84%) or important (14%) component of their devices / products. The figure also indicates that 76% of the respondents perceive their medical device (system and thus the included software) or their software-system (i.e., software as a stand alone product) as safety critical. In contrast, for only 16% of the companies perceive their product as non-safety-critical.

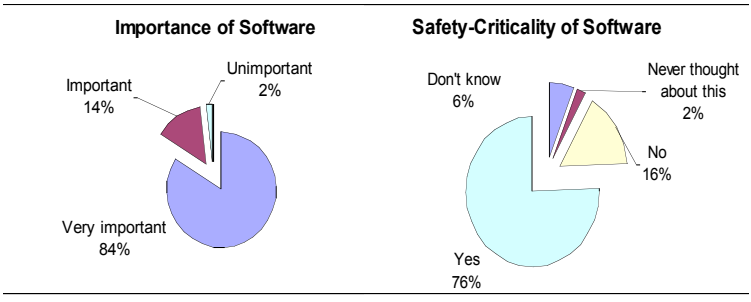


Figure 5: Importance and Safety Criticality of Software

Given the high importance of software for medical device products, the usage of software development standards and maturity models in this domain is also interesting.

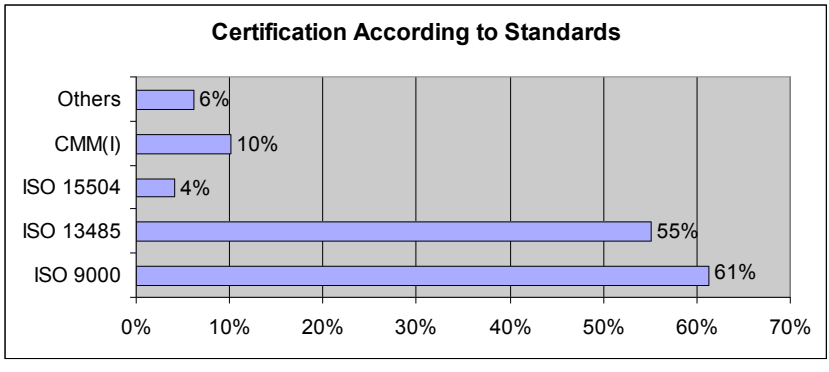


Figure 6: Software and Quality Management Related Standards

Figure 6 shows that quality management-related standards are of high importance in medical device production. 55% of the companies are certified according to the ISO 13485 standard for

medical devices quality management, and 61% are certified according to the ISO 9000 series, which provide requirements for general (domain-independent) quality management. Standards specific to software are of lower importance: 10% of the companies have a CMM(I) rating (most of them at level 2 or 3) and in 4% of the companies ISO15504 (SPICE) is followed, both of which aim at assessing software processes. Regarding plans for future certifications 12% aim for a CMM(I) assessment and 2% for a ISO15504 assessment.

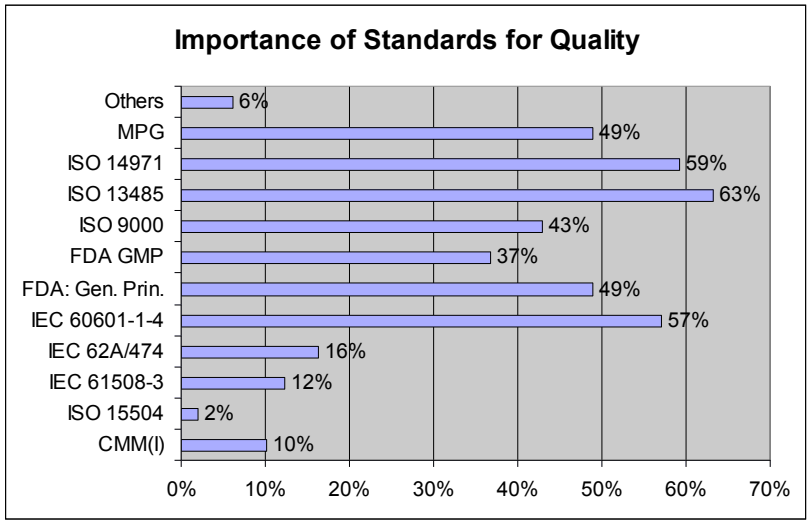


Figure 7: Standards and their perceived relevance to product quality

When asked about the perception of relevance, respondents rated medical device standards as more important for product quality than software-specific standards. As Figure 7 indicates, CMMI and ISO 15504 are perceived as less important for device quality then the more general process- and product-focused standards (ISO13485, ISO 14971). The most important software-related standard is the IEC 60601 1-4, which is applied by 57% of the companies.

To get a more concrete picture of how respondents define quality (of the resulting product and hence also of the integrated software) in their context, we asked respondents to rate whether various attributes were important to their products. Figure 8 shows the relative importance of different quality attributes of the medical devices and the integrated software. Realizing the specified functionality is the most important quality attribute (100% agreement) followed by usability (96% agreement) and reliability (90% agreement). Interestingly, the reusability, testability, and portability of the integrated software have quite low importance. Security also seems to be a minor concern, even though more and more medical devices are networked.

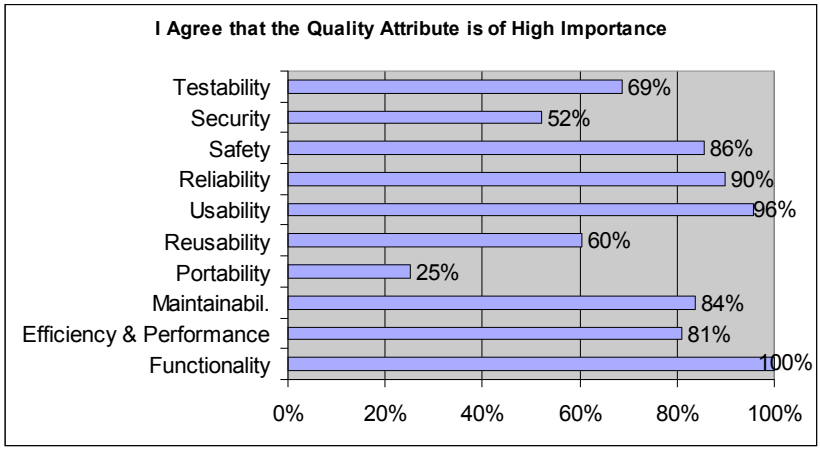


Figure 8: Importance of quality characteristics

The respondents were asked about the primary challenges to their system- and software-development. Figure 9 shows the summary of this rating. The percentage numbers indicate that a company either strongly agrees or agrees that a given challenge is faced in its development context.

Ensuring good usability of the system and consequently of the integrated software is perceived as the most challenging task (86% of the respondents agree or strongly agree to this). The

increasing complexity of systems and software, as well as ensuring good software maintainability, are perceived as the second and third most important challenges (84% and 80% of agreement). Another interesting aspect is that 74% of the respondents perceive the establishment of high quality under given development constraints as a major challenge.

We can also note that hardware-software interaction (38%) and guaranteeing efficient resource usage by the software (35%) were not perceived as challenges by the majority of the respondents. Thus, development activities close to hardware seem to be generally well handled.

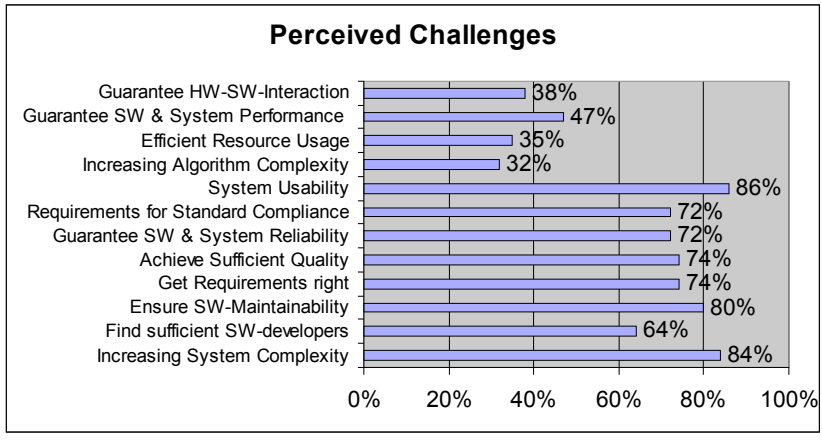


Figure 9: Challenges in system and software development

4.1.2 Constructive software development activities

This section deals with the constructive activities of the software development process, that is, those activities which produce intermediate work products or actually implement the software. The following terms were used to categorize the types of constructive activities, practices and techniques applied by companies:

- *Requirements engineering* referred to any activities in which the relevant functional and quality requirements of the software are elicited, analyzed, and specified.
- *Architecture and design activities* referred to any activities in which software requirements are used to generate software components, and an architecture that shows how they relate to one another to provide the technical solution.
- *Implementation* refers to any activities by which the architecture and design specification is transferred into working code.

To begin with, we asked respondents which of the above types of activities is perceived as the most challenging one when creating medical device software. Figure 10 shows that most of the issues regarding software quality stem from activities that involve planning the software, the functionality it should accomplish, and how it will accomplish it, that is, requirements activities (63%) and architecture and design activities (16%). Actually implementing the software code is perceived as the most challenging activity by only 10% of the respondents.

This indicates that investments in requirements engineering activities seem to be most promising to gain significant improvements in the software development process.

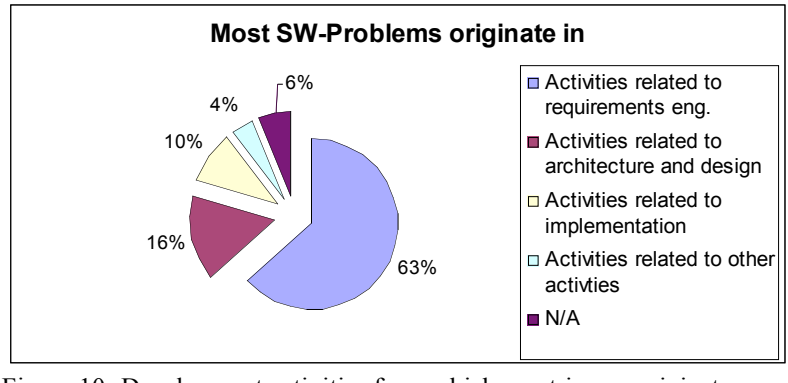


Figure 10: Development activities from which most issues originate

Looking in more detail at issues that cause problems for each type of activity, for requirements-related activities 86% of the companies perceive changing requirements as the main problem. Missing requirements (33%) and misinterpreting requirements (39%) are also perceived as important. Related to architecture and design, the main challenge is missing information in the diagrams (53%) and inconsistencies between the planned architecture and the software (39%). Missing opportunities to reuse software code in a systematic way (33%) and difficulties in maintaining the software code (29%) are perceived as the main issues during implementation.

Figure 11 shows that around 50% of the companies follow a defined process to perform the above mentioned activities on a regular basis (i.e., the companies always or frequently follow such a process). If those companies that follow defined processes in about half of their projects are counted, too, then 78% of the respondents had a defined process for implementation. The other activities have slightly lower values (71% for architecture, 69% for requirements).

In order to document the results of the various constructive activities, different notations and languages can be applied. Most of our respondents were using relatively
informal notations and techniques to do so. Formal languages (e.g., temporal logic, architecture description languages) describing software requirements or architectures were rarely utilized. For example, for describing software requirements as well as architecture and design, only 2% of the companies use formal languages in all of their projects. In 22% of the companies formal languages are used frequently in the requirements phase and in 14% formal languages are used frequently for architecture and design.

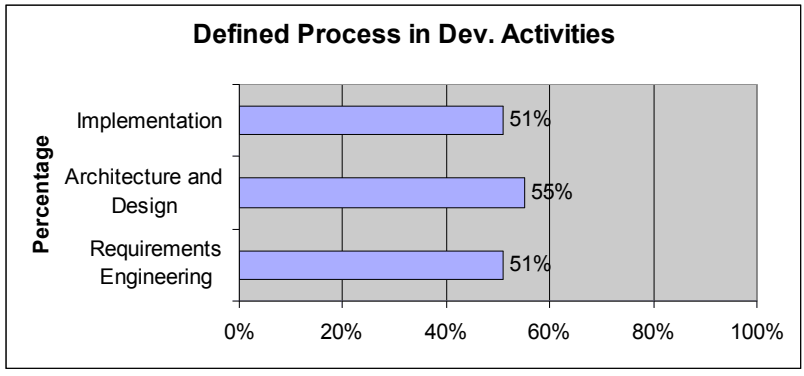


Figure 11: Defined processes for development activities

Consequently, less formal notations and languages are generally more often applied. Figure 12 shows the results for requirements engineering. There, natural language is used in almost all companies in all projects. In 92% of the companies this kind of notation is used always or frequently. A detailed analysis of the answers reveals that for 46% of the companies natural language is the one and only notation to specify requirements. Structured notations such as use cases are used by 40% of the companies on a regular basis (i.e., always or frequently used).

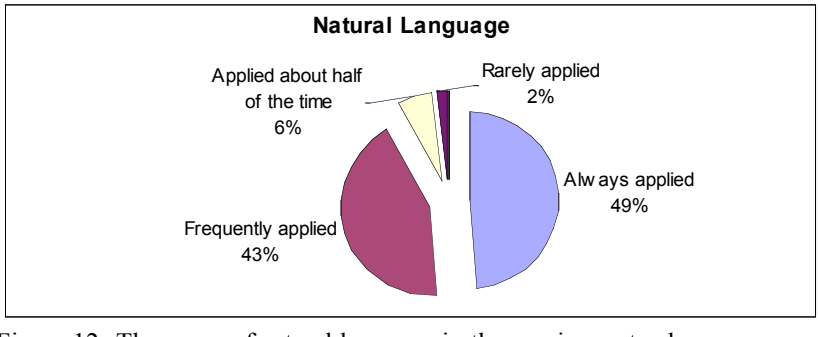


Figure 12: The usage of natural language in the requirements phase

For architecture and design, structural diagrams are the most frequently applied notations for modelling the software. These diagram types (e.g., class diagrams, package diagrams, functional block diagrams) are used by 64% of the companies on a regular

basis (always or frequently applied). Sequence and data flow diagrams seem to have a lower importance. These diagrams are frequently used by 40% and 36% of the companies, respectively. More formal notations such as state charts and/or Time Petri Nets seem to be of low importance in the medical device domain. State charts are used on a regular basis by 23% of the companies, while Time Petri Nets are not applied regularly by any respondent. Only 6% of the companies occasionally use this notation.

For implementation, programming languages of the C-family (C, C++, C#) dominate. C++ is the most frequently applied language (49% of the companies use C++ always or frequently), whereas C is always or frequently applied in 31% of the companies. Java, in contrast, is the only programming language used in 2% of the companies. In 10% of the companies Java is used in combination with other languages.

In addition to these more general questions, we asked the respondents about several good engineering practices for software development, again focusing on the three activities requirements, architecture and design and implementation. During requirements engineering, a good practice is to refine customer requirements into internal, more detailed (developer) requirements. Figure 13 shows that this practice is applied on a regular basis by 78% of the companies. Consequently, this good practice seems to be a standard practice in software development for medical devices.

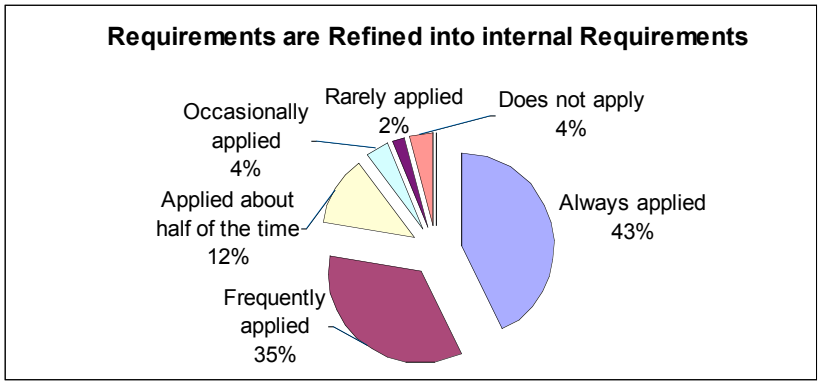


Figure 13: Refinement of requirements into developer requirements

Another good practice, the elicitation of requirements in close cooperation with the customers is also performed by most of the companies (75% follow this practice).

Several good practices for architecture and design activities are also commonly adopted. For example, the architecture and design models are verified against the requirements by 65% of the companies. Focusing on fault detection in the architecture is used by more than 42% of the companies on a frequent basis. On the other hand, concepts for dynamic system reconfiguration in case of a fault are applied by only 10% of companies.

Regarding implementation, coding standards are defined and followed by 63% of the companies on a regular basis. Other good practices, such as limiting the size of modules or the number of parameters, are not as frequently applied (Figure 14). The size of modules is limited on a regular basis by 43% of the companies and the number of parameters is constrained by 26%.

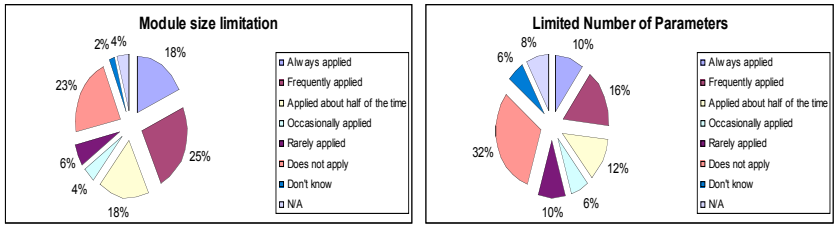


Figure 14: Application of good practices for implementation

Another important practice is the explicit specification of the interfaces of code modules. This practice is applied frequently by 48% of the companies and in about half of their projects for another 20% of respondents.

A relatively new approach to implementation, pair programming, has only a low importance in this domain. It is frequently applied by only 8% of the companies.

Many tools exist that assist with various constructive techniques (e.g., requirements management tools, design modelling tools, test execution tools). Frequent tool usage may be an indication of higher process maturity in software development activities. However, in order to make use of a tool it is important to have a good process definition in advance. A tool cannot overcome deficiencies regarding the followed processes. Consequently, tool support can only be an indication for higher process maturity under the assumption that well-defined processes are applied along with them. Figure 15 shows that tools are not frequently used in this domain. For example, even though testing can be easily supported by tools, less than 13% of the companies use tools on a regular basis for this purpose.

Tool support for other types of activities is also infrequent. Specifying and managing requirements is tool-supported in 20% of the companies. (For this purpose DOORS™ (16%) and RequisitePro™ (10%) are the most frequently used tools.) Architecture and design activities are supported by tools in 35%

of the companies. Here UML modelling tools (25%) had the highest importance. Tools that are typically applied in many other domains of embedded software (e.g., in the automotive industry), such as Matlab Simulink or LabView, seem to have a lower importance in the medical device domain. Matlab is applied by 8% of the companies on a regular basis; LabView by 12%.

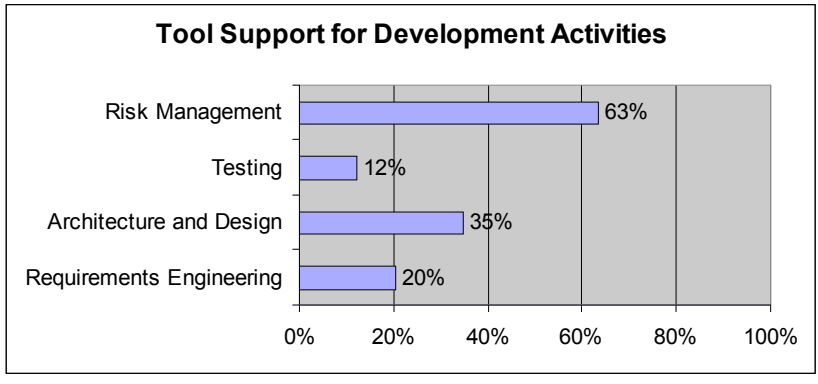


Figure 15: Usage of tools for different activities

Risk management activities seem to be the activities that are most frequently supported by tools (63% of the companies use a risk management tool on a regular basis). However, these numbers must be carefully interpreted. A detailed analysis of the data shows that almost all of the companies state that they do not use commercial tools specialized to support this activity but common applications such as text editors (like MS Word™) or tabular calculation sheets (e.g., MS Excel™).

4.1.3 Quality assurance and risk management

Quality assurance activities are an integral part of software development processes. Especially in safety-critical domains such as the medical device domain, risk management activities

that are used to analyze, control and monitor safety risks of the devices and the software included therein are of high importance. The following diagrams indicate the state of the practice regarding general quality assurance and specific techniques such as testing, inspections, and risk management.

Figure 16 indicates that the quality assurance activities are important in the overall development process. 60% of the companies systematically plan quality assurance activities while another 14% establish such plans for half of their software development projects. Considering the high number of small- and medium- sized software development teams among our respondents, it is interesting to find such a high amount of planned quality strategies. Figure 17 shows that the result of software quality assurance techniques are frequently documented. 62%of the companies always or frequently document them; 16% of the companies document the results in half of their projects.

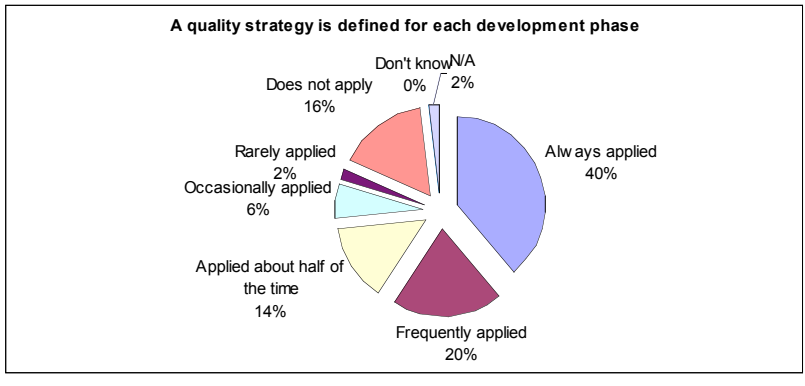


Figure 16: Quality Strategy planning.

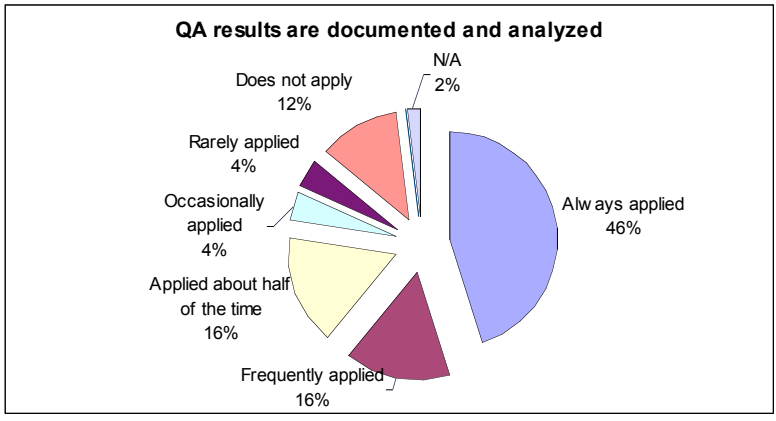


Figure 17: Documentation of quality assurance results

Planning software quality assurance activities can be supported by applying activities such as the collection of code or design metrics or performing benchmarks of the software. It seems that these supporting activities are not applied in a wide range of medical device companies. Less than 5% of the companies always collect code or design metrics of their software. Benchmarks are frequently used by less than 13% of the companies. The most frequently applied supporting measure is defect classification: 35% of the companies use a defect classification to understand better the nature of the software faults. However, it remains unclear how detailed this classification is defined in practice. Consequently, judging their value for supporting quality assurance planning is not possible.

Looking in more detail at the quality assurance techniques, testing was the most frequently applied technique. System validation activities applied to the software (i.e., testing / validating that the software fulfils its requirements) are performed by 78% of the companies in all projects and in a further 16% of the companies frequently (Figure 18). Inspections and reviews (i.e., the verification of the quality of intermediate development products) are also applied quite frequently: 32% of the companies perform this activity in all of

their software development projects, 24% apply the technique frequently, and another 16% perform inspections in half of their projects (see Figure 19)

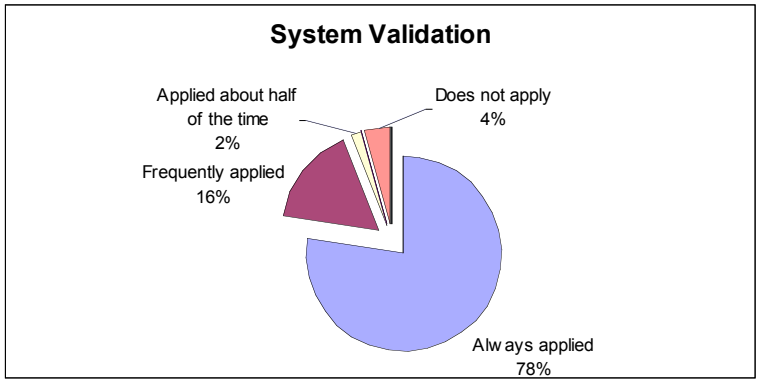


Figure 18: The use of system validation

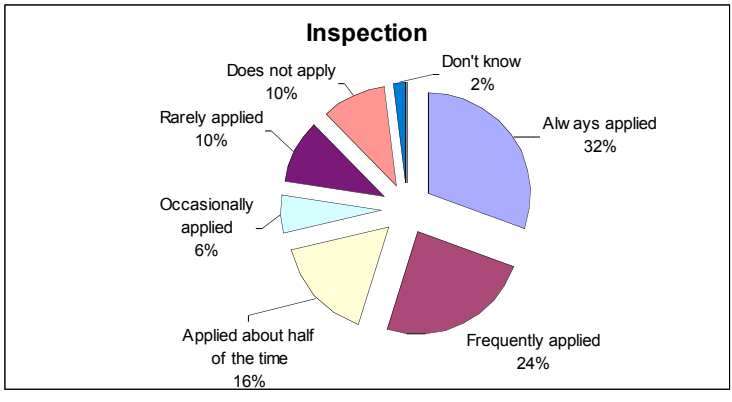


Figure 19: The use of inspections

Testing can be performed using different techniques with different quality foci. For example, test cases can be created using functional testing techniques that focus on the final functionality to be provided by the system (e.g. equivalence class testing or boundary value testing), or they can be created based on structural testing techniques that focus around the internal structure of the software. The two top diagrams of

Figure 20: Test Techniques and Test Scope represent some quality foci that can be prioritized in testing activities. The responses show that software performance testing is an important issue: More than 50% of the companies always or frequently apply performance tests to their products. Another 24% apply these tests in half of their projects. Interface tests are also frequently applied. For 66% of the companies these tests are always or frequently part of the validation and verification activities. Another 24% of the companies include these tests in the test strategies of half of their projects. Furthermore, Figure 20 indicates that in the medical device domain functional testing techniques (also known as black-box testing since the techniques operate with no knowledge of the internal structure of the software) are the most frequently applied. 77% of the companies state that they use this technique always or frequently for testing purposes. Another 12% apply these techniques in about half of their projects. Interestingly, structural (or white box) testing techniques, which examine the coverage criteria to assess how much of the internal structure of the software has actually been activated during tests, are less frequently applied. Only 29% of the companies apply coverage criteria and structural test techniques on a regular basis.

Another interesting finding is that model-based testing is of low importance in the medical device domain. This is an emerging test strategy where test cases are derived from software models; if executable software models are used then the test cases are actually executed on the models themselves. This approach is applied in less than 21% of the companies. This finding is consistent with other findings indicating that in the medical device domain, few formal models are used and tools are rarely used to model or design the software.

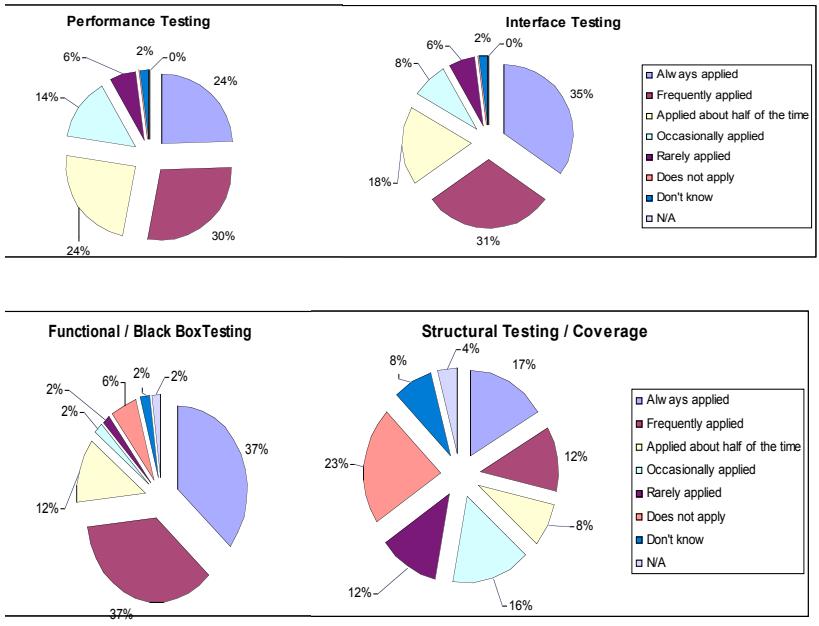


Figure 20: Test Techniques and Test Scope

In accordance with the requirements of the standard for the application of risk management to medical devices (ISO 14971), risk management activities are frequently performed during the development process. Most companies (76%) apply risk analysis and risk monitoring activities (i.e., the continuous update of identified and documented risks in case of changes and the continuous validation of the effectiveness of risk mitigation measures) in the development processes. However, a defined risk management strategy according to a standard is followed by only 44% of the companies on a regular basis, another 6% of the companies have such a strategy in half of their projects.

Risk analysis can be performed using different techniques such as Fault Tree Analysis (FTA) or Failure Mode and Effect Analysis (FMEA). These techniques are recommended by various standards as suitable approaches for systematically identifying the most relevant risks during development of a medical device. Figure 21 indicates that FMEA is the most

frequently applied technique for risk analysis. 42% of the companies always or frequently use the FMEA technique for this purpose. FTA seems to be of lower importance as only 18% of the companies always or frequently use this technique.

Another interesting finding is presented in Figure 22. Only 18% of the respondents stated that they perform software FMEA on a regular basis in their development processes. Since this is rather less than the number of respondents applying FMEA in general, companies seem to be applying the FMEA technique at the system level, rather than specifically to the software component.

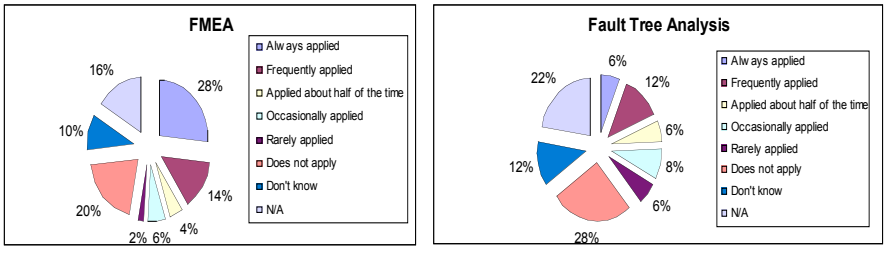


Figure 21: FMEA and FTA Usage

This finding is consistent with our practical experiences that even though risk analysis should be performed on the software and the software components it seems to remain unclear how this should be performed.

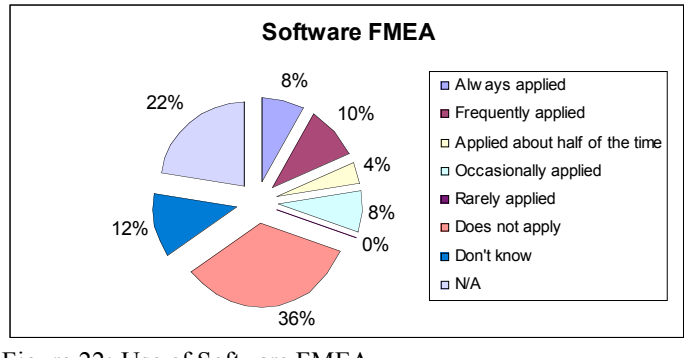


Figure 22: Use of Software FMEA

Other risk analysis techniques (e.g., Hazard and operability analysis (HAZOP), reliability block diagrams, event trees) seem not widely used in the medical device domain as they were not mentioned by the respondents as alternative techniques for risk analysis.

4.1.4 Software reuse and third party software

The software embedded in the medical device can be developed by the manufacturers themselves or by sub-contracted suppliers. Furthermore, the software for the device can be developed from scratch or by reusing existing parts. Results in other domains have shown that, if performed in a systematic way, reusing existing software components can lead to higher quality, higher productivity and shorter time-to-market.

Along with software being developed in-house by device manufacturers, integrating software obtained from a third party is also a frequently applied strategy in this domain. Figure 23 shows the sources of the software embedded in the products of the respondents. 63% of the companies use third party software either solely (12%) or in combination with their own software (51%). 35% of the companies exclusively use their own software development team.

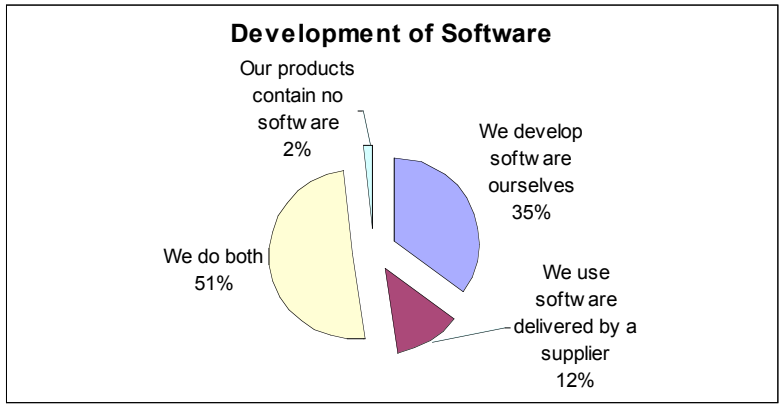


Figure 23: Sources of the software in the medical devices

Third-party software accounts for up to 40% of the total software in more than 2/3 of the companies.

As the delivered software is of a high importance for the overall product quality, companies are using different methods to ensure the quality of the delivered software (see Figure 24). Testing the delivered software is the most frequently applied method for quality assurance of delivered software (89%). Other activities, such as assessing the software process of the supplier (50%), analyzing the results of quality assurance activities such as reviews and testing at the supplier's site (47%), or checking compliance with either company-specific or industry wide standards (44%) are applied by many fewer respondents. Using software metrics to evaluate the delivered software is applied by almost no respondents (6%).

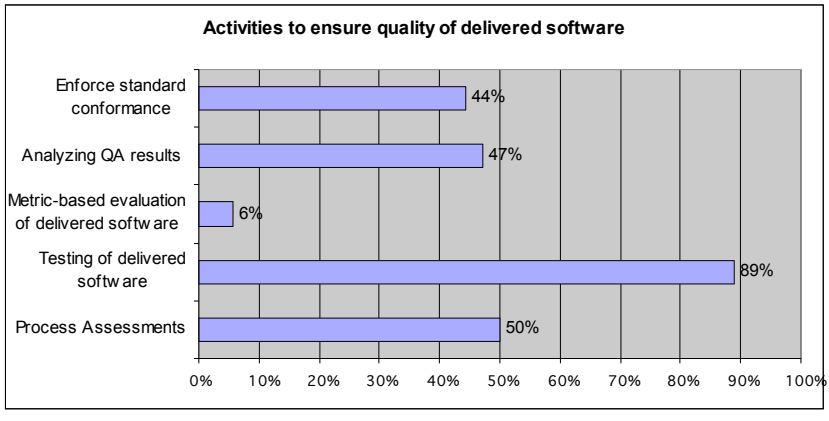


Figure 24: Quality assurance activities for third-party software

Figure 25 shows the most significant problems that are encountered with software delivered from suppliers. For 56% of the companies incorrect implementation of the functionality is problematic, followed by problems with interfaces (i.e., the delivered software is not compatible with interfaces to either other software components or to hardware elements (47%).

More than one-third (36%) of the respondents perceive it hard to judge the quality of the delivered software at all.

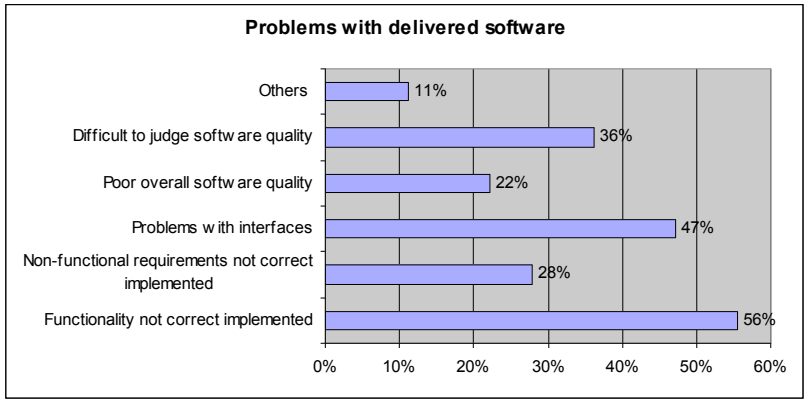


Figure 25: Problems with delivered software

To respond to such quality problems, 22% of the companies are thinking of developing the delivered software on their own in the future. 58% are planning to use more rigorous quality assurance techniques and processes on the delivered software to improve its quality.

Reusing existing software that was previously developed in-house by the companies also seems to be a frequently applied strategy. 48% of those companies reusing software state that they reuse components from earlier version of the software by including them in the new version of the product. In these cases, the reused parts make up 21-60% of the whole software. For 20% of the companies the software contains between 1 and 20% reused parts, 24% of the companies state that the reused parts count for 61-100% of the software (see Figure 26).

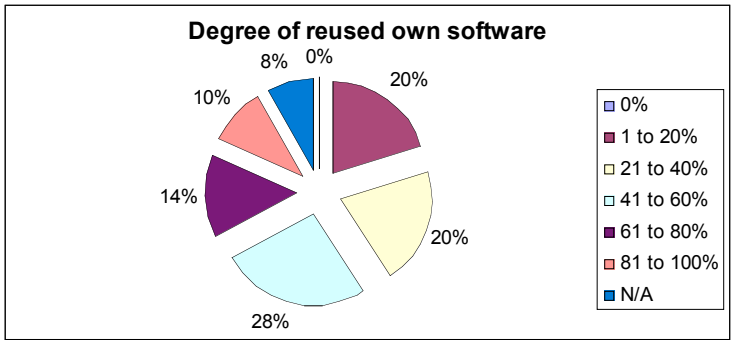


Figure 26: Degree of reused software developed by the companies

Beside software components developed by the companies themselves, open source software and components purchased "off the shelf" (COTS) can be used in the products. However these types of software components have a much lower application in the medical device domain. 29% of the companies do not use open source software and for another 45% open source software components account for less than 20% of the software. Interestingly, in 4% of the companies, open source components make up greater than 80% of the software. The numbers for COTS are quite similar (27% do not use COTS, 39% use COTS components as less than 20% of their software product, and 16% of the companies use COTS for between 21% and 60% of the software).

In summary, it is interesting to note that 98% of the companies perform reuse of software components in some form or another (either from in-house, COTS, or open source). Only 2% of the companies develop the entire software component from scratch for each product indicating that the embedded software continuously evolved over different versions of the medical devices.

4.2 Investigation of relationships between different variables

This section moves beyond descriptive statistics to look for correlations between different aspects of the responses. For certain key questions, we examine here whether different types of companies within the domain behave in characteristically different ways. Because of the number of comparisons and the unbalanced nature of the dataset, we did not run the Chi-square test. Rather, we treat this as a provisional study, which generates some hypotheses based on patterns in the data.

4.2.1 Relationship of country and application of standards

Many software development organizations in many domains undergo assessments of their development processes against standards such as the SEI's CMMI, ISO, etc. Although the specifics of the methods differ, the broad goal of any assessment is to ensure that proven engineering practices are being applied and are being performed with proper rigor. In some domains, assessment against these standards may be mandatory while in others it is up to individual organizations whether or not they want to ensure that their procedures are in compliance. Moreover, various standards are of different importance in different geographic regions, based on local regulations and what is better known to local industry. Respondents of our survey were asked a number of questions that targeted their usage of these standards.

One interesting observation is that all 8 US companies are not CMMI certified, despite the popularity of CMMI in many other market segments. Hence it seems like CMMI does not play an important role for the medical device developers. Similarly low rates of adoption are found in the other geographic reasons. Thus, even though 3 of the 8 US companies plan to get

certified in the near future, CMMI seems to not be of large interest for the medical domain. As only a few companies were certified against a software related standard it was not possible to perform any statistic analysis.

4.2.2 Relationship between tools used and organizational characteristics

An organization's investment in software development and management tools is a good indicator of its commitment to processes and the areas it finds most important. Because tools require an investment in money (to buy or to build the tool itself) as well as effort (people's time invested in training on the tool or overcoming the learning curve), tool usage is not a casual undertaking for an organization. In designing and analyzing this survey, we were interested in whether patterns could be detected for usage of different types of tools, or usage by different types of organizations.

One important factor that we considered in this analysis was the type of activity (requirements, design, implementation, risk management) that the tool addressed. Activities for which a high percentage of respondents had adopted tools were likely to be activities that were felt to be very important or very challenging, and thus requiring of extra support. Another factor in our analysis was the background of developers on the team. In particular, we looked for any differences between teams which contained a majority of members with a background in Computer Science, CS (who might be expected to learn about state-of-thepractice tools as part of their formal education), and teams that had a majority of members from other domains. This distinction can help to identify whether there are tools from particular domains that can be useful for the development of medical device software, although they may not have been widely disseminated yet.

In the following analyses, we consider the percentage of teams of various types who were using the specific tools versus the percentage of teams who were not using them. Respondents whose teams were counted as using a particular tool were those that answered that they used a tool "always," "frequently," "half of the time," or "occasionally." Teams that were counted as not using a particular tool were those for which respondents answered that the tool was "rarely" used or that the question did not apply. Respondents who answered, "don't know" were dropped from the analysis, as were all respondents who simply skipped the relevant question.

The following results were obtained for tools related to **requirements** activities, which we defined as any activities in which the functional and quality requirements of the software are elicited, analyzed, and specified:

Tool:	% of teams, with a majority of CS personnel, who used	% of teams, with a minority of CS personnel, who used
Doors	50%	22%
Reqtify	0%	0%
Requisite Pro	36%	14%
Caliber	38%	0%
Others	90%	67%

Table 4.1: Usage of requirements tools for different types of teams

All of the named tools had very low adoption rates among teams with less CS background. In all cases, CS teams had much higher rates of adoption (see Table 4.1). There are a lot of

"other" tools being used, but few of these were used by more than one organization.

The following results were obtained for tools related to **architecture and design** activities, which we defined as any activities in which software requirements are used to generate software components, and an architecture that shows how they relate to one another to provide the technical solution (see Table 4.2):

Tool:	**% of teams, with a majority of CS personnel, who used**	**% of teams, with a minority of CS personnel, who used**
Matlab/Simulink	42%	36%
LabView	42%	75%
UML Modeler	55%	40%
Room Modeler	5%	0%
SDL Modeler	10%	0%
Statemate	29%	20%
Others	60%	33%

Table 4.2: Usage of architecture and design tools for different types of teams

The number of tool users in the architecture and design category was much higher than in any other. In general, the differences between CS teams and the others were much lower for this type of activity than for any other. Interestingly, LabView is shown to be the only tool that was mentioned in the survey which was adopted by a higher percentage of the teams

with a minority of CS members than by teams with a mostly CS background.

There are several possible explanations for the generally high rates of adoption of these tools by both types of teams: Architecture and design may be rated as a very important activity for many teams, regardless of whether or not they have a background in CS; the tools for this area may be more helpful and/or more easy to use than tools for other activities, leading to more widespread adoption; this activity may simply be too difficult to do without some kind of tool support.

The following results were obtained for tools related to **test** activities, which we defined as any activities, which are performed to ensure the quality of software components or the software system:

Tool:	% of teams, with a majority of CS personnel, who used	% of teams, with a minority of CS personnel, who used
TestView	0%	0%
Test Director	18%	8%
Automated Test Manager	13%	0%
WinRunner	32%	8%
QACenter	7%	8%
Others	77%	0%

Table 4.3: Usage of test tools for different types of teams

The tools mentioned in the survey for performing this activity had generally very low adoption rates, especially for the teams

in which members with a CS background were in the minority (see Table 4.3). CS teams had a lot of other tools that they used for this, most of which were developed internally at their own organization.

The following results were obtained for tools related to **risk management** activities (i.e., identifying and managing risks):

Tool:	% of teams, with a majority of CS personnel, who used	% of teams, with a minority of CS personnel, who used
Risk Radar	0%	0%
Apis IQ	0%	0%
SCIO[tm] FMEA	0%	0%
RQM-FMEA	7%	0%
Standard applications	88%	85%
Others	78%	0%

Table 4.4: Usage of risk management tools for different types of teams

Adoption rates for all of the tools specifically designed for risk management were at or close to 0% (see Table 4.4). However, most respondents regardless of background were using standard applications (like Microsoft Office Word™ or Excel™) for purposes of accomplishing this task. As with the test tools, CS teams had developed a lot of their own organization-specific tools for this task.

Looking across all of these types of activities, the one task for which respondents seemed most interested in tool support was architecture/design modelling. This was also the task for which there was the smallest difference in adoption rates between the

CS teams and the others. Most of the other types of activities had rather small rates of tool adoption for the alternatives that were listed. Since few respondents had other tools that were commonly used for these activities but that were not included on our lists, the low rates of tool usage seem to accurately characterize tool usage in the medical device domain. We are unable to say at this point whether this is because tools would simply need to be better customized this domain in order to be more useful, if developers in this domain are less likely to be aware of or research the tools that are available, or if the existing tools are too expensive or are not used for some other non-technical reason.

In summary, there were some characteristic differences seen for teams that had a majority of members with CS (and hence presumably software development) training, and those that did not. For all tools except one, teams with majority CS expertise were at least as or more likely to have adopted tools than other teams. It is also interesting that for requirements, test, and risk management, CS teams were likely to have developed their own tools (although it is not clear as to whether this was seen as a cheaper alternative or one that better fit the domain). Teams with a minority of CS expertise had very low rates of doing this.

4.3 Relationships between device characteristics and SE-techniques

There are a wide variety of techniques and practices available for use in software development, but software engineers recognize that the techniques, which are selected, will vary from one project to another, as they need to be tailored to the final quality goals and constraints of a development task. This section examines the ways in which those tradeoffs are made in the medical domain.

4.3.1 Relationship of device function and safety criticality

Medical devices fulfil several functions and purposes. The respondents were asked which kind of medical device they produce and had nine different categories from which to choose. The organization of the respondent can produce more than one product, which means that therefore the respondents can give more than one answer to the question. The options available were:

1. Invasive medical devices (e.g. dialysis machines)
2. Non-invasive devices (e.g. defibrillator)
3. Measurements systems (e.g. sphygmomanometer)
4. Analysis system (e.g. spectrometers)
5. Diagnostic systems (e.g. fluoroscopes)
6. Surgical devices
7. Implants and prostheses (e.g. cardiac pacemakers, artificial legs)
8. Medical information systems
9. Others

The most respondents were in the categories Non-invasive devices (24%), Medical information systems (22%) and Diagnostic systems (18%). The fewest responses were in the areas of Implants (4%).

The respondents were asked in the survey if the software or the medical system as a whole is considered safety critical by the respondent's organisation. For products matched to the four categories Invasive medical devices, Analysis system, Surgical devices, Implants and prostheses, all of the respondents stated that their products are safety critical. In the other categories (Non-invasive devices, Measurements systems, Diagnostic systems, Medical information systems and Others) there are products that are not safety critical according to the respondents. The category Others with 36% non-safety critical systems is the category with the maximum percentage of non-

safety critical systems followed by the category Measurements systems (15%). In the category Others the respondent have mentioned for example microscope, navigation systems and dental CAD. We can find products that are not safety critical according to the respondents. In general the categories were not significantly different from one another regarding the percentage of safety vs. non safety-critical systems in each.

The primary capabilities of the software classified as non-safety critical were described by the respondents as providing functionality for control, analysis and human-computer interaction, as some examples. For the software described as safety critical by the respondents, in almost half of the cases the most important capabilities of the software were described as the control and management of hardware, followed by handling the quality of the user interface.

Since the classes are not orthogonal, no statistical test about the difference has been carried out. However, the data do seem to match intuition: those systems that are used in situations that are intuitively safety critical are all seen as safety critical by the respondents, while systems that are not intuitively safety critical are to a larger extent seen as non safety critical.

4.3.2 Relationship between safety criticality and the model and techniques applied

Medical devices can be classified according to the risks that the human body can be exposed to due to the design, the use or the mode of manufacturer of the medical device. The software that runs a medical device or affects the use of a device automatically belongs to the same class as the device. This is valid both in the USA and in the EU. In the EU, medical devices are categorized according to the Medical Device Directive (MDD) into four classes (Class I, IIa, IIb and III), and in the US they are

categorized by the FDA into three different classes (Class I, Class II and Class III).

The participants in the survey were asked if the software or the medical system as a whole is considered safety critical by the respondent's organization. 37 of the respondents answered yes (considered safety critical) to this question and 8 respondents answered no. As shown in Table 4.5 we compared the respondents' opinion of the safety criticality of their products to the actual classification according to the relevant government organization.

The results show something of a mismatch between the safety criticality rating according to the MDD or FDA, and how the respondents considered their own software. In Table 4.5 and Table 4.6 "Rated as safety critical" means that the system is considered to be safety critical by the respondents and "Rated as not safety critical" means that it is not seen as safety critical. In Table 4.5"Classified as safety crit." means that the product is classified as safety critical according to MDD or FDA regulations and in Table 4.6 the distribution between the different types of classes is shown. The respondents have stated themselves in which class(es) their products are classified according to MDD or FDA regulations. The majority of systems were rated in agreement with the regulatory classification (30/36 vs. 7/9). It is important to notice that companies can have products in several different classes and the respondent can have stated more than one class, which is reflected in the figures.

	Rated as safety critical	**Rated as non safety critical**
Classified as safety crit.	30	6
Classified as non-safety crit.	7	2

Table 4.5: Respondents' rating vs. regulatory classification

	Rated as safety critical	Rated as non safety critical
Class EU IIa, II b, III + Class FDA II, III	33	7
Class EU I + Class FDA I + Not classified	20	5

Table 4.6: Type of class vs. safety criticality

Seven of the medical systems or devices that are considered not safety critical by the respondents (see Table 4.6) are classified in the higher classes II and III and should therefore according to MDD and FDA be considered as safety critical. There is a risk that these respondents have misunderstood the questions or they are not aware of the meaning of their MDD or FDA classification. If this is not the case it can maybe reflect the respondents apprehension of the part of the system or device he/she are working with, the respondent do not consider it to be safety critical even if the classification indicates the opposite.

All 8 of the respondents that consider their software or the medical system as a whole to be not safety critical state that their organisation develops their software themselves. 6 of the 8 also state that they buy some of the software.

7 of the 8 respondents who do not consider their software safety critical state that the software is “very important” for the realisation of the functionality in their product. (The other 1 respondent said that it was “important.”) This basic view of the importance of software in the larger system is the same as for the safety critical group, i.e., no significant difference was found.

The majority of respondents (57%) state that the process model used is the V-model, followed by the Iterative model (49%). It is not possible to show any significant difference in the distribution of chosen process model for systems that are considered to be safety critical and for systems that are not considered safety critical as shown in Table 4.7. The data is unbalanced since in the data set the majority of the respondents consider that their system or device is safety critical.

	Safety critical	Not safety critical
Use V-model	21	5
Not use V-mode	16	3

Table 4.7: Chosen process model vs. safety criticality

At a more detailed level, the respondents were asked to characterize their organisations´ approaches regarding requirements engineering activities, architecture and design activities and activities in the implementation. To characterize their activities they were presented in the survey with a set of common good work practices (represented as the statements listed in Table 4.8), and asked to indicate which of these they use in their own work activities:

Requirements engineering	**Architecture and design**	**Implementation**
We follow a defined requirements engineering process	We follow a defined architecture and design process	We follow a defined implementation process
Requirements are elicited and defined together with the customer	The architecture and/or software design is explicitly traced back to the requirements	Elements of the code are explicitly traceable to the related elements in the architecture and design
Requirements are refined into a more specific set of requirements used internally by developers	The architecture and/or design is verified against the requirements	We follow well-defined coding standards
Requirements are verified to ensure their quality	The outcomes of architectural and design activities are provided to all affected parties	A unit strategy is established
Changes to requirements are under change management control		

Table 4.8: Statements for characterizing techniques used by organizations

Regarding requirements activities, there is a difference between the approaches taken by respondents with safety critical software and those without. Most (about 80%) of the respondents with safety critical products state that they always or frequently refined the requirements into a more specific set of requirements used internally by developers, and that the requirements are elicited and defined together with the customer. They agree to the other statements but to a lesser extent (65% or less). For example 35% state that they occasionally or never do changes to requirements under change management control. A majority of the respondents who said that their software is non-safety critical agreed with all the statements regarding requirements engineering to a greater extent, (88%) than the respondents with safety critical software. With one exception, a defined requirement process is always or frequently used by 71% of the respondents. There were no techniques listed that were *never* used by the non-safety critical respondents. These results are interesting, as they seem counter-intuitive: It might have been expected that the safety critical respondents would be more inclined to agree with these statements than the non-safety critical respondents.

When it comes to the work procedures for software architecture and design we see the same trend as for the requirements. The work procedures are slightly more frequently used by the non safety critical than by the safety critical group. One of the respondents in the safety critical group even stated that they do not use any of the mentioned work procedures. In the non safety critical group 90% state that the outcomes of architectural and design activities are always or frequently provided to all of the affected parties; the same figure for the safety critical group is 50%.

The only difference for the working procedures for implementation is that there are two respondents in the safety

critical group who state that they use none of the mentioned work procedures. In the non safety group all of the respondents state that they use the work procedures at least occasionally.

The respondents were also asked about the notation used to record software requirements, the concepts and notations used to specify the architecture or design and the principles and techniques applied in the context to create code.

A set of common techniques was provided for each, as shown in Table 4.9, and respondents were asked to indicate which they use.

Requirements engineering	**Architecture and design**	**Implementation**
Use cases	Structural diagrams	Software module size is limited
Formal notations	Logic/function block diagrams	Information hiding is used to structure the code
Natural language	Sequence diagrams	The number of method/function parameters is limited
Others	Data flow diagrams	Interfaces of modules and units are fully specified in an interface specification
	Finite state machine	Joint code reviews or walkthroughs are used to find problems
	Time Petri net	Formal code inspections are used to find problems
	Formal Language	Pair programming is used to create the code
	Fault detection techniques are implemented	Others
	Dynamic reconfiguration of the system during run-time	
	Others	

Table 4.9: Lists of common techniques and notations for various software activities

When it comes to the notation used to document software requirements, there is no significant difference found between

the respondents who have stated their products to be safety critical and those who have stated theirs non-safety critical. For both groups, natural language is the most frequently used notation and formal notation the least frequently used. In the safety critical group, 92% answered that they always or frequently use natural language; for the non-safety critical group that figure is 100%. Formal notation is used always or frequently by 27% of the safety critical group and by 13% of the non-safety critical group. This seems to correspond to intuition, as it could be expected that the safety critical group would more frequently use formal notation because of the products safety criticality, in order to avoid ambiguous or inconsistent requirement.

Between the two groups no difference is found when it comes to principles and techniques applied to create code. Interface specifications are the most often used technique for both the safety critical (49%) and non-safety critical (50%) groups. Neither groups stated that they always or frequently use pair programming.

Regarding the use of concepts and notations to specify architecture or design, a difference can be seen between the two groups. Fault detection is implemented (i.e., the system can detect faults and trigger reactions) by all of the respondents in the non safety critical group but only by half of the members in the safety critical group. The use of structural diagrams is the most used notation in both the safety critical group (62%) and the non safety critical group (75%).

To survey the use of different tools and programming languages the respondents were asked about tools used to track and manage requirements, tools and/or notations used in the context to support architecture and design activities and what programming language used by the companies. The different tools used by the safety critical group and non safety critical group to track and manage requirements are presented in Figure

27. In Figure 28 are the tools that are used to support architecture and design activities for each group.

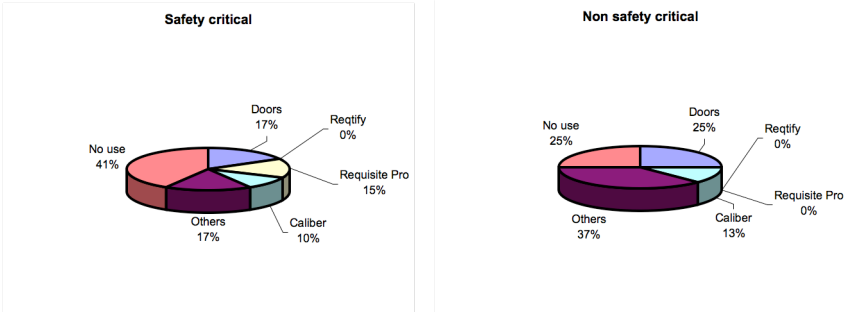


Figure 27: Tools to support requirements activities, by group.

It can be worth observing that 41% of the respondents in the safety critical group state that their company does not use any tools at all to trace and manage requirements (the "No use" category shown in Figure 27). The same figure for the non safety critical group is 25%.

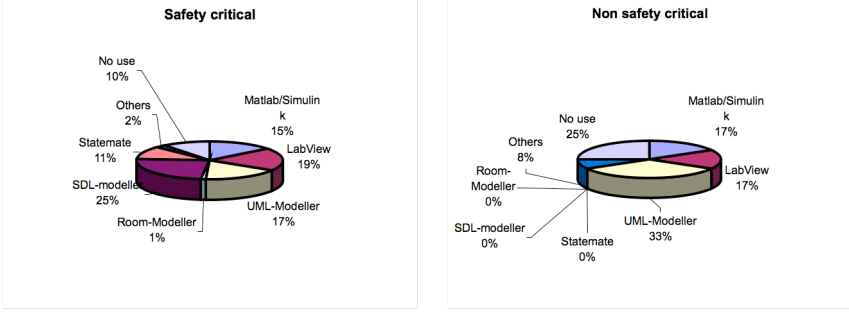


Figure 28: Tools to support architecture and design activities, by group.

There are some differences between the two groups. The most often-used tool in the safety critical group is the SDL modelling tool, but this tool is not used in the non safety critical group at

all. UML modellers (e.g. Rationale), which are the most used tools in the non safety critical group, are much less used in the safety critical group. Statemate (state flow diagrams) are used in by 11% of the safety critical group but not used at all by the non safety critical group. When it comes to the programming language used by the two groups, there are no significant differences between the groups, as seen in Figure 29 C++ is the most used programming language in the both groups.

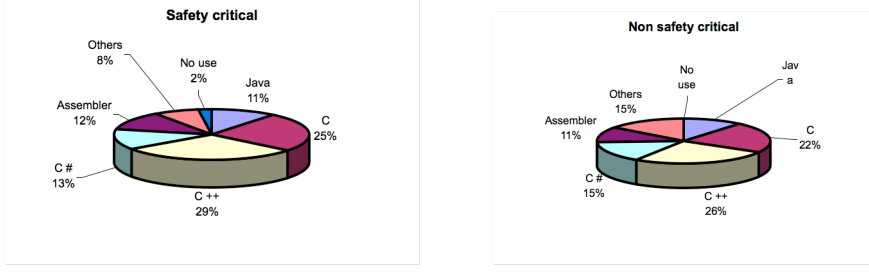


Figure 29: Programming languages used vs. safety criticality

4.3.3 Relationship between the safety-criticality level and the software quality characteristics?

Different quality characteristics are important for different devices and products. The respondents were asked to consider a list of eleven quality characteristics and state the importance of each of the quality characteristics regarding their importance for the company's final products.

Table 4.10 shows the answers from the respondents that state that their products are safety critical (37 respondents). The respondents have answered if they agree to that a certain quality characteristic is important or disagree to it's importance or may be neither agreed nor disagreed. The respondents were asked to consider each of the listed quality characteristics. The results

shown in Table 4.11 are from the respondents that state that their products are not safety critical (8 respondents)

Quality characteristics	**Agree**	**Disagree**	**Neither agree nor disagree**
Functionality	36	0	0
Safety	35	0	1
Usability	34	0	1
Reliability	32	0	5
Maintainability	30	0	7
Efficiency and performance	27	1	7
Testability	25	1	10
Reusability	20	5	11
Security	19	4	13
Portability	9	13	14

Table 4.10: Safety critical systems and the importance of different quality characteristics

Quality characteristics	Agree	Disagree	Neither agree nor disagree
Functionality	8	0	0
Maintainability	8	0	0
Usability	8	0	0
Reliability	8	0	0
Efficiency and performance	8	0	0
Testability	7	0	1
Reusability	7	0	0
Safety	5	2	1
Security	4	2	2
Portability	3	2	3
Others	0	0	0

Table 4.11: Non safety critical systems and the importance of different quality characteristics

Functionality is the quality characteristic that is stated as important by most of the respondents whether the system is safety critical or not. The respondents who have stated their systems not to be safety critical are chose maintainability, usability, reliability, efficiency and performance as important quality characteristics.

As expected, safety is considered very important to most of the respondents with safety critical systems. When it comes to the non-safety critical systems, 5 of the 8 agree that safety is important and 2 disagree. So even for products considered non-safety critical, a majority of respondents think that safety is an important quality characteristic.

Worth noticing is that only 25 of 37 of the respondents for the safety critical systems thinks that testability is an important quality characteristic (10 neither agree nor disagree). Most of the respondents for non safety critical systems on the other hand consider testability to be important. However, the relative difference between the two groups is not very large.

4.3.4 Relationship between the safety-criticality level and the software quality assurance techniques?

Quality assurance activities in this survey refer to any activity that focuses on ensuring the quality of the software and/or the intermediate work products throughout the development lifecycle. The respondents were asked to characterize their quality assurance system according to the following five statements:

- A quality assurance strategy is implemented for all development phases.
- Quality assurance processes are defined for work-products in the different development phases.
- Inspections and reviews are performed on intermediate work products.
- Quality assurance activities are systematically planned for the software and system components.
- The results of quality assurance activities are documented and analyzed.

The most frequent activity, as stated by all the respondents, is that “the results of quality assurance activities are documented and analyzed” (67%). Inspections and reviews are the activity used the least (60)%.

15 of the 37 respondents with safety critical software state that all the five mentioned activities are always or frequently used. 2 of the 8 non safety critical respondents state that all the

five activities are always or frequently used. This is not a significant difference (p=0.44).

For the safety critical respondents, “the results of quality assurance activities are documented and analyzed” is the most often used activity (70%). However, for the non safety critical cases, inspections are the most used technique (88%, all but one stated that they always or frequently use inspections).

It is interesting that in three cases for the systems stated safety critical, none of the five mentioned activities is used. When it comes to the non safety critical system there is one case where none of the five activities are used. Many of the medical device companies are put through external audits (third party assessment) by a Notified Body or other authority. 17 respondents state that they have no external audits and 34 states that they have. The frequency of audits varies from once a year to more than four times a year. There are no clear differences between the two groups with respect to how often they are assessed in audits.

It is worth noticing that 10 of the safety critical systems do not have external audits. Three of those are classified in Class IIb and Class III and have a higher risk potential and should, according to law, always be assessed by a Notified Body. When it comes to the non safety critical systems, two out of four are not put thought external audits, but should have external audits because they are classified in Class IIb and Class III.

Quality assurance activities can be supported and focused by applying certain techniques. The respondents were asked to state which of the following techniques they use:

1. Source code metrics
2. Benchmarking of the software
3. Design metrics
4. Defect classification
5. Others

Among the 37 respondents that regard their products to be safety critical, 42% state that they use none of the supporting techniques. Only one out of eight respondents in the non safety critical groups states that they don't use any of the techniques. There is no difference between the two groups (safety critical and non safety critical) regarding the techniques that are most and least used. The most often used technique is defect classification (safety critical 35% and non safety critical 38%) and the least often used is design metrics (both groups 13%).

4.4 Investment in certification

Almost all medical devices need to be certified by a government institution to get clearance for the market. Respondents were asked to estimate the size of this investment in certification, as a measure of how rigorous these certification processes are. Since certification expenses are expected to vary with the type and criticality of the device, we asked for an estimate of the certification effort as a percentage of the total product development cost. The results are summarized below:

	America	European	
	US/Canada	Scandinavia	Europe + Israel
Total	**8**	**6**	**41**
0%	0	2	1
1-20%	5	2	21
21-40%	0	0	11
41-60%	0	0	2
61-80%	1	0	1
81-100%	0	1	0
Don't Know	2	1	4

Table 4.12: Efforts needed for certification of the product

From this data we can see that the majority of responses indicate that it takes between 1% and 20% of the product cost, regardless of where the company is located.

5 Discussion of the Results

This section discusses the findings in relation to the research questions of the survey. Based on the analyses presented in the last chapter we summarize and discuss the most interesting aspects of software development for medical devices in Section 5.2. The threats to validity of our findings are discussed in Section 5.1.

5.1 Trustworthiness of study

The trustworthiness of the study depends on both the validity and the generalizability. The validity denotes the accuracy of the results, i.e., to what extent the results and relationships reflect the true reality. Generalizability denotes how general the results are. In this kind of study there are a number of threats to both the validity and the generalizability, as discussed for example by Robson [Rob02]. These are summarized below.

5.1.1 Threats to validity

The validity is affected by the *reliability*, which denotes the degree to which the measures correspond to the real world phenomena of interest. If measurements for example to a large extent are affected by what time of day they are collected, who collects them, etc, the reliability will be lower.

In this survey the participant error, i.e., the effect of subjects answering incorrectly, is removed by formulating simple questions that are not to a large extent possible to misinterpret. Also, in this survey, the answers from subjects who did not complete the whole questionnaire are discarded in the main analysis, which means that only the answers from subjects who have completed the whole survey are included. We feel that the persons who have taken the time to complete the whole

questionnaire are also more likely to have taken the time to carefully read the questions. However, there is always a risk that the participants have misunderstood terminology in the questions or the whole questions.

Participant bias is the effect of participants deliberately answering incorrectly, for example to describe an overly positive picture of their organization or project. The chance of participant bias in this study is lowered by allowing the participants to be anonymous. Hence, the subjects would not gain anything from exaggerating the positive aspects of their projects. There is probably a larger risk that subjects who are frustrated over the procedures, or lack of procedures, in the projects they are participating in consciously or unconsciously exaggerate the lack of quality assurance procedures. Care must be taken when interpreting the results for this reason.

The observer bias, i.e., the effect of errors of the researchers when collecting and analyzing the results, is assumed to be small, since the measures are rather straightforward and easy to collect. By using a level of detail in the allowed responses for participants that minimize subjectivity or the need for interpretation, the effect of observer bias is probably small. The researchers have not favored any specific results. However, the choice of questions has probably been affected by the previous experience of the researchers in other domains (mainly telecommunication and automotive).

The construct validity is affected by how well the collected metrics represent the concept that the researchers intend to study. In this survey the concepts and metrics are rather uncomplicated. For example, when the concept of study is the domain of the answering subjects, the subjects were simply asked to answer which domain they were working in. However, as also mentioned above, there is always a risk that participants interpret terminology in questions in a different way than the researcher. In this study there is also a risk that people come

from the same company or even the same project, when they have been treated as coming from different companies in the analysis. This risk is probably not very large, but it exists.

The internal validity denotes to what extent factors other than the ones controlled by the researcher affect the result. In this survey, the primary risk of this problem stems from the threat commonly referred to as "ambiguity about causal direction" when we analyze relations between different survey responses (i.e., does A cause B or B cause A?). However, in this survey, no conclusions are drawn about causal direction, only relations between factors are analyzed. Another potential threat is due to the unbalanced data set. This means that it is hard to find a significant difference between answers from people who see their product as safety critical and people who not see their product as safety critical. This should not be interpreted as a proof of non-difference between the groups. The only conclusion that can be drawn is that no difference can be shown.

5.1.2 Threats to generalizability

The generalizability can be analysed with respect to selection, history, and construct.

Selection refers to effects of obtaining findings, which are specific to a too small group. In this survey, the strategy was to avoid this by sampling from a large population. Invitations were sent out to a large group, which means that people from a large group had the possibility to participate. Also, reminders were sent on two occasions in order to avoid effects of certain types of organizations not participating in the survey.

History effects denote the effects of carrying out research at a specific point in time, which is not relevant later. To our knowledge the survey questions were not answered at a point in time that was seen as special by the participants.

5.2 Summary the results

Overall the survey results indicate that software is and will be a decisive element in the medical device production. Consequently, good software development processes, methods and techniques that help to great high quality software products are a decisive factor for medical device manufacturers. This is supported by the fact the software realizes in more than 83% of the companies safety critical functions. It is also interesting to note that the importance of the software is independent form the size of the company, the product type or product classification. Therefore, there is the need for medical device manufacturers to establish efficient and effective software development units to ensure long-term success of their products.

Based on these findings of the high importance of software it is surprising that software development processes in different activities seem to be of less formality and rigor. The survey results indicate that software development standards and good practices as defined, for example, by SEI's Capability Maturity Model or standards such as ISO 15401 are of low importance in the medical device domain. We can conclude, that these software related standards have in fact no importance in the current state of the practice when developing software embedded in medical devices (only for 10% of the companies CMMI is of relevance). The results further indicate that in the three phases (requirements, architecture, and implementation) only approximately 50% of the companies frequently follow a defined process to perform the activities. Beside the low importance of general standards and models for the software development in medical devices, this finding indicates that many companies seem to perform the different steps in a less formalized, ad-hoc way.

Furthermore, in most companies the majority of the software developers have a background other than computer-science. This, in fact, must not be perceived as

a negative finding, as also in education programs other than computer science programming of embedded systems is taught and people from other domains can be highly efficient and good programmers / software engineers. Nevertheless, the findings support the impression that software development might be performed following less formalized processes.

One reason for the potentially lower importance of systematic software engineering principles might be the small size of the software development teams. Almost 50% of the companies have a software development team with 10 or less people. It is obvious that such a small team might not be able to follow a systematic, formalized software development process. On the other hand this indicates again that the software development might not have the importance during the development process as it should have considering the high impact of software on the products quality and success of the company. This finding also contains an interesting result for applied research institutes developing new techniques and methods for software development. To allow a transfer to industry it is highly important that the techniques are applicable by small and medium sized teams. That is, the techniques, methods and software processes must be flexible and scale for different sizes for software development teams.

Another reason for less formalized and systematic software development processes might be that the medical device domain is currently in a phase where high importance of software development becomes evident for the majority of the companies. We are convinced that key players in the medical device market understand software as a key to success and therefore invest in software development. On the other side it seems that software processes are just becoming a matter of interest in the whole domain. For example, the first standard dealing explicitly with the software life-cycle and some recommended processes to apply within the software life-cycle of medical devices was

released in mid 2006 (IEC 62304). Existing standards (such as the IEC 60601-1-4, or the third edition of the IEC 60601) are applied by the majority of the companies but from our perspective these are often too abstract to give a clear advice on how to set up and execute an efficient software development process. If we look into other domains, for example the safety critical automotive domain or avionics, software related standards have been established much earlier. For example the IEC 61508, CENELEC or DO178-B are quasi-standards in these domains that regulate how safety critical, embedded software must be developed. Furthermore, the automotive industry is currently developing an automotive SPICE standard that is tailored to the software development processes in their domain. We have to carefully monitor the development of similar activities in the medical device domain in the up-coming years.

Another important finding is that the early software development activities are perceived as the most error-prone steps. 63% of the companies perceive requirements engineering as the activities in which most problems with software quality originate from. On the other hand, most of the companies follow good practices in the requirements engineering activities (inspecting the requirements, derive the requirements with customers, refine requirements into internal developer requirements etc.). Consequently, we have to ask why the requirements are still perceived as the most problematic phase. Having a closer look at the requirements related issues, 86% of the companies perceive changing requirements as the main problem. Missing requirements (33%) and misinterpretation of requirements (39%) are perceived as additional issues important in these development activities.

These problems might stem from the less formalized way in which the requirements engineering activities seem to be performed. As the results in section 4 show, the most frequently used notation for the requirements is natural language.

However, this notation is prone to misunderstandings and misinterpretations. It is surprising that formal languages do not have a higher importance in the medical device domain. This finding is also valid for the architecture and design activities where in most cases high-level diagrams are used to describe the structure of the medical device software. More detailed diagrams to analyze the behavior of the software are of lower importance (e.g., dataflow or sequence charts) and more formal concepts (state-machines, petri-nets, formal languages) are of low importance. Again, this might stem from the small size of the software development teams and also from a poor knowledge of alternative notations for software requirements or architecture and design aspects.

The low importance of tools in all development phases in general and especially in the requirements activities might also be a reason for the above mentioned problems (remark: this again is also a indication for the less formalized software development activities). Only 20% of the companies manage their software requirements with a tool. This indicates that in case of changing requirements it might be difficult to estimate the impact of the change on the software. Here a tool supported requirements management process might be a first step to overcome this challenge. However, the introduction of a tool (commercial, open source, or self-made) does not solve the problem as such. It is important to define a tailored and suitable requirements management process first. Only if such a process is established it makes sense to introduce a tool that supports the execution of the process.

Again this finding also indicates some requests from industry on research institutes. Suitable and efficient techniques, methods and processes that support the elicitation, specification and management of software requirements can improve the software development of medical device software in a significant way. Furthermore, a second topic, related to requirements engineering

is of highest interest. Usability engineering is more and more important for the medical devices. As most devices include digital, graphical user interfaces, it is essential to ensure that these interfaces are designed in a way that supports users in a most efficient way.

Regarding quality assurance it is interesting that these activities seem to be the best defined and executed ones in the medical device domain. Especially dynamic verification techniques (i.e., testing) are executed according to a defined plan and results are systematically documented. The reason for this high rigor in executing dynamic verification might be the regulations defined by general (i.e., valid for the product as a whole) domain-specific, standards. In most standards the system (and thus the software) testing activities are the quality assurance techniques that seem to have the highest importance. However, if we take a closer look into the details on performing testing it gets evident that there are also improvement potentials. Even though most companies have a defined testing process, it remains unclear how the process is executed, i.e., which techniques for test case creation and which tools for test execution and analysis are used. Most of the companies state that they use functional / specification based testing techniques for test case creation. However, whether this means error guessing, experience based testing or equivalence based testing remains unclear. Test coverage criteria are only applied by 29% of the companies, indicating that structural tests seem to have a low importance in the domain. Furthermore, it seems that tools for test execution and test analysis are of low importance. Again, companies might have a high potential to improve their software testing activities by introducing systematic test case creation techniques and supporting test execution with tools.

In the context of quality assurance it is surprising that inspections of intermediate software work-products are frequently performed. Again it remains unclear in which way

these inspections are executed (ad-hoc, using a defined process, using reading techniques such as checklists) but the high percentage of companies performing the technique indicates that verification of intermediate software artefacts is an established technique in the medical device domain.

The results from the survey show that a majority of the products that are developed are classified as safety critical by the MDD or the FDA. It is interesting to notice that there are examples from these cases where the respondents regard the developed products as non-safety critical even if they are classified as safety critical. In fact, the relative number of products that are regarded as non safety critical by the respondents is about the same for products that are classified as safety critical as for products that are not classified as safety critical.

Top priority in the medical device domain has the analysis of the safety of the created product. The results indicate that risk analysis is performed on a regular basis by the majority of the companies. This stems from the regulations that such a process must be executed and the results documented in order to get a market clearance for the product. However, the survey results also indicate that safety analysis techniques such as Failure Mode and Effect Analysis (FMEA) or Fault Tree Analysis (FTA) seem to be less frequently applied. FMEA is performed in only 42% of the companies on a frequent basis FTA in only 20%. The question is why these established and systematic techniques are not used more frequently to analyze the risks of the product. One reason might be that the techniques require too much effort in relation to the overall safety risk of the product. However, in such a case it is the question how manufacturers can show that the product is still safe or its risk is acceptable.

A second important finding is that the safety analyses seem to focus on the overall system but are less frequently applied on the software in detail. An explicit software FMEA is performed

by only 18% of the companies. This indicates that the software might be perceived as one piece of the final product that can be analyzed similar to a hardware element of the device. However, safety analyses for software should be performed with much more rigor and detail as characteristics of software are totally different from those of hardware elements (e.g., all failures are systematic, discrete behavior, state and event based behavior). Especially the fact that software is typically responsible to control hardware elements and to realize risk control measures in many products stresses the need for special consideration of software during the risk management activities. A reason for the low percentage of software FMEA performance might be the unsolved question on how to perform safety analyses on software in general. This again is an important research field that can contribute to a more efficient development of save, high-quality medical devices. Such a research investment should include the development of flexible and efficient tools to support the risk analysis activities. The survey indicates that most of the companies do not use commercial tools but standard applications such as word processors or table calculation sheets to support their risk management.

Due to the high market and innovation pressure in the medical device domain it is important to have highly efficient and productive software development processes. Consequently, reusing software is quite frequently performed in the medical device domain. Most frequently own software parts are reused in new versions of a product or for new products that are similar to existing products. It was not possible to analyze the reuse strategies of the companies in detail and thus to analyze whether reuse of software is performed in a systematic and repeatable way. However, one indicator that reuse might not be performed systematically is that 80% of the companies state that they perceive the increasing effort for software maintenance as a crucial challenge. Of course this might have other reasons but it

is also a fact that due to unsystematic reuse the maintenance of software gets more and more difficult with an increasing number of software releases and software versions.

With the support of systematic reuse approaches such as component-based software development or software-product lines it is possible to overcome this issue and to make software reuse a real benefit for the company. Thus, manufactures should investigate in which way such approaches could be applied in their context and help to develop high quality software in reduced time. A pre-requisite for component-based development and software-product lines is that the software architecture is explicitly defined and specified. This, in turn, helps to improve the maintainability of the software as the impact of changes can be analyzed more easily and efficiently. Again, it is important to ensure that concepts for systematic reuse are scalable to different sizes of companies and software development teams.

Finally, the survey results also indicate that a quite high amount of companies integrate third party software in their products. Testing is the most important technique to ensure the quality of the products. However, it seems that manufacturers perceive a need for more rigorous quality assurance activities (58%). Thus, efficient processes, methods, and techniques to judge the quality of delivered software components would be beneficial for manufacturers to ensure high quality of their products.

In summary, the results of the survey indicate the following findings and therefore request the following activities:

- Systematic software engineering processes, methods and techniques do not seem to be used as much as would be possible. Consequently, software development might be improved by the application of standard software engineering concepts
- Requirements and usability issues seem to be the most challenging aspects in software development.

Consequently, method providers, such as research institutes should provide efficient techniques and methods, tailored to the medical device domain, to elicit, specify, manage and test requirements and usability aspects.

- Software reuse seems to be a promising concept to ensure high software development performance. Consequently, manufacturers should consider the introduction and usage of systematic reuse approaches such as component based software development and software product lines. This includes systematic approaches to specify and define software architectures.
- Software development teams are often of a small size. Consequently, software engineering concepts must be flexible and scale for different development team sizes
- Formalized techniques and methods are used infrequently in all development phases. Manufacturers should consider whether more formalized methods could contribute to addressing some of the challenges they face. The most important objective is to identify the right degree of formality for the type of product developed
- Software safety analyses seem to be not frequently applied, consequently efficient technique to perform these analyses are requested from research
- Software processes do not seem to be defined and standardized to a large extent in the domain. Consequently, more detailed and guiding standards on how to perform software development for medical device software should be defined and established in the future

Beside these findings the following aspects need to be considered when developing solutions according to the findings discussed above. These aspects are derived from recent trend-studies in the medical device domain

- Software architectures of the devices need to support interconnectivity and interoperability of devices
- Diagnosis of medical device networks need to be supported by embedded software elements
- Software elements must ensure the safety of stand alone devices and networks of devices, at best with concepts such as safety by construction
- E-Health applications need to be supported by medical device software (as more and more care is shifted from hospitals to the private home)
- Software techniques, methods and processes must be conformant with international regulations and standards
- Tools are needed that ensure (semi-automated) the standard conformance of the (software) development processes

6 Conclusion and Future Steps

In this paper we report on the findings of an internationally conducted survey focusing on software engineering techniques used in the medical device industry. The survey was conducted in 2006 and describes the current state of the practice in more than 90 companies throughout Europe, Scandinavia and the United States of America.

The data indicate that software is an integral part of medical devices - often realizing safety critical functions. In counterpoint to this indication, our survey also found a lack of usage of formal techniques throughout the different development phases and activities. On one hand developers often have to rely on informal– and thereby highly ambiguous–descriptions of requirements and designs in natural language. On the other hand the importance of good software developing standards is acknowledged as an important factor through out the industry.

This seems to be a contradiction; however, it can be explained by the fact that development standards are often required from external sources, such as certification agencies.

A better understanding is needed that even small sized development teams, which seem to be prominent in this domain, can benefit from systematic development process, methods, and techniques. This may indicate a need for additional training and education programs for both management and developers, tailored to the medical software domain.

The low utilization of tool support may be caused by a lack of adequate tools that meet the medical device industry's specific needs. A detailed survey on this topic might uncover some interesting aspects of why current tools are not more widely used, which might help focus future research and development efforts in this area.

Finally, it would be interesting to replicate at least parts of the survey in other countries such as Australia, Asia or South America. This would indicate if our results hold true just for the companies included in this US- and European-centered study or throughout the world. Especially in today's global economy, this indication would be a useful input for future research and process improvement efforts.

In addition, as with all studies and surveys, a replication of the survey would increase the trustability in our results.

References

[Adva04] Future Trends in Medical Device Innovation; Advanced Medical Technologie Association (AdvaMed) 2004. http://advamed.org

[Arte05] Artemis Strategic Research Agenda, Short Version, June 2005.

[BCR94] V.R. Basili, G. Caldiera, H. Rombach: Goal question metric paradigm,in Encyclopedia of Software Engineering, 1994.

[BDI05] Wirtschaftsfaktor Gesundheit: Chancen und Potentziale für Deutschland. BDI-Initiative Vitale Gesellschaft, Bundesverband der Deutschen Industrie, 2005, http://www.vitalegesellschaft.de

[BMB05] Studie zur Situation der Medizintechnik in Deutschland im internationalen Vergleich; Bundesministerium für Bildung und Forschung; http://www.bmbf.de

[CDRH02] CDRH, General Principles of Software Validation; Final Guidance for Industry and FDA Staff. 2002, (see also http://www.fda.gov/cdrh/comp/guidance/938.html)

[HCM06] Lee, I. Pappas, G.J. Cleaveland, R. Hatcliff, J. Krogh, B.H. Lee, P. Rubin, H. Sha, L. High-confidence medical device software andsystems, IEEE Computer Volume: 39, Issue: 4, 2006

[IEC06] Medical Device Software – Software Life Cycle Processes, IEC 62304 Ed. 1.0 b, 2006

[ISO00] ISO 14971:2000(E): Medical Devices - Application of risk management to medical devices. Genf, 2000.

[IEC00] IEC 60601-1-4, Medical electrical equipment - Part 1-4: General requirements for safety - Collateral Standard: Programmable electrical medical systems. Ed. 1.1, 2000.

[ITEA05] ITEA 2 Blue Book, September 2005, www.itea-office.org

[Rob02] C. Robson, Real World Research, 2:nd ed., Blackwell, 2002.

[SB99] Solingen, van R., Berghout, E.; The goal/question/metric method, a practical method for quality improvement of software development, McGraw-Hill ISBN 007-709553-7, 1999

[SC00] Siegel, S., and Castellan, J.N., Nonparametric Statistics for the Behavioral Sciences, Second Edition, McGraw-Hill, 2000.

[WK01] Dolores R. Wallace and D. Richard Kuhn: Failure Modes in Medical Device Software: An analysis of 15 years of recall data. International Journal of Reliability, Quality and Safety, vol. 8, no 4, 2001

Paper II

Development of Software for Safety Critical Medical Devices – an Interview-Based Survey of State of Practice

Christin Lindholm, Martin Höst

In proceedings of the eighth Conference on Software Engineering Research and Practice in Sweden (SERPS´08), Karlskrona, Sweden, November 2008.

Abstract

To be able to survive in the long run the medical device industry of today needs effective development processes and ways to secure quality. These development processes and quality assurance processes must follow the different laws and regulations over the world depending on what market the organisations are established on. Organisations have been developing medical devices and systems over many years but now this type of products contain more and more software. The development of software is often appended in to the existing development and quality assurance processes and these processes may not be the most efficient and correct processes when it comes to software.

This paper presents the results from an interview study with the purpose to survey how the medical device companies work today, what development processes and quality assurance techniques they use and how laws and regulations affect their way of working. Safety is very essential for the medical device organisations and all the interviewed organisations consider the

software in their medical device as safety critical. Risk and risk analysis is an important part of the safety thinking and is frequently performed by the organisations. However established and systematic techniques to analyse risks of the medical devices are not so frequently used as expected.

The intension is that the results from the study could be used as a help to find more adapted processes and techniques for software development in the medical device domain. The results have also been used to derive a set of requirements on new techniques and methods in the area. The derived requirements can serve as guidance to researchers aiming at improving processes, methods and techniques in the medical device domain.

1 Introduction

Quality requirements on medical systems and devices are high. If they do not work as intended, e.g. because of errors committed in the development process, it may result in threatening of human lives. The high requirements in combination with the high complexity of this kind of systems make quality assurance procedures during development crucial.

A large and growing share of the development effort of this kind of systems is devoted to development of software. An increasing part of the functionality is implemented in software and many new features of these systems would not be possible to implement without software. This means that the requirements of the software development process are as high as for development of other parts of the systems. Important quality attributes of software include, for example, inclusion of correct functionality, reliability with respect to fault content, usability for all users, and maintainability for software engineers in continued evolution of the product [1]. A failure to comply with the high requirements on any of these quality aspects may in time result in serious failures in operation.

Development of software differs to some extent to development of other engineering domains [2]. Software is abstract and "intangible" for managers and others which means that it is hard to envision the current quality, e.g. during development and testing. Software is also easier to change than many other entities. This gives, of course, flexibility during development, but it also puts high requirements on quality assurance during development. Software is also of very high complexity and it is hard to develop fault free software in general. This means that it is an important aspect in development of medical devices where software is only one part of the product, and where there are high quality requirements.

The software that runs a medical device or affects the use of a medical device automatically belongs to the same safety classification as the medical device [3] and has to follow the same laws and regulations as the rest of the medical device. It is important to notice that it is the manufacturer's purpose and the operation of the product that decides if the product is classified as a medical device, not the designer or the user. The laws and regulations state that the medical device organisations must have quality systems and that the quality system and the quality improvement actions must be documented. The quality system must cover the whole development process including the software development process and focus on the aspects and requirements to produce and provide safe and effective devices. The typical procedure for quality assurances of software is through the application of a structured development process (e.g. as described in [4]). Due to the high requirement, e.g. on safety of medical devices, it is of interest to investigate what quality assurance procedures, development processes that are used in development of software for medical devices. It is also of interest to investigate closer what the driving sources are for quality assurance and process improvement in the area.

The outline of the paper is as follows. In Section 2 background and related work with definition of medical device and a short description of laws and regulations in the area. In Section 3 the research questions and the research method are presented. The results are presented in Section 4, discussed in Section 5 and in Section 6 requirements on processes and methods for the medical device domain is presented. Finally, the conclusions are presented in Section 7.

2 Background and related work

The term "medical device" is defined according to law in many countries. For example in the Swedish law (1993:584) [5] about medical devices the term is defined with the following definition: "Medical device" means any instrument, apparatus, appliance, material or other article, whether used alone or in combination, including the software necessary for its proper application intended by the manufacturer to be used for human beings for the purpose of:

- *diagnosis, prevention, monitoring, treatment or alleviation of disease,*
- *diagnosis, monitoring, treatment, alleviation of or compensation for an injury or handicap, investigation, replacement or modification of the anatomy or of a physiological process*
- *control of conception*

Medical devices for the European market are regulated by Council Directive 93/42/EEC concerning medical devices (MDD) [3]. In the US the regulatory body of the Food and Drug Administration (FDA) [6] must approve medical devices. A medical device has to go through one or two evaluation processes, premarket notification (510(k)) or premarket approval (PMA) [7]. Every Member state in the EU must adopt and publish laws, regulations and administrative provisions to

implement the Directive [3]. There are some variations in national requirements, most of these concerns the need to notify the Competent Authorities when medical devices are placed on the market in their countries. There are different laws, legislations and a duplication of registration procedures for a medical device placed on the US market and the European market even if it is the same medical device.

According to the MDD [3], medical devices in EU are divided into four classes Class I, IIa, IIb and III based on the level of control necessary to assure safety and effectiveness. Class III is reserved for the most critical devices. The classification in the U.S. differs and they have three different classes, named FDA Class I, FDA Class II and FDA Class III. The software that runs a medical device or affects the use of a device for example surveillance the medical device automatically belongs to the same class as the device. The classification are build up on the risks the human body can be exposed to due to the design, the use or the mode of manufacture of the medical device. It is assigned to the manufacturers, based on the regulations to establish in which class the medical device belongs and after that establish which procedure to apply to ensure that all the demands in the regulations are met. The manufacturer carries out the classification of the devices, possibly in cooperation with a Notified Body (third part assessment).

Medical healthcare is one of the traditional areas considered as safety critical according to Knight [8] and he defines safety criticality as “Safety-critical systems are those systems whose failure could result in loss of life, significant property damage or damage to the environment”. Embedded systems have increasingly become predominant in a rage of safety critical applications for example in medicine, nuclear power plants, aviation and aerospace industries [9]. According to Hewett and Seker [9] other safety critical industries as well as medical device industries mandate certification for the code and its development

process to assure quality of the system. The certification process is a highly labour activity and the cost for developing a safety critical software system is reported by Nilsen [10] to be 20 to 30 times the cost of developing typical management information software.

Risk management is according to Doernemann [11] highly accepted in safety critical industries as for example aerospace and healthcare but more and more branches see the value or establishing risk management processes. In a recent article Rakitin [12] states that it is the medical device companies that must show that their software is safe and efficient. He means that for the companies to meet these responsibilities it is required of the companies to have expertise in effective risk management practices, to be familiar with software safety and to be able to adopt risk management mind set. Prakash et al [13] have examined requirements engineering process practices in three multinational pharmaceutical and healthcare companies and found large differences in the processes used between the companies in the development and that none of the projects followed the recommended best practice

How FDA´s laws and regulations can affect the development of software for medical devices are for example discussed by Branningan [14] where the effects of the "Safe medical device act of 1990" (replaced by Federal Food, Drug and Cosmetic Act in 1995) on non-embedded software is discussed. To the best of our knowledge the effects of European or Swedish laws and regulations on the development of software for medical devices has not prior been systematically analysed so in 2006 a survey was done by the authors of this article together with the Fraunhofer Institute for Experimental Software Engineering in Germany and the Fraunhofer Center in Maryland, USA [15]. The main objective of the survey was to characterise the state of the practice of software development in the context of medical devices. The survey was carried out through a web-designed

questionnaire and the study presented in this paper is based on interviews with special focus in special areas such as quality, standards and risk analysis.

3 Interview research methods

This section describes the interview study, the objectives for the study, the interview planning, the operation as well as the analysis and validity threats.

3.1 Objective

The objective of the research presented in this paper is to try to investigate how the medical device companies works today, what development processes and quality assurance techniques they use and how laws and regulations affect their way of working. More specifically, the objectives are as follows:

- To examine what type of products the organisations develop and for what market. This question is meant to provide background knowledge that is important when the answers to the other questions are interpreted.
- To understand the role of software in medical device and to investigate to what extent the organisations regards and treats it as safety critical.
- To investigate what standards and techniques that is used by organisations that develop safety critical medical devices and how laws and regulations affect the work.
- To investigate how the organisations guarantee the quality of the software in the medical devices.
- To investigate how requirements are handled and the use of risk analyses.
- To derive requirements that can serve as guidance to researchers aiming to improve processes, methods and techniques in the medical device area

The objectives are investigated in an interview study containing interviews with eight development sites in Sweden. The last

objective arises during the analysis of the interview answers when the need of requirements on techniques, methods and processes was identified based on the answers.

3.2 Method

The research in this interview study can be described as flexible according to Robson [16]. Flexible design allows the high-level research questions to be specified in advance but it also allows the study to develop. With a flexible design, a common way of collecting data is to carry out interviews. According to Lethbridge [17] interviews are inquisitive first-degree (direct involvement of software engineers) techniques that allow the researcher to obtain a general understanding of the software engineering process. Since the overall objectives for this study is to get a good general understanding of the role of software and software engineering processes in the medical device industry are interviews suitable form for the study. They are flexible and allow the researcher to clarify questions. Another advantage is that people are familiar with answering questions and often the participants enjoy the opportunity to answerer questions about their work.

The interview questions are open-end questions written to cover the objectives of this study and the interviews are based on an open dialogue between the researcher and the respondent. Each interview took between half an hour and forty-five minutes depending o the extent of the answers from the respondent.
There are twelve main questions areas the interview questions try to cover and these question areas are:

1. **The organisations.** Information about the organisations' background and products.
2. **Software**. Information about the use of software, safety criticality, development process, platforms etc

3. **Quality and standards**. Information about quality systems, quality assurance, use of standards, reviews, test etc.
4. **Law and regulations** Information about the laws and regulations the companies has to adjust to, classification and CE-mark.
5. **Requirements and risk analysis**. Information about the requirement process and risk analysis.
6. **Challenges** pointed out by the persons interviewed.
7. **Problems** described by the interviewed persons
8. **Verification** how it is done for the whole product and the verification of the safety critical parts.
9. **Statements**, interesting statements connected to some of the questions from the interviewed persons.
10. **Validation**, how the validation process looks like for the different companies.
11. **Traceability**, how requirements are traced during the development process.
12. **Observation**, cause and effect expressed during the interviews.

These question areas are the same areas as the twelve main categories used in the analysis phase.

The interview question document has been updated over time. The first version of questions was put to the three first interviewed organisations and then some questions was removed and added according to Table 1. Removed and added question on subject relate to what subject the question covered.

Table 1. History of interview questions

Question document	Nr of org.	Removed question on subject	Added question on subject
1:st version	3		
2:nd version	1	Product year?	Dev. process Risk analysis Clinical test
3:rd version	4		ISO 13485 standard

After the interviews, the material was transcribed and pieces of text were labelled with predefined factors. The text pieces were sorted after predefined factor and codes (keywords) were derived from the text according to each factor. A factor can consist of several codes for example factor Standard consist of the codes that are names of the different used standards ISO 9001, ISO 13485 etc. The material was then analysed by two researchers described in chapter 3.4

When the analysis of the interview answers was conducted a need for requirements on techniques, methods and processes was identified. The requirements were identified in three different areas: a) process, b) quality system and c) validation, methods, techniques. The requirements were identified in the interview answers, then specified and the cause for the different requirements was explained. After the specification of the requirements was conducted, the requirements were checked against the interview answers in order to assure that all requirements are grounded in the collected data. The requirements are presented in section 6.

3.3 Interview subjects and context

This interview study contains interviews with eight development sites in Sweden. The organisations were chosen in order to obtain valid sample and geographical vicinity. One of the organisations' devices does not contain software but is used

for comparison, to investigate similarities or differences. Many safety critical medical devices on the Swedish market are not developed in Sweden, just manufactured and this limits the suitable selection of organisations for the interview study.

The preparation of the interview questions was made by the researcher (the first authors of this article) with the intention to cover the objectives of the study. According to Robson [16] there are different types and styles of interviews and a commonly made distinction is based on the degree of structure or standardisation of the interview. The three types are fully structured interview, semi-structured interview and unstructured interview. In this case the semi-structured interview form was chosen, considered a suitable form since it is an early study and a respondent interview study. It is important to be able to update the interview questions according to the interviews and to get flexibility. A semi-structured interview has predetermined questions but the order of the questions can be modified based on what is most appropriate during the interview. It is also possible to add more questions, omit inappropriate questions, change question wording and give explanations.

All interviews were face-to-face interviews carried out in Swedish by the same interviewer. One person was interviewed from each organisation and it was one person interviewed at the time. All the interviews were recorded and then transferred to computer. The technique provides a permanent record and allows the interviewer to fully concentrate on the interview. The interviews were held at the respective organisation and the persons that were interviewed worked as quality assurance manager, clinical affairs manager, strategy manager, development or technical manager. The interview questions were updated twice (see Table 1) but there were no significant changes made to the original questions. However, a couple of new questions were added about development processes, risk analysis and clinical test, and one question about the product was removed.

3.4 Interview analysis

Data reduction is a part of the analysis and denotes a systematic way of selecting information for the continued analysis and also simplifies and abstract raw data. The next step is to find summarising word or symbols for a segment of words and in some way code the material without the sense getting lost. The interviews were then fully transcribed to text format before the analysis was done. The researchers specified thirty-four predefined factors before data was collected from the interviews. The predefined factors were derived from the interview questions and were for example development process, class and standard. The data collected from the interviews was then reduced to remove irrelevant information and the text was labelled with the predefined factors. One of the interviews was labelled with the predefined factors individually by the two authors of this article and then the result was compared to see that the labelling not diverted too much which it did not do. The predefined factors were organised in twelve main categories to systemise the factors so a category contain several factors. These twelve main categories are as mentioned the same as the twelve question areas presented in section 3.3.

The factors and codes were then put together in a matrix. The matrix was constructed with the predefined factors in the column (in Table 2. Dev. Process, Class and Standard) and one row containing codes (in Table 2. for example V-model, III, ISO 9001) for each interviewed subject.

Table 2. Example of matrix data

Org.	Dev. Process	Class	Standard
1	V-model	III	ISO 9001
2	Own model similar V-model	IIb	ISO 13485

The constructing of matrix was done the first time after the first four interviews and the second time was after all eight interviews were made. The second round of factor was similar to the first round but was extended with some more factors to make sure that no meaningful material was overlooked.
The matrix was then analysed and discussed by the two researchers, and the results for each factor in the matrix were written down. The codes and parts of the interviews were reviewed again before conclusions were drawn.

3.5 Validity

Validity can according to Yin [18] be classified in construct validity, internal validity, external validity and reliability. Construct validity is affected by how correct the colleted operational measures represent the concepts studied by the researcher. There is a risk that the interviewer and the interviewee interpret terms or concepts different. To reduce this risk, the interviewer explained concepts as for example “quality plan” and “inspections” during the interview. Another risk in this study is that the interviews were only preformed by one researcher but this risk was reduced since all the interviews were recoded.

Internal validity is affected by factors that are outside the control of the researcher but affects the measures. A threat to this study could be to establish incorrect causal relationships when we analyse relations between different interview responses. However, in this study no conclusions about causal relationships are drawn, only relationships between factors were analysed.

External validity concerns the problem of how general findings are with respect to the subject population and beyond the immediate study. The result from this study is based on interviews with a limited numbers of subjects from a limited

number of organisations, so it should be regarded upon as an exploratory study and further studies are needed in this area.

The validity is also affected by the reliability, how well described procedures are followed and documented so that the study can be repeated in the same way again. The goal is also to minimize the errors and biases in a study. In this study we have tried to minimize threats by recording the interviews and then fully transcribe them. The procedures and all changes to the study over time have been closely documented in a special document so that the study procedure can be reflected on and repeated in the same way again. In order to reduce researcher bias, one of the interviews was categorised and classified by two researchers individually in parallel, and the results were compared to verify that the results did not differ too much. The analysis of the results was also made the two researchers. A threat in this study can be participant's bias, if the interviewees have deliberately answered incorrectly, for example to given a more positive picture of their way of working or their organisation.

4 Results

4.1 The background of the organisations

Eight organisations took part in this interview study. Four of the organisations are multinational companies with a branch of the organisations in Sweden and the rest of the organisations are located only in Sweden. The organisations vary in size from a couple of hundred to a couple of thousand employees. The medical devices are mainly embedded, real time systems containing software. The devices supplied are various surgical equipment, equipment for microwave thermotherapy, analytic instruments, cardiac and respiratory equipment, sterilisation equipment and modifications of patient management systems. For all the medical device systems, they are indented for

continuous use for not more than 30 days per occasion and so according to MDD [3] definitions they are short-term medical devices. The medical devices are mainly used by experienced personal that frequently use the medical devices but have no deeper technical knowledge. The users are mainly physicians but in most case (six out of eight) other personnel in the health care sector, e.g. nurses are also users of the medical devices. The organisations' main customers are hospitals and some private medical clinics all over the world. In most cases the devices are procured by departments with procurement responsibilities, which means that the customer are often not the same as the users of the medical devices.

The development processes used by the organisations differ. The answers given of the organisations are presented in Table 3 where "Org." represents the eight different organisations and "A. Dev. Process whole product" is the development the organisation states that they use for the development process for the whole product and "B. Dev. Process software is the development process" stated for the development of the software.

Table 3. Development processes

Org.	A. Dev. Process whole product	B. Dev. Process software
1	V-model	CAPA process
2	V-model	V-model
3	Design and control	Design and control
4	Own model - similar to V-model	QSR quality system
5	V-model (modified)	No software
6	GAMP4	GAMP4
7	*No answer*	*No answer*
8	*No answer*	*No answer*

The V-model mentioned as a modified V-model is a variant of the basic V-model with product specifications and standards for type tests and environment tests influenced by the Swedish defence industry. One of the respondents states the use of an own model but describes it very similar to the V-model. It can be noticed that three of the organisations have the same development process for both the software development process and the development process for the whole medical device.

Two of the respondents chose to not state their choice of development process at all and one of the organisations' devices does not contain software but is used in this study for comparison, to investigate similarities or differences, however in respect to development process, standards and quality assurance. No differences were how ever found according to organisations with medical devices containing software.

4.2 Software

All the organisations consider the software in their medical device a safety critical and this corresponds well to the classifications of the medical devices according to MDD [3]. The software that belongs to a medical device is classified in the same class as the medical device and based on the level of control necessary to assure safety and effectiveness a device is assigned to a regulatory class where devices in Class IIa has a high risk potential and Class IIb has an even higher risk potential (Class III are the class for the most critical devices). The software in this study is classified in Class IIa or Class IIb. Since the software belongs to the medical device it is also included in the CE labelling. All the organisations medical devices are labelled with the CE-mark, otherwise the organisations are not able to manufacture the medical devices on the EU market. The software in the study is used in different types of systems for example control systems, automation systems, safety systems

and information systems and the software is used for example for control, regulation, registration, navigation and protection. When it comes to developing the software three of the organisations develop their own software, one of the companies only uses software from suppliers and three of the companies both develop their own software and use software from suppliers.

It was difficult to get information from the interviewees about the programming languages and platforms used by the organisation. Either the interviewees did not possess the information or they did not want to share this type of information. However three organisations uses C++ but the same organisations also uses other programming languages for example C, C#, PCL. Three interviewees stated that their organisation uses PC platforms and there were no answer to this question by the rest. But some interesting remarks were made by the interviewees according to platforms and programming languages. One of the interviewees stated, “I think it will take a while before our customers accept PC based code” and another one of them means that C++ is on its way out and is replaced with C# and .NET platforms instead.

As presented in Table 3 the organisations follow different development processes for their software, and the processes stated are the V-model, Design Control, CAPA process (Corrective Action and Prevented Action), QSR quality system and GAMP4. The V-model is a traditional development process and one of the interviewees motivated the use of the process with “because it is easy to repeat and to describe”. GAMP4 is a guide for validation of automated systems and medical devices. It focuses on for example risk assessment, design reviews and traceability. CAPA is a process for existing products problems, customer complains etc and also a process for detecting potential problems. The process also includes risk assessment of the problems. Both FDA and ISO require an active CAPA

process as an essential element of a quality system. Quality system regulation (QSR) is an American law for medical devices and corresponds to the international quality standard ISO 13485, but they differ on a detailed level. Design Control is a major subsystem to QSR and its purpose is to assure that devices meet user needs, intended use and specified requirements. It focuses for example on design review, design verification and design validation. The similarities between the processes are that they are all managed processes with focus on risk assessment, validation and design reviews.

4.3 Quality and standards

All the interviewed organisations have quality systems, a system of regulations and methods for how the work with quality assurance and quality management is carried out in an organisation. The laws and regulations state that the medical device organisations must document their quality systems and also document all the quality improvements made over time. The quality system must cover the whole development process and focus on the aspects and requirements to produce and provide safe and effective devices. To be able to CE-mark a device it is also required of the organisations to have some form of quality management system.

The organisations follow different standards and an organisation can often follow several different standards. The majority of the organisations in the study have stated that they follow more than one standard and this explains total number of organisations in Figure 1.

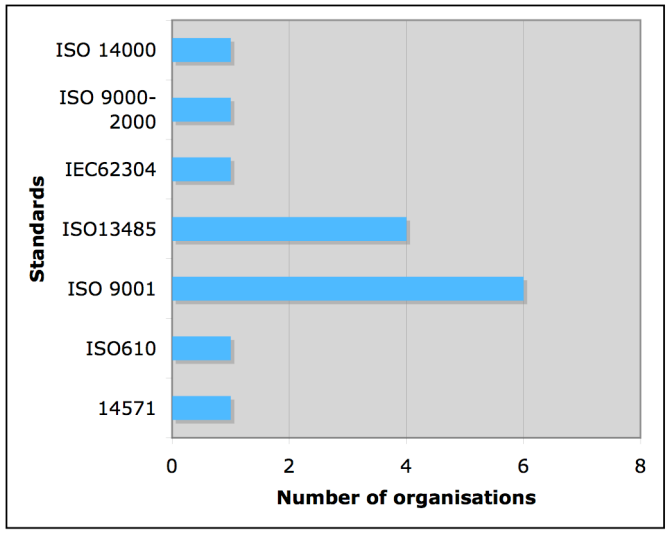


Figure 1. Standards in use

As shown in Figure 1 the dominant standards used are ISO 9001 and ISO 13485. ISO 9001 is a quality system standard applicable to many kinds of industries where as ISO 13485 is a standard specific to medical device quality systems, which is especially developed to harmonise with constitutional requirements and is also a supplement to the ISO 9001 standard. Some of the additional requirements in ISO 13485 compared to ISO 9001 relate to design controls, process controls, traceability and regulatory actions, which are more critical for the medical device industry. All the organisations work with quality in some way, for example with quality plans, manuals and quality systems. According to standards, the organisations have to work with quality improvement and focusing on the customers, which means that management shall guarantee that the customers' requirements are established and fulfilled. One of the organisations with medical devices on the US market is working according to Quality system regulation and this process covers design, production, servicing and corrective/preventive (CAPA) activities.

That the organisations according to the standards must focus on the customer is something the organisations are well aware about. They work, for example, with user feedback, measuring change orders, forms for quality remarks and complaints, post market meetings, and incident handling. Looking at different quality assurance techniques, the organisations have, for example, different kinds of internal inspections. Most common are reviews of requirements specifications, design reviews and code reviews. Similarities found are that the same person who has written the documents or the code does not make the reviews. The reviews are usually carried out without tool support. One of the interviewees states that reviews are "a cultural matter and are experienced as trespassing on the personal work". According to ISO 13485 and ISO 9001 the organisation must review the product requirements before they make a commitment to provide the product to the customer. American law states that design reviews must be conducted and in the FDA guidelines for software code reviews are recommended.

The organisations' quality systems and quality improvements documentation are inspected at external inspections conducted by a Notified Body (third part). All the organisations develop devices that are classified in Class IIa or IIb, so a Notified Body must assess them. Three of the organisations also have external inspections carried out by experts from FDA.

The organisations state that they do verification, for example verification of requirements, verification against standards, pilot study, or evaluation of performance. One organisation also describes that they start with code review, then conduct unit testing, followed by integration testing, and last system testing. According to standards and regulations, verification shall be conducted in a way that the organisation secure that the requirements are fulfilled and so the documentation of the verification is preserved.

When it comes to validation it is also regulated in standards and by FDA´s Quality system regulation and guidelines. The validation shall secure that devices fulfil the requirements for

intended use. As a part of the validation of the development results shall the organisation do clinical evaluation and/or evaluation of performance. Also the documentation of the validation must be preserved and validation must be done before delivery or use of the device. Four organisations state that they have validation, the rest of the organisations ought to have validations according to regulations but they chose to not comment on the questions about validation given during the interview. One of the organisations has usability studies, a special usability group and a special validation group. They also attempt to use human factors and to get early feedback from the end users.

4.4 Requirement engineering and risk analysis

Requirement engineering is an important area for the organisations and the sources where the requirements are collected from are standards, laws, users, vendors and sales departments. A problem mentioned by an interviewee is that "we engineers do not always understand what the sales-department means and they do not always understand what we mean". It is stated in standards and regulations that the organisations must establish requirements from customers, including requirements concerning delivery. The organisations must also establish that the constitutional requirements for the device are covered in the requirement specification.
The organisations using the V-model state the requirement engineering is carried out as part of the model, starting out with a requirements specification with user and marketing requirements. This specification is then broken down in one or several technical requirements specifications. A few of the organisations treat safety critical requirements different than non-safety critical requirements ad this means that safety critical requirements are traced more thoroughly though the development process, from requirement to design to code to verification and validation and backwards.

Top priority for all organisations is the safety of the medical devices. All the organisations state that they make risk analysis on a regular basis and this is exactly according to the regulations which state that such a process must be executed and the results must be documented. The organisations do the risk analysis in different areas such as development, production, problems, users and risk analysis if changes are made. One of the organisations follows for example ISO 14971. This standard describes that continual control of the final residual risks shall be done. The risk is evaluated, the risk level is updated and corrective measures are taken to reduce the remaining risk. Risk management reviews are also done where all available data about the risks are collected to control the risk level. Risk analysis can be performed by using different techniques such as for example HazOp [19], Failure Modes and Effects Analysis (FMEA) [20] and Fault Tree Analysis (FTA) [21]. These risk analysis techniques are suitable for identifying risk during development of medical devices and are recommended by different standards. HazOp and FTA are more suitable for systems and software whereas FMEA is more suitable for components. Two of the organisations state that they use FMEA and one that they use FTA. The other organisations do risk analysis, but they have not specified any special risk analysis technique. One of the organisations, however, mentioned that it has worked with so called “emergency plans”, ready to use if a risk should appear.

5 Discussion

For many of the organisations it is seen as a problem that the laws and regulations are different in different parts of the world. It had for example been an advantage for all organisations of medical devices if the laws and regulations were the same over the whole world and no duplication of registration procedures and many external inspections of different third part were

needed. The Global Harmonization Task Force[2] (GHTF) is a voluntary group of representatives from national medical device regulatory authorities and the regulated industry working towards harmonisation in medical device regulations.

Laws, regulations and standards affect the organisations' way of working and the processes used are at a large extent managed. The development of software is often appended in to the existing development and quality assurance processes and these processes may not be the most efficient and right processes when it comes to software. Maybe the organisations should benefit more from tailor made development processes for the medical device domain with focus on correct functionality, reliability, safety, risk assessment usability and other important areas for the domain.

The dominant standards used in the interviewed organisations are the ISO 9001 and ISO 13485. According to the survey focusing on software engineering techniques used in medical device industry [15] ISO 13485 is also stated as one the most frequent used (53%) standard. A probable reason for this is that the standard is specific for medical device quality systems and it is also especially produced to harmonise with constitutional requirements. Maybe more organisations should be guided and helped to change to ISO 13485 and this standard should be well incorporated in tailor made development processes.

Quality assurance is one of the major areas that are dealt with in laws and standards and in the context of quality assurance it was found that inspections are frequently performed but it is was not cleared during the interview how they were done (checklists, reading techniques, ad hoc etc). It should be possible to find and recommend quality assurance techniques that are especially suitable for the medical device organisations. When it comes to risk analysis it is also frequently performed by the organisations however some of the organisations do not use established and systematic techniques as HazOp or FMEA to analyse the risks

[2] http://www.ghtf.org/

of the medical devices. A reason for that is may be that is too much effort to use these techniques in relation to the safety risk. It was not mentioned during the interviews how the manufacturers actually prove that their medical devices are safe. It would be interesting to investigate the real reason why systematic techniques are not used and based on the results maybe tailor make a technique that fits the organisation's way of working, are easy to use, cost effective and adapted to what are requested by laws and regulation.

The interviewed persons stated that the software is safety critical work on management level in the organisations as for example quality assurance manager, clinical affairs manager, strategy manager, development or technical manager and they are well aware of the classification of the medical device, maybe the developers have another opinion regarding the software. In the previously conducted survey [15] there were participants that did not consider their medical device as safety critical even if the classification indicated the opposite. The reason can be that the participant's apprehend the part of the medical device he/she is working with not to be safety critical and this maybe effect the way of working.

Only one of the organisations procure all their software from third part and one organisation pointed out that use of third part software increases and this is also a trend seen among developing organisations in other areas [22]. The question is how the use of third part software affects the organisations way of working with quality assurance, can the organisations guarantee that laws, regulations, standards and work procedures are followed, how the inspections shall be done and who has the responsibility.

6 Requirments on developed thechniques and processes

Software process improvement is, as in other fields, important in development of medical systems. However, there are a number of important issues to think about when new

procedures, processes, techniques and methods are developed in every domain. Based on Software process improvement is, as in other fields, important in development of medical systems. However, there are a number of important issues to think about when new procedures, processes, techniques and methods are developed in every domain. Based on the findings from the interview studies we have identified a number of requirements on techniques and processes, which are intended to be used in software development of medical systems. The requirements are presented in table 5-6 below. These requirements can serve as guidance to, for example, researchers aiming at developing methods that are used in this domain.

The requirements are divided into requirements on process level, requirements on quality systems and requirements on individual techniques.

The development process and other processes over time for example the document process; quality assurance process and validation process must fulfil these two key requirements for medical device organisations found in Table 4.

Table 4. Requirements for processes

PROCESS	
Requirements	**Cause**
1) Must fulfil laws in different countries	Medical devices that are marketed in several countries have to be inspected and obey the law in these different countries
2) Must be designed to be able to fulfil several different standards	A medical device organisation are often certified according to several different standards

The two requirements in Table 4 have been derived from the interviews in this study where the organisations state that they have to follow different laws according to the different countries they are marketed in and that they are certified according to

several different standards. Most of the interviewed medical device organisations state that they are certified according to three to five different standards. Since these two requirements are key requirements they will also be found in Table 5 and 6.

Many of the development processes used by the interviewed medical device organisation focus on quality, risk, validation traceability and design control. These areas can be found in Table 5 and Table 6 as requirements if for example a new quality system, validations process or a new method or technique should be developed. Example of new methods or techniques could be a new quality assurance method or a new risk analysis technique.

Table 5. Requirements for quality system

QUALITY SYSTEM	
Requirements	**Cause**
1) Must fulfil laws in different countries	Medical devices that are marketed in several countries have to be inspected and obey the law in these different countries
2) Must be designed to be able to fulfil several different standards	A medical device organisation are often certified according to several different standards
3) Must cover the whole development process	The whole development process must be covered by the quality system according to law
4) Must be documented	All the quality assurance activities must be documented according to law
5) Quality improvements over time must be documented	According to law and standards the medical device organisations must document all quality improvements.
6) Must have focus on producing safe and effective medical devices	Safety and efficiency are key areas for the medical device organisations
7) Must include design control	According to law the medical device organisations must have design control

	that are an interrelated set of practices and procedures that are incorporated into the design and development process
8) Must include process control	According to standards the medical device organisation must monitor, measure and analyse processes
9) Must secure that the customers requirements are established and fulfilled	According to standards it is the top management of the medical device organisations that have the responsibility to secure the customers requirements.
10) Must include procedures for risk analysis	All medical device organisations must perform risk analysis according to law
11) Must have traceability for example from requirements to design to development to product and backwards	All quality activities must be able to trace during to whole development process according to law.
12) Must be available in a inspectable format for third part	Most of the medical device organisations quality systems are inspected by Notify Body

The requirements in Table 5 and Table 6 are derived from both laws and regulations and from the interviews made by the medical device organisations. All the organisations pointed out that the laws and standards really affect their way of working very much. They have to have these standards and laws in mind in every thing they do.

In the interviews it appeared, for example, discussions about focus on customer, safety and risks. It was stated that the organisations focus more and more on the customers and users in different. This seems to be a relatively new issue for the organisations but an area they have to work with according to standards.

All the medical devices in the study are classified according to the MDD [3] and/or FDA [6] as safety critical and all the organisations state in the interviews that they consider the software in their medical devices as safety critical. All the organisation point out that they focus on safety and that safety is a very important area for this type of organisation.

All interviewed organisations do risk analysis on a regularly basis and also in this area different processes and standards effects the organisations way of working and this fact is very obvious to them.

Table 6. Requirements for validation, methods and techniques

VALIDATION, METHODS AND TECHNIQUES	
Requirements	**Cause**
1) Must fulfil laws in different countries	Medical devices that are marketed in several countries have to be inspected and obey the law in these different countries
2) Must be designed to be able to fulfil several different standards	A medical device organisation are often certified according to several different standards
3) Must be carried out before delivery or use	Validation and risk analysis must be done before delivery or use
4) Must be documented	All validation and use of methods and techniques must be documented according to law
5) Must have clinical evaluation and/or evaluation of performance for the whole product including software	All medical device organisations must evaluate performance and/or do clinical evaluation

Since systematic techniques and tool not seems to be used as widely in the domain as could be expected this requirements

above can hopefully be a help in the process of developing such techniques and tools.

7 Conclusion

There are no global laws and regulations so the developers and manufactures of medical devices have many different rules, laws, regulations and standards to adjust to depending on what market they are interested in. This leads for example to duplication of registration procedures, which takes a lot of effort, and that external inspections of more than one third part are needed. It had been facilitated for all developers and manufacturers of medical devices, if the laws and regulations were the same over the whole world and no duplication of registration procedures and many external inspections of different third part were needed.

The medical devices are often short term embedded systems, defined as normally indented for continuous use for not more than 30 days, this by MDD [3] labelled with the CE-mark and classified according to MDD in Class IIa or IIb. The developing organisations consider their software to be safety critical and this corresponds well to the classifications of the medical devices according to MDD. All the organisations follow development processes for their software; they have for example the V-model, and Design Control but it could be possible to design special development processes, especially adapted to handle the difficulties of developing medical devices and also a development process that fulfils the requirements from laws and regulations.

In accordance with available laws and regulations all organisations have quality systems. They are all certified according to ISO standards such as ISO 9001, ISO 13485. ISO 13485 is a standard specific to medical device quality systems where as ISO 9001 is quality system standards applicable to many kinds of industries. According to the standards the organisations have to work with quality improvement and focus

on the customers. The organisations do that by, for example, working with user feedback, classification of problems, and special forms for quality remarks and complaints. To guarantee the quality they have internal inspections of the requirements or the quality system. They have reviews of requirements specification, design reviews and reviews of their quality system. A Notified body also performs external inspections of the quality system documentation on regular bases. As part of the quality assessment process the organisations do risk analysis and they are oblige to do so according to law. The used risk analysis techniques differ between the organisation and the area in which the risk analysis is performed differs from organisation to organisations. The organisations are very eager to follow the law so risk analysis is frequently performed but it is a little bit surprising to notice that some or the organisations do not use established and systematic techniques as for example HazOp and FMEA to analyse the risk of the medical devices, not in that extent as it is expected.

This interview study is an exploratory study. The objective has been to try to get an understanding of the organisations developing safety critical medical devices containing software and their developing and quality processes. The medical device organisations have to focus more on safety and risks than other kind of industries and they are at a large extent managed by laws and regulations, this taken together makes the requirements on the development process for the medical devices more specific and more work can be done in this area to help the organisations.

Some areas of special interest have been found were improvements can be made. These findings could lead to further research in the quality, process and risk areas that in the end could give the medical device organisations specially adopted processes and techniques that further could lead to more cost effective work for the organisations. As a guide to e.g. researchers and based on the findings from the interview studies we have identified a number of requirements on processes and

techniques. The intension is that these requirements can serve as guidance to researchers aiming at developing methods and techniques that are used in this domain.

References

[1] ISO/IEC-9126-1 Software engineering— Product quality Part 1: Quality model, 2001.

[2] F. Brooks, No Silver Bullet Essence and Accidents of Software Engineering, IEEE Computer, Vol 20, No 4, 1987 pp 10-19.

[3] Commission of the European Communities, Council Directive 93/42/EEC EEC concerning medical devices.

[4] Software Engineering Institute, CMMI for Development, technical report CMU/SEI-2006-TR-008, ESC-TR-2006-008, 2005.

[5] Swedish Code of Statutes *The Act* (1993:584) Medical Devices. www.riskdagen.se (2008-09-16)

[6] U.S. Food and Drug Administration, Federal Food, Drug and Cosmetic Act section 201(h).

[7] U.S. Food and Drug Administration, 1995. Premarket Notification 510 (k), Regulatory Requirements for Medical Devices, HHS Publication, FDA 95-4158

[8] J. C. Knight, "Safety Critical systems: Challenge and Directions", In proceedings of the 24th International Conference on Software Engineering, 2002. ICSE 2002, IEEE Computer Society, 2002, pp 547-550.

[9] R.Hewett, R. Seker, "A Risk assessment Model of Embedded Software Systems", In proceedings of the 29th Annual IEEE/NASA Software Engineering Workshop, IEEE Publication, 2005

[10] K. Nilsen, “Stringent certification requirements for safety-critical software”, Embedded Control Europe, pp. 18-21. 2004

[11] H. Doernemann, “Tool-based risk management made practical”, In proceedings of the IEEE Joint International Conference on Requirements Engineering (RE´02), 2002.

Part C

Risk Management in the Medical Devices Domain

Paper III

Different Conceptions in Software Project Risk Assessment

Martin Höst, Christin Lindholm

In proceedings of the Software Engineering Track at the 22:nd Annual ACM Symposium on Applied Computing (SAC), Seoul, Korea, March 2007, pp. 1422-1426.

Abstract

During software project risk management, a number of decisions are taken based on discussions and subjective opinions about the importance of identified risks. In this paper, different people's opinions about the importance of identified risks are investigated in a controlled experiment through the use of utility functions. Engineering students participated as subjects in the experiment. Differences have been found with respect to the perceived impor-tance, although the experiment could not explain the differences based on undertaken role in a development course

1 Introduction

During project planning and management, procedures for risk management are crucial. This is, for example, acknowledged by the presence of risk management issues at level 3 in the Software Engineering Institute's Capability Maturity Model (e.g. [1]). The objective of risk management is to identify relevant risks as early as possible in a project, in order to avoid or limit the effect of potential problems, such as project delays and cost overruns. More formally, risk management can be defined as "an organized process for identifying and handling risk factors; including initial identification and handling of risk factors as well as continuous risk management [2]. Safety critical projects include, as all other projects, a large amount of software. When it comes to risks that are related to the product, e.g., the number of persistent faults in the product, they are very important for two reasons. One reason is that it is important to identify these as early as possible in order to secure the quality of the developed product. The second reason is that it is important to limit the number of problems during the project even if the quality of the product with respect to the number of dormant faults is acceptable when the product is delivered. This is because a large amount of changes during a project deteriorates the structure of the code, which results in new faults later on.

Risk management is often carried out in a number of steps, e.g.: risk identification, risk analysis, risk planning, and risk monitoring [5]. During risk identification, risks are identified by relevant people, e.g. by using checklists and brainstorming techniques. The identified risks are prioritized with respect to their probability of actually occurring in the project and their potential impact. The risks that are expected to have both high probability and large unwanted effects are the most important risks to continue to work with in the process. In the risk-planning step, plans are made in order to either lower the effects

of the prioritized risk, lower their probability, or to prepare for what to do if they actually occur. In the monitoring step, the risks are monitored during the course of the project. There are, of course, no clear and objective rules available for how to prioritize the identified risks in the second step. This is instead carried out through discussions and subjective evaluations, where participants have different values and see the risks in different ways [4]. This means that it is important to investigate how large differences there are between different participants, and whether it is possible to explain identified differences.

Utility functions (e.g. [7]) describe how different people value a property. For example, a utility function could describe how people value the expected life-duration after different alternative medical treatments. If the utility function is linear, a life-duration of $2x$ years would be perceived as twice as good as a life-duration of x years. The utility function does, however, not have to be linear, which affects how people make decisions when choosing different treatments. Based on the shape of the utility function it is possible to discuss whether different individuals act as risk-averse, i.e. they tend to avoid risks and choose a lower safe gain, or risk-seeking, i.e. seeking a possible high gain instead of a more certain lower gain.

2 The utility function

2.1 The Trade-off method

The objective of the Trade-off (TO) method is to estimate the utility function for one person. According to the TO method [7] the subject is iteratively asked to compare different "lotteries". A lottery is shown graphically in Figure 1, which should be interpreted as that one of two events (event 1 and event 2) will occur If the probability of event 1 is p, then the probability of event 2 is $1-p$. If event 1 occurs this will result in result 1 and if event 2 occurs this will result in result 2.

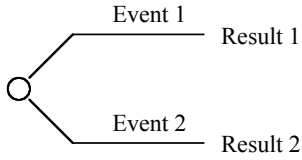


Figure1: A lottery

An example of possible values in the lottery is shown in 0.

Table1: An example of a lottery.

Property	**Meaning**
Event 1	Design expert NN is unable to follow the project
Event 2	Design expert NN is able to follow the project
Result 1	There will be 10 faults at the acceptance test
Result 2	There will be 3 faults at the acceptance test

In the TO method participants should iteratively compare pairs of lotteries. An example of a pair of lotteries is shown in Figure 2. The upper lottery shows what could happen if one condition is true (an old design is chosen) and the lower shows what could happen if another condition is true (a new design is chosen). The probabilities of the events are assumed to be independents of the conditions, i.e. the probability that design expert NN will be able to participate in the project is the same in the two lotteries. An advantage of the TO-method compared to other methods for eliciting utility functions is that the value of the probability need not be explained to the person using the method.

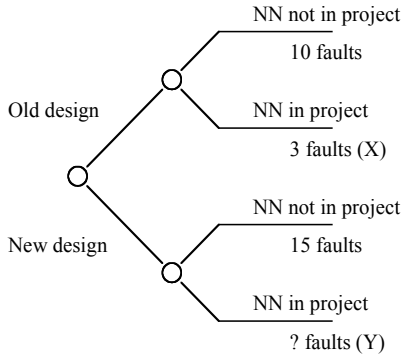


Figure 2: A pair of lotteries.

In the TO method the subject is first asked to select a value of the number of faults in acceptance test in the second lottery (Y in Figure 2) that makes the two lotteries equally attractable. When this has been done the subject is asked to compare two new lotteries. These two lotteries are similar to the first two lotteries, but with value X (see Figure 2) changed to the value that the subject chose in the last question. The subject is now asked to give a new value of Y that makes these two lotteries equally attractable. This process is iterated in order to give values of the utility function for the result factor.

If the X-value in the first comparison is called x_0, the first Y-value is called x_1, the second Y-value called x_2, etc., then it can be shown that the utility function u, can be estimated as [7]

$$u(x_i) = i \times u(x_1)$$

which can be normalized to $u(x_i) = i \times a$ where $a = 1/n$, and n is the number of Y-values given by the subject. The proof for this is not provided in this paper; instead the reader is referred to [7]. In Figure 3 a hypothetical example of a utility function is shown. This example shows a concave curve, i.e. $x_i - x_{i-1} < x_{i+1} - x_i$, $1<i<n$. Curves can also be linear ($x_i - x_{i-1} = x_{i+1} - x_i$), convex ($x_i - x_{i-1} > x_{i+1} - x_i$), or a combination of these shapes.

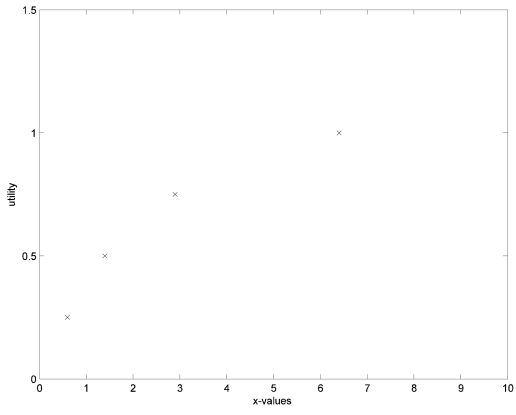


Figure 3: An example of a (concave) utility function.

2.2 Interpretation of utility functions

In most cases utility functions describe properties where a large number is better than a low number (e.g. monetary gain). The factors that are considered in software risk assessment often refer to negative aspects and not to positive aspects. For example, factors such as number of remaining faults, delay, etc. are analysed instead of positive factors such as revenue, life-duration, etc. In [3] the typical shape of utility functions for losses is discussed.

If the utility function e.g. for the remaining number of faults is concave (i.e. as in 0) this means that relatively the effect of every fault is higher if there are few faults than if there are many faults. This means that a person with this interpretation thinks that $2x$ faults is less than twice as serious than if there are x faults. If this person would choose between a fixed value x and a lottery with value 0 with probability 1/2 and value $2x$ with probability 1/2, this person would probably choose the lottery since the expected utility value of the lottery is lower than for the fixed value x. Since this person chooses the lottery instead of the fixed value, we say that a person with a concave utility function is risk seeking. If the function is convex, the value of

every fault is higher if there are many faults compared to if there are few faults. This means that a person with a convex utility function is risk averse.

Imagine a situation where a person should compare two different alternative ways of handling a risk in a project. Based on subjective evaluations it might be estimated that one of the alternatives will results in a certain expected number of remaining faults and the other alternative will results in a higher number of expected faults. In this case a person with a concave utility function would probably not see the second alternative as negative as a person with a convex utility function. This will of course affect how different people act during discussion on risk evaluation during risk management. It is therefore interesting to investigate how different individual utility functions for this type of properties are.

3 The experiment

3.1 Objectives

The objective of the research presented in this paper is to investigate the shape of utility functions for factors that are relevant in software project risk management. More specifically, the research questions are as follows:

- RQ1: What is the distribution between convex, concave and linear utility functions for properties that are relevant in software project risk assessment?
- RQ2: Is there any difference between different roles in a project with respect to the shape of the utility functions?
- RQ3: Is there any difference between the shapes of the utility functions for normal projects and projects developing safety-critical products.

3.2 Experiment subjects, objects, and context

The research questions are investigated in an experiment where students act as subjects. The experiment was conducted as part of a software engineering project course given at Lund University during the spring of 2005. The students followed programmes in Computer Science, Software Engineering, Electrical Engineering, and Multimedia. The course is attended in the 2:nd year of their university studies.

The course is a project-course where the students work in projects of typically 17 persons in each project. All projects are given the assignment of implementing a number of services for a basic telephone switching system. In the beginning of the course the students are given a basic version of the system where only basic functions such as providing simple telephone calls, managing what happens if the called party is already involved in a telephone call, etc. are provided. Their assignment is to develop more advanced services such as call forwarding, billing, etc. The project group should follow a software development process based on the waterfall model with steps such as project planning, requirements engineering, implementation, and testing. This experiment was conducted during the test-phase of the project, i.e. after the project planning was carried out. In every project groups the students are divided into the following roles: Project leaders (PL), Technical responsibility (TR), Developers (D), and Testers (T).

The experiment was conducted during a seminar where all students participated. At the seminar the seminar-leader first held a lecture on risk management, and then the students carried out the tasks of the experiment.

In the experiment the utility function of every student was elicited with the TO-method. The students were presented with two scenarios (scenario 1 and scenario 2). Scenario 1 is based on the project assignment in the course (translated from Swedish to English):

Assume that there was a design expert (NN) in your project that could decide the design. NN is part of the "technical responsibility"-group of your project and NN has some new ideas about the design that are not exactly as the teachers in the course have thought. The design proposed by NN is called "new design" and the ordinary design, as proposed by the teachers is called "old design". Based on experience data, the project leaders estimate that there will be a certain number of faults remaining in the product at the acceptance test.

Consider the following four cases:

Case 1A: The old design is used and NN is able to participate in the project. Then there will be 5 faults at the acceptance test.

Case 1B: The old design is used and NN is unable to participate in the project due to illness. Then there will be 6 faults at the acceptance test.

Case 2A: The new design is used and NN is able to participate in the project. Then there will be 2 faults at the acceptance test.

Case 2B: The new design is used and NN is unable to participate in the project due to illness. How many faults can there be at the acceptance meeting if the two designs should be equally attractable?
Scenario 2 is based on another system than they worked with in the course. It describes instead a safety critical system and was presented as follows (translated from Swedish to English):

In an intensive care unit you have surveillance equipment connected to the patient that monitors the patient condition. Different values is continuously registered, such as patient's absorption of oxygen, cardiac activity etc. The values are

analysed by software in the surveillance equipment. The surveillance equipment sends an alarm if the analysed values in any way differ form the normal values. If no attention is taken to the abnormal values (i.e. absence of alarm) it can cause severe injury to the patient and in some cases even death. There is a great risk for serious damage if the alarm fails. The personnel need proper training to be able to connect and manage the surveillance equipment correct. Most of the personnel have this type of training, but some times they do not have the training, due to lack of time. If the surveillance equipment is connected the wrong way there is a risk for absence of alarm and the patient are exposed to danger. Now the intention is to try out new software in the surveillance equipment. Consider the following four cases:

Case 1A: Present software is used. The personal are trained on the surveillance equipment. At 7 occasions in a three-month period, there was absence of alarm from the surveillance equipment, despite the fact that there should have been alarms.

Case 1B: Present software is used. In this case personnel who have not received proper training on the equipment use the equipment. At 9 occasions in a three-month period, there was absence of alarm from the surveillance equipment, despite the fact that there should have been alarms.

Case 2A: New software is used. The personal are trained on the surveillance equipment. At 4 occasions in a three-month period, there was absence of alarm from the surveillance equipment, despite the fact that there should have been alarms.

Case 2B: New software is used. In this case personnel who have not received proper training on the equipment use the equipment.

How many alarms can be missed if the new software should be equally attractable?

In the TO-method the questions that are asked to the subject should, as it is described in Section 2.1, be based on the previous answer given by the subject. For example, if the subject answered "250" in the last round, then "250" should be one of the results that should compared to in the next round. This means that it is hard to use the TO-method based on completely pre-developed and parameterized instrumentation, e.g. paper forms. For the purpose of this research, a simple tool was developed, see Figure 4. From the screen-shot it can be seen that the appearance of the tool was not identical to the questionnaire that is described in [7], where a decision tree (e.g. Figure 2) was graphically presented to the subjects.

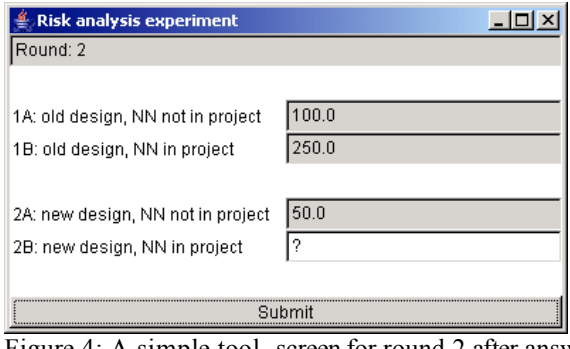


Figure 4: A simple tool, screen for round 2 after answering "250" in round 1.

3.3 Experiment design

All students first worked with scenario 1 and after that with scenario 2. In the analysis the results from each student is characterized as concave, convex, linear or "other". A curve is classified as "other" if it has not the same shape (convex or concave) for all x-values, e.g. the first half of the curve is convex and the second half is concave. In order to investigate research

question RQ1 the data from all students are pooled and the number of curves of each shape is analysed.

In order to investigate research question RQ2 the role in project was chosen as independent variable and the number of curves of each shape was chosen as dependent variable. In order to investigate research question RQ3 the scenario was chosen as independent variable and the number of curves of each shape was chosen as dependent variable.

3.4 Validity

In order to evaluate the validity of the study, a checklist from [8] is used. Validity threats may be classified as conclusion validity, construct validity, internal validity, and external validity.

The *conclusion validity* is related to the possibilities to draw correct conclusions about relations between the independent and dependent variables of the experiment. Typical threats of this type are, for example, to use wrong statistical tests, to use statistical tests with too low power, or to obtain significant differences by measuring too many dependent variables ("fishing and the error rate"). Since there were only moderately many participants in the study, care must be taken when it is stated that no difference between two groups are found. It can only indicate that there is no difference, which is further discussed in Section 4.

The *internal validity* is affected by confounding factors that affect the measured values outside the control, or knowledge, of the researcher. This may, for example, be that the groups of subjects carried out their assignments under different conditions, or maturation of participants. In order to lower the internal threats in this experiment all students carried out the assignment the same time during a 90 minutes seminar when one of the researchers was present. One threat to this study is that the two scenarios were analysed in the same order by all students. This

should be taken into account when the difference between the scenarios is analysed, i.e. when RQ3 is analysed. The reason for letting every participant work with the scenarios in the same order was that it was seen as positive that the students started with a scenario that presents a familiar project and system.

Threats to *construct validity* denote the relation between the concepts and theories behind the experiment, and the measurements and treatments that were analyzed. We have not identified any serious threats of this kind.

The *external validity* reflects primarily how general the results are with respect to the subject population and the experiment object. The intention is that the subjects in this experiment should be representative of engineers working with this type of estimation in live projects. As we see it, the largest threat to validity is of this kind. It cannot be concluded with any large validity that the students that participated in this experiment are representative of professional practitioners. Scenario 2 is not in any way related to the students' course work, but scenario 1 was based on the projects that the students participated in the course. However, the scenario was still a hypothetical scenario and it was studied in the testing phase of the project, i.e. after the risk assessment in a real project.

4 Results and analysis

The experiment was conducted with 47 students, but one of them did not hand in any results, which means that there were 46 students that completed the tasks. The number of subjects that completed scenario 1 was 44, since 2 of the subjects were discarded because the scenario was only iterated three times. The minimum of iterations was set to four times. In scenario 2, 3 subjects were discarded for the same reason so the number of subjects that retained for further analysis was 43.

Table 2. Distribution of utility functions

	Concave	Convex	Linear	Other
Scen.1	20 % (9)	32 % (14)	30 % (13)	18 % (8)
Scen.2	5 % (2)	23 % (10)	58 % (25)	14 % (6)

In order to investigate research question RQ1 the distribution of utility functions were analysed. The distribution between concave, convex, linear and other utility functions for the two scenarios are displayed in 0. The result is presented in percent of the total number of subject for each scenario, and in absolute figures in parenthesis. The students had different roles in their project groups. There is a difference in the number of students connected to the various roles. The largest group were developers (18 students) and the smallest group were project leaders (6 students). The values for each role and type of utility function are presented in 0.

Table 3. Roles and utility functions

Role	Scen	Concave	Convex	Linear	Other
PL	1	17% (1)	50% (3)	33% (2)	0% (0)
	2	0% (0)	40% (2)	60% (3)	0% (0)
TR	1	25% (2)	50% (4)	0% (4)	25%(2)
	2	25% (2)	0% (0)	50% (4)	25%(2)
D	1	17% (3)	28% (5)	39% (7)	17%(3)
	2	0% (0)	29% (7)	65%(11)	6%(1)
T	1	25% (3)	17% (2)	33% (4)	25%(3)
	2	0% (0)	23% (3)	54% (7)	23%(3)

The data has been analysed with a number of chi-2 tests [6] as summarized in 0. In the analysis, data from people with responses other than convex, concave and linear was discarded.

For RQ1, a chi-2 goodness of fit test was carried out in order to see whether the three shapes were equally probable. Data from both scenarios was pooled. It was clear that the shapes were not equally probable, which shows that the shape that results from the method is not completely random. Concerning RQ1, the most important contribution lies in the fact that different people respond in different ways, and the distribution of the different shapes.

Table 4. chi-2 tests

RQ	Independent variable	p
RQ1	-	0.0006***
RQ2	PL+TR vs D+T	0.66
RQ3	Scenario	0.012*

*significant at the 5% level, **1% level, ***0.1% level

Concerning RQ2 there are too few data points to be able to carry out a Chi-2 test that compares the shapes of each role. Therefore, data from project leaders and "technical responsibility" was pooled and data from developers and testers were pooled, which means that an analysis comparing "management roles" to more developer-oriented roles could be carried out. There is no statistically significant difference.

In the analysis of RQ3 it was found that there is a clear difference between the two scenarios, i.e. the distribution of curves is different for the two scenarios.

5 Discussion and Conclusions

From this study it is possible to conclude that different study participants have different opinions about how serious risks concerning faults remaining after testing are. It is probably possible to generalize this and conclude that different people in the software engineering process are more or less risk seeking. This is important to know in a risk management process.

Methods for assessing the level of risk seeking are available (e.g. the TO method), but in most cases it is probably enough to be aware of the differences.

Based on this study, it has not been possible to state that any role is more risk seeking than any other role. This is either because there are too few subjects or that there actually are no large differences. This means that it is not possible to formulate any simple ways to assess how risk seeking a person is based on the role that he/she has in a project.

It is possible to observe a difference between the two scenarios. In scenario 1 there are more convex (risk averse) curves than in scenario 2. The result from scenario 2 shows dominance of linear utility functions. In scenario 2 a more risk-averse tendency may be expected since the scenario concerns severe injury to patients or even death, but this is not the case. The only explanation that has been found is the fact that the subjects were used to the TO-method and the tool and knew how it works during scenario 2, see Section 3.4. However, this has to be further analysed.

There are, as described in Section 3.4, some threats to the validity of this study and future studies will be adjusted. People with more experience in general and with more experience from their project-roles should be involved in the study. If a similar experiment design is chosen, it should be adapted so that all subjects do not work with both scenarios in the same order. There were reasons for choosing this design in this research, but in further studies it is probably better not to have the same order for all participants.

References

[1] CMMI Product Team, Capability Maturity Model Integration (CMMI), Version 1.1, Staged representation, Software Engineering Institute, Technical report CMU/SEI-2002-TR-029, 2002.

[2] Fairley, R.E. Software Risk Mangement, *IEEE Software*, May/June 2005, p. 101.

[3] Fennema, H. and van Assen, M. Measuring the Utility of Losse by Means of the Tradeoff Method, *Journal of Risk and Uncertainty*, Vol. 17, No. 3, 1999, pp. 277-295.

[4] Pfleeger, S.L. Risky Business: What We Have Yet to Learn About Risk Management, *Journal of Systems and Software*, Vol. 53, 2000, pp. 265-273.

[5] Sommerville, I. *Software Engineering*, 7:th edition, Addison Wesley, 2004.

[6] Sidney, S. and Castellan, N.J. *Non-Parametric Statistics for the Behavioral Science*, McGraw-Hill, 1988.

[7] Wakker, P. and Deneffe, D. Eliciting von Neumann-Morgenstern Utilities when Probabilities are Distorted or Unknown, *Management Science*, Vol. 42, No. 8, August 1996, pp. 1131-1150.

[8] Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., and Wesslén, A. *Experimentation in Software Engineering: An Introduction*, Kluwer Academic Publishers, 1999.

Paper IV

Risk Identification by Physicians and Developers - Differences Investigated in a Controlled Experiment

Christin Lindholm, Martin Höst

Accepted for publication in proceeding of the ICSE 2009 Workshop on Software Engineering in Healthcare, Vancouver, Canada, May 2009.

Abstract

Risk management is an important process and risk identification is an important part of this process, especially in development of medical software. This paper presents an experiment where physicians, developers and software developers for medical devices are asked to identify risk in a given scenario describing the procurement of a patient monitoring system. It is concluded that multiple roles and thereby different experiences, will affect the risk identification process. Involving multiple roles, for example users and developers in the risk identification process, will result in a more complete set of identified risks than if only one role is included in the process.

1 Introduction

A small fault or mistake made in our daily life might not be so severe but in the health care domain the smallest mistake in development can make the difference between life and death. It is crucial that medical devices do not fail in any way. If they do, they can harm both the patients and the medical staff.

Physicians deal with risks as part of their work in different ways. Physicians working in the intensive care unit, the surgical ward or the emergency care unit comes in contact with a lot of different medical devices and these medical devices can also add different risk to the rest of the risks in the care situation.

All organisations that develop medical devices must have a risk management process according to law [1]. How strict and detailed this process must be depends on the safety criticality of the product. The same law as the medical device itself, regulates all software included in the medical device.

It is crucial for all types of project planning and management to carry out risk management. It is important, as early as possible in the project, to identify all the relevant risks in order to avoid or minimise the effect of the potential problems.

Risk management is often performed in several steps e.g. as described by Hall [2]. A typical process for risk management includes risk identification, risk analysis, risk planning and risk monitoring. Relevant people identify risks during the risk identification and then the risks are prioritised with the respect to the probability of the risk actually occurring and the potential effect it will have if the risk occurs. According to Pfleeger [3] the prioritising of risks are often decided through discussions where participants see risks in different ways and different values.

The research in this paper focuses on risk identification and is conducted through a controlled experiment, where physicians, developers and medical device developers have identified risks in

a given risk scenario and also prioritised risks by giving them risk values based on estimated probability and effect.

The outline of this paper is as follows. After this introduction, related work is presented in Section 2 followed by a description of the experimental design in Section 3. The results are presented in Section 4 and discussed in Section 5. Finally the conclusions are summarised in Section 6.

2 Related work

This paper concerns research about the first step of the risk management process, i.e. the risk identification step. A basic assumption is that if multiple roles, and thereby different experiences, are involved in the identification activity, the resulting list of identified risks will be more complete than if only one role was included. There is not much research about the effects of including many roles in this step, but there is some research on the risk management process and on the risk identification step. For example, in [4] risk management procedures from 133 projects were analysed in a recent survey, although the focus was not on the roles conducting risk identification.

However, in [5] the risk identification step is analysed in some detail also with respect to the participating roles. 11 different techniques for identification are listed, and at least two are directly related to the question of what roles to include. Theses techniques are "brainstorming", where a group of experts are given the task to identify risks, and "cross functional teams", where the teams identifying risk are composed of different functional areas in he organisation. However, in [5] no empirical evaluation of the performance of different types of teams are presented.

In [6], a related issue is discussed. Instead of discussing what kind of different risks that are identified by different roles it is

discussed how to handle that different roles give different priorities to different risks, i.e. that different persons give different assessments of risk probability and risk effect. This is related to how risk value of different risks is combined, although the focus of this paper is not as much on this part.

In [7] it is reported from a study where 51 middle managers were interviewed based on a scenario concerning risk identification. It was found that there was no significant effect, neither of number of years in management role nor of number of years of current job title on risk identification performance. However, there was an effect of "predominant style of information search", e.g. in what way information is sought and to what extent decisions are based on prior experience. This is obviously not the same as investigating different roles, although it concerns the question of what persons that actually do the risk identification.

Not much research is found about risk management and risk identification in specific in the medical device domain, even though all organisations developing medical device have to have a risk management process according to law. Ratkin [8] states however that companies are required to have expertise in effective risk management practices, to be familiar with software safety and to be able to adopt a risk management mind-set.

3 Experiment design

The research is conducted through an experiment where the groups of physicians, software developers, and software developers specialised in development of software for medical devices are compared with respect to performance in risk identification. This experiment can a bit more formally be described as a "quasi experiment" [9]. A quasi-experimental design follows the experimental approach to design but does not

involve random allocation of participants to different groups [9]. Since the subjects involved in this experiment are different categories of professional practitioners, i.e., physicians, developers and medical device developers, this is not possible to randomise over a sample of people.

That is, the independent variable of the experiment is the role, which can have three different values: physicians, software developer, and software developer specialised in development of software for medical devices.

3.1 Research questions

The objective of the research in this paper is to investigate if there is any difference between physicians who are users of systems and developers of systems regarding the view of risks. More specific research questions are:

- RQ1: Which difference can be identified between the numbers of risks identified by users and the number of risks identified by developers?
- RQ2: What are the differences between the kind of risks identified by users and the kind of risks identified by developers?
- RQ3: What are the differences between the groups with respect to which risks they see as important?
- RQ4: What risk overlap can be identified between the professional groups?

For research question RQ1, the dependent variable is the number of people from each role that have found the risk. That is, for each risk it is counted how many physicians that has found it, how many developers that has found it, etc.

For research question RQ2, the dependent variable is the kind of risks identified by the participants from each role related to the defined categories in Table 1 in Section 3.3. The number of risks for each category is counted in total and for each group of participants.

The differences between the groups with respect to which risks they see as important, research question RQ3, the dependent variable is the risk value assigned to the risks by the participants. The risks identified by all three groups have been analysed to find the risks most important for all the participating groups. In order to see the difference between the roles, the risks with highest risk values for each group was analysed and then compared.

In the analyse of research question RQ4 the risks in common for all the three participant groups have been used to identify what risk overlap that exists.

Research question RQ1 have been analysed with statistical test and research question RQ2-RQ4 have been analysed by descriptive statistics.

3.2 The experiment

The experiment was performed during the year of 2008, with 15 physicians, 15 developers and 6 medical device developers as participants. All the participants were presented to the same risk scenario describing the procurement of a patient monitoring system.

The involved subjects are professional developers and physicians working in Sweden. The physicians are all anesthetists employed at the same clinic in a hospital in Sweden. As an anesthetist they alter their work between three different units, the intensive care unit, the surgical ward and the emergency care unit. Working in these three different units involves handling a lot of different medical devices. Before the risk scenario was sent out to the physicians an information meeting were held to explain the experiment and the purpose of the experiment. The risk scenario was not exposed to the physicians at this information meeting. A description of the risk scenario, a reply form and a self-addressed envelope were sent out to the 37 physicians by ordinary mail. The physicians were

asked to send in their answers within 2 weeks. After 3 weeks a reminder with the same content was sent out. In total 15 physicians returned their answers, i.e., a reply rate of 15/37 = 40%.

The developers were asked if they would like to participate in the experiment by e-mail. This e-mail was sent out to developers in different Swedish companies developing software and in the e-mail the experiment and the purpose of the experiment was explained. The developers were addressed directly and e-mail was sent out to 26 developers and 15 accepted to take part in the experiment. That is the reply rate of the developers was 15/26 = 58%.

The same risk scenario as the one sent to the physicians was sent to the developers with a minor difference, that some of the medical terms was explained. The risk scenario and reply form was sent by e-mail and the developer returned the reply forms after answering, by e-mail. The most difficult group to get participants from was the medical device developers. Despite 32 different information e-mails directly to individuals and different companies developing medical devices in Sweden and several telephone calls only 6 participants participated in the experiment. This means that the reply rate of this group was 6/32 = 19%. The major reason given for not participating in the experiment was lack of time. The risk scenario and reply form that was sent out to the medical device developers by e-mail were identically as the one sent to the developers. The reply forms received from the medical device developers were also returned by e-mail.

The risk scenario is around 1¼ page long and is written in Swedish describing the procurement of a patient monitoring system. When the risk scenario was written some things considered as risks by the researchers was deliberately incorporated in the scenario but no too obvious risks were chosen. The risk scenario describes the following scenario:

”It is the county council that have decided to buy a new patient monitoring system with the intention to have the same monitoring system at all the units at the hospital, for example at the intensive care unit, the surgical ward and the emergency care unit. The goal for the county council is to rationalise the care and reduce cost for the daily activity. The county council have requested 10 companies for tender and the tender text is presented in the scenario. The tender text is divided in four parts; Dimensioning and functionality, Location, presentation and compability, Mobile monitoring and Communication and use. The system shall, for example, have one central server, it shall be possible to monitor 22 patients through wireless communication, and the functionality that the system at least shall include is specified. It shall be possible to use the new system together with other medical devices, the new system must guarantee patient safety, it shall be possible to easy physically transport the patient together with the system, the system should be able to communicate with other wards and external partners, a user should be able to use the system after five hours of training etc. That is, the scenario includes both functional and non-functional requirements. The risk scenario ends with a description of the company the county council have decided to give the contract to. This company is a new, promising and expanding company on the medical device market and they promises to deliver the system in 6 month to the lowest cost. The majority of the staff is developers that recently have taken their degree. That is, the scenario also includes information about the development organisation and a description of the experience and competences of the developers”.

All the participant in the experiment were asked to study the risk scenario, identify risks and write down the risks in their own words (i.e. "free text form") in the reply form. The participants were also asked to estimate the probability of the

risk and to estimate the effect of the risk. For the probability, the participants were given a graded scale 1 – 4, where 1 represents that it is very unlikely for the risk to occur, 2 represents that it is unlikely, 3 that is likely and finally 4 represents that it is very likely the risk will occur. A scale was also given for the estimate of the effect if the risk occurs. This scale was graded 1-5 where 1 represents that the effect would be insignificant if the risk would occur 2 that the effect would be acceptable, 3 that the effect would be serious, 4 that the effect would be very serious and 5 the effect would be catastrophic if the risk would occur.

In the analysis the researcher then calculated the risk value, R, for each given risk by multiplying the given figure for probability, P, by the given figure for effect, E, as $R=P \times E$. Thus, the highest possible risk value a risk can get in this experiment is $R = 4 \times 5 = 20$.

3.3 Analysis

The data was collected from the reply forms that the participants sent in. All the risks were put together in a digital format. Each risk was given a standardised risk description and a risk identifier by the researcher. The risk with the same content and meaning was counted but registered as the same risk with same risk identifier and risk description.

Each risk was then categorised according to the type of risk in 15 categories shown in Table 1.

Table 1. The risk categories

Category	Example of risk
Education	Too short education of the users before using the new system
Delivery time	Too short time to delivery (6 month)

Support	Upgrading of the new system in the future
Cost	The budget is out of control
Alarm	The alarm do not work as intended
Wireless transmission	The wireless transmission do not work as intended
Back up	One central server is not enough
Requirement specification	Vague requirement specification
Security	The security is at risk if there is communication with extern partners
Experience	The developers that recently passed their degree have none or slight experience in developing this kind of system
Company	Because it is a new company there is a risk that the company goes bankrupt
Bidding procedure	Vague tender
Introducing the new system	The new and old system will not run in parallel. The old system is shut down when the new system is installed
Problem with the new system	The new system give the wrong information
Development process risks	The customer does not have time to participate in the process

These categories have been defined based on both the researchers assumptions and intentions about the present risks in the scenario, and based on the risks that actually were identified by the participants. That is, the identified risks were allowed to affect the categories of risks.

The risk values for each risk and the professional groups were also registered and analysed. The experiment data was analysed with descriptive statistics and statistical tests. Research question RQ1 was analysed with statistical tests, and research questions RQ2-RQ4 were analysed with descriptive statistics.

Research question RQ1 was analysed by comparing how many of the people from every role that identified every specific risk. That is, the following metric was calculated:

$$M_{Role,Risk} = \frac{F_{Role,Risk}}{N_{Role}}$$

where $F_{Role,Risk}$ denotes the number of individuals from a role that found a specific Risk, and N_{Role} denotes how many people there are available from that role, i.e. how many that could have found the risk. This means that the relative number of people from each group is compared, which is necessary since there are fewer physicians than developers.

This data can be analysed with a repeated measures analysis of variance (ANOVA) design with the following model:

$$M_{Role,Risk} = m + a_{Role} + b_{Risk} + c_{Role,Risk,}$$

where $c_{Role,Risk}$ is an error term which is normally distributed with mean 0. In this model, m represents the overall mean value, *a* represents the effect of the roles, i.e. that roles do not have to be equally effective in identifying risks, and *b* represents the effect of the risks, i.e. the fact that all risks are not equally hard to identify. This means that it is possible to test the null hypothesis

H0: $a_{Role} = 0$, for all roles,

i.e. that there is no effect of role on the number of identified risk. That is, if it is possible to show that a_{Role} is not 0 (i.e. a_{Role} is not the same for all roles) it is shown that all roles are not

equally effective in identifying risks. A repeated measures design was chosen because it takes into account the fact that all risks are not equally hard to identify. For more information about this kind of model and analysis, refer e.g. to [11] or [12]. The data cannot be assumed to be normally distributed so a non-parametric statistical test, i.e. the Friedman test [12], which is a non-parametric alternative to the above described analysis, was also applied.

3.4 Validity

A major concern in quasi-experiments is the threats to validity. The interpretation of findings is more complex than "true" experimental design according to Robson [9]. It is important to consider the validity threats already during the design of the experiment, so the validity threats can be reduced as much as possible. Validity refers to accuracy of results and validity threats may be dived into four types [10]. These four types are conclusion validity, construct validity, internal validity and external validity.

Conclusion validity is related to the possibilities to draw the correct conclusions regarding the relationship between the independent and dependent variables of the experiment. Threats to conclusion validity can be the use of wrong statistical tests, for example using statistical tests where the statistical power of the tests are to low. To reduce this threat the choice of statistical test have been made very carefully and for example analysis with non-parametric test has also been done.

Construct validity is concerned with the getting the correct measures for the concept and the relationship between the concepts and theories behind the experiment. A threat can be that all the participants do not interpret risk the same way as the researchers intended. The scales for estimating the probability and the effect of the risks can also be interpreted in different ways even if the intention from the researchers has

been to make the instructions and the scales as unambiguous as possible. In order to try to mitigate construct validity the instructions and scales were written and defined with the up most intension to be as clear and unambiguous as possible. Two developers and one physician whom not participated in the experiment reviewed and commented the instructions, scales and scenario before use.

Internal validity is affected by factors that affect the measures but that are outside the control of the researchers. A threat to this study and an affecting factor can for example be that the participants read and filled in the replay form under different conditions that was out of the researchers control. We have, however, not seen any signs of this. Another common threat with respect to this is that the participants in different groups have different experience. The physicians have, as it is described in Section 4.1, somewhat longer experience in average than the other two roles. Even if this is a threat to the study, we do not see it as too serious. This distribution could in the future be compared to the typical distribution of the roles, i.e. physicians and software developers, although this has not yet been done.

External validity primarily relates to how general the result of the experiment is for example how representative the problem in the assignment is. Another threat could also be that the participants is not representative of the target population, so to lower this threat in the experiment the subjects asked to participate in the experiment is working as professional developers or physicians. However the physicians have the same specialty and practice in the same clinic at the same hospital, this can be a threat to how general the results are. In further experiments physicians from different hospitals should be included. All the participants may not have done this kind of risk analysis before or been in contact with this kind of scenario before but however all participants are used to dealing with risks and the scenario is based on a scenario they quite likely could be

confronted with. A threat can though be that ca 1/3 of the developers in this experiment have taken a degree at Lund University where courses containing risk analysis have been taught. No checklist for risks was introduced to the participants but this was a deliberate choice of the researches with the intension not to control the participants. However there were no restrictions against using one. The risk scenario itself is a threat because the scenario is fictional but written with the up most intension to be as realistic as possible. Before the use of the scenario, it has been reviewed and commented on by one physician and two developers; none of them have taken part as participants in the experiment.

4. Results

4.1 Results from the controlled experiment

The experiment was preformed by 36 participants, 15 physicians, 15 developers and 6 medical device developers. All the participants were asked to specify how many years they have been working in their profession. They were asked to mark "< 2 years", "2-5 years", or "> 5 years". There are differences in the three groups as can be seen in Table 2, and which is also discussed in Section 3.3. The majority of the physicians and the medical device developers in this experiment have been working in their professions for more than 5 years while the majority of the developers have been working for 2-5 years in their profession.

Table 2. Working years in profession

Year in profession	Physicians	Developers	Medical device developers
< 2	7 %	33 %	33 %
2-5	0	40 %	17 %
>5	93 %	27 %	50 %

The participants have also stated how many minutes they have used for the experiment. The time in average in the different groups varies, and not all the participants have answered this question. 12 of the 15 physicians have spent in average 26 minutes per person on the experiment where 45 minutes is the longest time spent and 5 minutes the shortest time spent. 14 of the 15 developers spend in average 1 hour and 23 minutes per person where the longest time spent is 6 hours and the shortest time spent is 30 minutes. The average time for the third group, medical device developers is 1 hour and 2 minutes per person. 5 of the 6 medical device developers answered and time varied between longest time spent 150 minutes and shortest time spent 20 minutes.

All the risks have been analysed and categorised and the risks that appeared to be the same risk have been counted and registered together. This has resulted in 197 specified risks with a unique identifier and risk description given by the researcher. Out of these 197 risks there are 54 risks that are stated by only one of the groups (risks unique for the group). Developers have identified more risks per person than the other groups as shown in Figure 1. Also here it should be taken in count that the medical device developers are a smaller group.

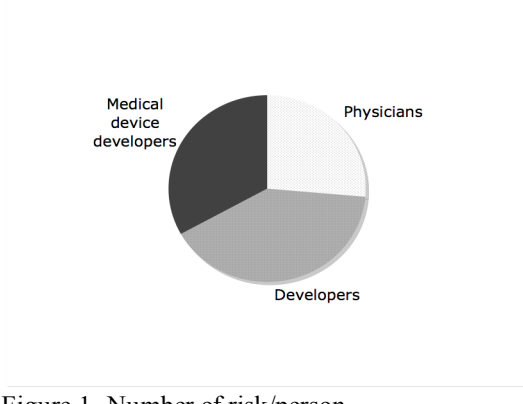


Figure 1. Number of risk/person

The total number of risks identified in this experiment is 390 risks distributed as shown in Table 3.

Table 3. Number of risks

Professional group	Members	Number of risks	Number of risks/person
Physicians	15	130	8.7
Developers	15	200	13.3
Medical developers	6	60	10.0
Total	**36**	**390**	**10.8**

The developers are the group that has identified the largest amount of risks followed by the physicians and medical device developers. However the group with medical device developers is a much smaller group than the other two groups so it could be expected that they should identify less amount of risks. Therefore looking at the number of risks found per person could be more interesting and it can be seen in Table 3 and Figure 1 that the developers found the largest amount of risks per person, followed by the medical device developers and smallest amount of risks per persons was found by the physicians. One possible reason that the developers found more risks per persons can be they spent more time on their risk identification process than the physicians.

Concerning research question RQ1, the resulting values of metric *M,* as described in Section 3.3, for each risk and role is displayed in Figure 2

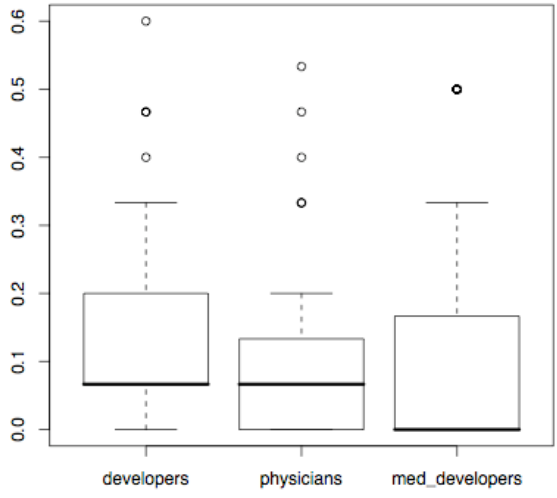


Figure 2. Box-plots of relative number of people identifying each risk (*M*).

Some differences between the groups can be found and the result of an analysis of variance is that there is a significant effect of the role on *M*. The resulting ANOVA-table is depicted in Table 4.

Table 4. Result of ANOVA-test for RQ1.

	Df	F-value	Pr(>F)
Role	2	7.4014	0.0007747 ***
Risk	110	2.9937	
Residuals	220		

A Friedman test for the same hypothesis gives a p-value of 4.785e-06, which means that this too indicates a significant difference.

An analysis of the difference between only physicians and developers also gives a significant difference (p-values are 1.23e-05 for a paired t-test and 1.13e-06 for a Wilcoxon test, which is a non-parametric alternative). That is, there is not only a difference between physicians and developers (from Figure 2

and the ANOVA-test) but there is also difference between physicians and medical device developers.

In order to investigate research question RQ2 and the differences between the kind of risks identified by physician and the kind of risks identified by developers and medical device developers each risk has been categorised according to type of risk in the 15 categories shown earlier in Table 1. The category with the highest number of different risks are “Problem with the new system” it contains 41 different risks with unique risk identifier, these risks relate to problems with the system when it has been delivered and are up and running. “The system gives the wrong information”, “The safety for the patient reduces” and “The hospitals paging system is “knocked out”” is example of risks in this category. All three participant groups have the highest number of different risks with unique risk identifiers in this category. The developers have identified 34 of the 41 risks listed in this category, the physicians 21 and the medical device developers 15 of the listed risks in this category.

The two categories that then follow with 10 different risks in total with unique risk identifiers in each category are “Wireless transmission” and “Development process risks”. In the category “Wireless transmission” the developers have identified 9 of the 10 risks and the physicians and medical device developers 5 of the 10 risks. When it comes to “Development process risks” the physicians have not identified any development process risks at all but the developers have identified 9 out of the 10 risks and the medical device developers 2 out of the 10 risks.

In order to survey research question RQ3 regarding the differences between the groups with respect to which risks they see as important, all the participants have, and as explained in Section 3.2, given each risk they have identified a risk value, R. The highest risk value a risk can receive is $R = 20$. When analysing the 29 risks common for all the three groups a total

risk value for each of these risks was calculated by summarising the average risk value given to these particular risks from each group of professionals. Table 5 shows the four risks with the highest total risk value for the risks in common for all the three groups.

Table 5. Total risk value

Risk	Phys.	Dev	Med dev	Total risk value
The new and old system does not run in parallel	17.5	16	20	53.5
The delivery of the new system is delayed	20	13.8	13.3	47.1
One central server is not enough	17.5	15.2	13	45.7
The company goes bankrupt	14	10.3	20	44.3

Of the 29 risks in common, 9 belong to the category "Problem with the new system" and 4 belong to the category "Support". The three different professional groups give different risk value to the different risks. A top ten risk list out of the 197 risks for a physician is not the same top ten lists as for a developer or for a medical device developer. The top four risks for each group are presented in Table 6.

Table 6. Top four risks

Physicians Risks	Phys.	Dev	Med dev
Vague requirement specification	20	0	0
Old documents is lost	20	0	0
The hospitals paging system is "knocked out"	20	0	0
One central server is not enough	17.5	15.2	13

DevelopersRisks	Dev	Phys.	Med dev
Upgrade of the new system in the future	20	0	0
Problem getting the new system running	20	12	6
One central server is not enough	15.2	17.5	13
No delivery of the system at all	15	0	0
Medical Developers Risks	Med dev	Phys.	Dev
The wireless transmission put other units out of order	20	0	4
Vague tender gives low quality	20	0	12
The company goes bankrupt	20	14	10.3
The new and old system does not run in parallel	20	17.5	16

The risks identified and given the highest risk value by the physicians are risks that are not identified at all by the other two groups. Also one of the risks given the highest risk value by the developers is a unique risk, a risk only identified by the developers. None of the risks identified by medical device developers and given high risk value is unique for the medical device developers as a group however, have the medical device developers given these risks higher risk value than the other two groups. The medical device developers have also a larger amount of risks with the highest risk value, 20, than the other two groups.

Regarding research question RQ4 and if their is any risk overlap between the groups it can be seen that there are 29 risks out of the 197 risks (15%) that are common to all the three groups, this means that the risk have been identified by at least one participant in each group of professionals.

If the groups are merged and studied, the group of physicians and medical device developers as one group and developers and medical device developers as another group give us two groups equal in size. It was found that developers and medical device developers have 8 risks that are in common for these two groups only, compared to physicians and medical device developers they have only one risk in common exclusive for them. Physicians and developers have the largest amount of risks, 19 risks, in common but this could be explained with that it is the largest group of participants (30).

5 Discussion

Risks and risk management is an important area. It is crucial for all types of project planning and management to perform risk management. All organisations developing medical devices are obliged by law to have a risk management process. Risks are also something that affects physicians whom deals with risks in all care situations. Risks are therefore a major concern for all of the professional groups participating in this experiment.

There is a difference regarding the view of risks between physicians, developers and medical device developers shown in this experiment. Looking at the number of identified risks, developers identified a larger amount of risk per person than physicians. A possible reason for this can be that developers are more used to the process of identifying risks in this way, presented in the experiment than the physicians.

Often most developers have sometimes worked in projects, which make it a familiar working form for them but probably not for physicians. This could explain that all the three groups identified similar types of risks with the exception that none of the physicians identified any development process risks at all.

It was shown that the physicians did not identify any risks that are typical medical risks so the developers could also have

identified all risks identified by physicians. The three risks the physicians gave the highest risk value was not identified by the other two groups but they easily could have. In this experiment no checklist for risks have been used in order not to control the participants. It could be interesting to do this experiment with a checklist and see if it has any effect on the result.

6 Conclusion

The research in this paper presents an experiment where physicians, developers and medical device developers are asked to identify risk in a given scenario in order to investigate if there is any difference regarding the view of risks. Our basic assumption is that multiple roles, and thereby different experiences, will affect the list of identified risks and that it will be more complete than if only one role is included.

It can be concluded that there is a difference between the different professional groups regarding the view of risks in this research study. The different experiences affect the risk identification and also the prioritisation of risks. There is a difference in the number of people from each role that has found a risk and there is a difference in between the groups with respect to which risks they see as important. However there is no distinct difference in the kind of risk identified by the participant groups with one exception that the physicians have not identified any risks that could be categorised in the development process risk category.

The risk overlap between the participant groups are rather small and given that the results can be replicated and generalised we can conclude that it is important to include participants from different professional groups in the risk identification process.

It can be concluded that it is necessary, at least for this kind of system, to include the users in the risk identification process in order to get more complete risk identification. It is not

sufficient to only include the developing organisation in identification of risks. Involving different roles in risk identification may probably be advantageous in several types of systems.

The researcher presented in this paper has indicated the necessity of incorporate the user in the risk identification process. Furthers research could involve the development of practical guidelines checklists and workshop processes for medical device industry and the goal in the long run should be to influence and contribute to standards as for example IEC 62304 and ISO 13485.

References

[1] Commission of the European Communities, Council Directive 93/42/EEC EEC concerning medical devices.

[2] E. M. Hall, *Managing Risk: Methods for Software systems development,* Addison Wesley, 1998.

[3] S.L. Pfleeger, Risky Business: What We Have Yet to Learn About Risk Management, *Journal of Systems and Software*, Vol. 53, 2000, pp. 265-273

[4] J. Li, R. Conradi, O. P. N. Slyngstad, M. Torchaniano, M. Morisio, C. Bunse, A State-of-the-Practice Survey of Risk Mangement in Development with Off-the-Shelf Software Components, IEEE Trans on Software Engineering, Vol 34, No. 2 March/April 2008.

[5] D. Kasap, M. Kaymak, Risk Identification Step of the Project Risk Management, International Conference on Management of Engineering & Technology, Portland 5-9 Aug. 2007.

[6] S.M.H. Mojtahedi, S.M. Mousavi, A. Makoui, Risk Identification and Analysis Concurrently: Group Decision Making Approach, 4th IEEE International Conference on on Management of Innovation and Technology, 2008.

[7] E. Maytorena, G.M. Winch, J. Freeman, T. Kiely, The Influence of Experience and Information Search Styles on Project

Risk Identification Performance, IEEE Trans on Engineering Management, Vol 54, No. 2, May 2007.

[8] S.R. Rakitin, Coping with Defective Software in Medical Devices, *IEEE Computer,* volume 39, 2006, pp 40-45.

[9] C. Robson, *Real world research*, second edition, Blackwell Publishers Ltd, Oxford, 2002

[10] C. Wohlin, P. Runeson, M. Höst, M.C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering: An Introduction*, Kluwer Academic Publishers, 1999.

[11] D. Montgomery, *Design and Analysis of Experiments*, Wiley, 2001.

[12] P. Dalgaard, *Introductory Statistics with R*, Springer, 2002.