Medical Device Standards' Requirements for Traceability during the Software Development Lifecycle and Implementation of a Traceability Assessment Model

Gilbert Regan, Fergal Mc Caffery, Kevin Mc Daid, Derek Flood

DKIT, Dublin Road, Dundalk, Ireland

{gilbert.regan,fergal.mccaffery, kevin.mcdaid, derek.flood}@dkit.ie

Abstract. Developing safety critical software is a complex process. Due to the fact that medical device software failure can lead to catastrophic consequences, numerous standards have been developed which govern software development in the medical device domain. Risk management has an important role in medical device software development as it is important to ensure that safe software is developed. Demonstrating traceability of requirements right throughout the medical device software development and maintenance lifecycles is an important part of demonstrating that 'safe' software has been produced through adopting defined processes. Consequently, medical device standards and guidelines emphasise the need for traceability.

This paper outlines the extent and diversity of traceability requirements within medical device standards and guidelines, and identifies the requirements for traceability through each phase of the software development lifecycle. The paper also summarises the findings obtained when a lightweight assessment method (Med-Trace), that we created, based upon the traceability practices within these standards, was implemented in two SME organisations. Finally we highlight how the findings indicate a lack of guidance as to what is required when implementing and maintaining a traceability process.

Keywords: Med-Trace, Medical Device Standard, Software Traceability, Software Process Assessment and Improvement, Software development lifecycle

1 Introduction

Software-based medical devices are playing an increasingly important part in healthcare [1]. Many medical devices must interface with other equipment, connect to hospital and laboratory information systems, and work in high-stress situations. The increased demands on such devices has resulted in increased software complexity and has created formidable development challenges for their manufacturers [2]. This increased complexity has resulted in the need for increased traceability and risk control measures. Software tools are highly important in ensuring traceability from requirements via specification to implementation [3].

In order to market their devices within a country, a medical device development company must comply with the regulatory requirements of that country [4]. Although guidance exists from regulatory bodies on what software activities must be performed, no specific method for performing these activities is outlined or enforced [5].

Although no particular software development lifecycle is mandated, organisations must clearly demonstrate how they had adhered to a chosen lifecycle through the on-going production of objective evidence throughout development. Organisations must demonstrate clear linkages through the software development and maintenance lifecycles. Numerous standards and guidance documents exist which interpret requirements for compliance to regulations. However, deciphering these requirements can be a difficult and onerous task as the requirements for traceability vary between the documents. The variation between the documents extends to other areas such that "In already developed guidance documents different software quality issues, software lifecycle phases and consequences of risk evaluation for software malfunction or fraud are addressed to different extents"[6].

This paper identifies the references to traceability within each of the relevant medical device standards and guidelines and defines the requirements for traceability during each stage of the software development lifecycle. It also considers the implementation of Med-Trace which is a traceability assessment model.

The remainder of this paper is structured as follows; Section 2 outlines the background to medical device regulations, standards and guidance documents. Section 3 discusses the importance of traceability in all domains culminating with the medical device domain. Section 4 identifies traceability requirements within the medical device standards. Section 5 considers the implementation and findings of Med-trace, a traceability assessment model. In Section 6 we draw our conclusions.

2 Background to Medical device software

2.1 Regulations, Standards, Guidance Documents and Technical Reports

Regulations are the instruments that governing bodies use to assure the safety and efficacy of medical devices. The important point about regulations is that they are mandatory. An organisation must understand the regulations of the region where they plan to market their medical device. In the United States the regulation for medical device is the **Code of Federal Regulations 21 CFR 800-1299**. In Europe the regulations are the **Medical Device Directives (MDD)** and amendments. The level of detail in the regulations is limited due to the following reasons: (a) a company may already have their own compliant processes in place and imposing new detailed regulations would incur unnecessary cost and effort, (b) developing a set of regulations that would cover every present and future situation would be extremely difficult and (c) imposing detailed regulation would stifle creativity.

However the lack of detail in the regulations raises issues. Organisations (especially those new to the medical device domain) need to understand what they must do in order to be compliant, which can be timely and costly. Standards, guidance documents and technical reports (TIR's) assist organisations in this understanding. Guidance documents such as the General Principles of Software Validation (GPSV) are published by the Food and Drug Administration (FDA). These documents explain the requirements of the FDAs' regulations. Guidance documents are voluntary. Standards typically are developed by workgroups comprising of members from the regulatory bodies and industry experts. The objective of the standards is similar to the FDA's guidance documents (help organisations to be regulatory compliant). As with the guidance documents, compliance with any standard is voluntary, although if you do not comply with the standards you have to present a strong argument to regulatory auditors that the practices that you have adopted are at least as good as those defined in the standards.

ANSI/AAMI/IEC 62304:2006 is the recognised standard for medical device software lifecycle processes and is adopted widely within the global medical device industry and is accepted within the industry as being best practice for the development and maintenance of medical device software. Technical reports contain industry best practices. Unlike standards and guidance documents they do not stipulate one particular way of meeting regulatory requirements.

2.2 History

In 1993 the European Council released the Medical Device Directive (MDD) 93/42/EC [7]. The purpose of this directive was to ensure the safety of medical devices placed on the European market. This directive has been amended by Directives 2000/70/EC [8], 2001/104/EC [9], 2003/32/EC [10] and 2007/47/EC [11].

To this end, in the USA, the FDA's Center for Devices and Radiological Health (CDRH), independently from the European Council, published guidance papers which include risk based activities to be performed when using off-the-shelf software [12], during software validation [13], and for pre-market submission [14]. These documents however did not enforce any specific method for performing the software activities; hence manufacturers could fail to comply with expected requirements.

Therefore the medical device industry decided to recognise ISO/IEC 12207 [15] (general software engineering process standard) as suitable for general medical device software development. However the Association for the Advancement of Medical Instrumentation (AAMI) identified pitfalls in ISO/IEC 12207 and produced AAMI SW68 [16] (Medical Device- Software Lifecycle Processes) which was based on ISO/IEC 12207. However, in 2006 AAMI/IEC 62304 [17] was released and this replaced AAMI SW68.

3 Traceability

3.1 Introduction

In engineering terms a trace is comprised of a source artifact, a target artifact and the link between them [18]. Traceability is the ability to establish and use these traces. Numerous definitions for traceability exist in the literature but one of the most popular and encompassing is:

> *"Requirements traceability refers to the ability to describe and follow the life of a requirement, in both a forwards and backwards direction (i.e., from its origins through its development and specification to its subsequent deployment and use, and through all periods of on-going refinement and iteration in any of these phases) "*[19].

In general, traceability is about understanding a design right through from the origin of the requirement to its implementation, test and maintenance. Traceability allows us to understand aspects such as to whether the customers' requirements are being met, the specific requirements that an artifact relates to, and the origins and motivation of a requirement. Traceability helps ensure that 'quality' software is developed.

3.2 Traceability in all domains

Software systems are becoming increasingly complex. Artifacts such as test cases, requirements documents, source code, design documents, bug reports etc., and the links between them are created over long periods of time by different people. Creating and maintaining these links is a difficult and expensive task. Therefore most existing software systems lack explicit traceability links between artifacts [20].

Traceability was initially used to trace requirements from their source to implementation and test, but now plays an increasing role in defect management, change management and project management. Increasingly software development is globally distributed across multiple teams and sites which makes traceability even more important [21]. As traceability provides an essential support for developing high quality software systems [22], it is vital to engage an efficient traceability process.

Traceability implementation is mandated in many software development standards and many industries, in particular the safety critical industries [23]. For example in the US the Food and Drug Administration states that code must be linked to requirements and test cases. Safety critical products can be dangerous because failure can result in loss of life, significant environmental damage, or major financial loss [24]. Safety critical systems must satisfy a range of functional and non-functional requirements, including safety, reliability and availability. Regulation normally requires safety critical systems are certified before entering service. This involves submission to the appropriate regulator of a safety case (a reasoned argument that safety requirements have been met and the system is acceptably safe to operate) must be made for a safety critical systems as regulation requires these systems are certified before entering service [25].

3.3 Traceability in the medical device domain

In the medical device domain, software development is a difficult and complex endeavor. Defective medical device software can cause serious injury or death. Therefore safety is a key concern [21]. In the period from 7th Feb 2011 and 7th Feb 2012 the FDA recorded 151 medical device recalls and state software as the cause [26]. The number of devices that have recently been recalled due to software and hardware problems is increasing at an alarming rate [27]. During 2009 the FDA recalled 63 medical devices because of software issues. During 2010 they recalled 107 medical devices for the same reason.

It is incumbent on medical device manufacturers to ensure, to the best of their ability, that software-based medical devices are safe and effective. Meeting this responsibility requires expertise in effective risk management practices, familiarity with software safety, and the adoption of a risk management mind-set [2]. Manufacturers must establish effective software development processes that are based on recognised engineering principles appropriate for safety critical systems. At the heart of such processes, they must incorporate traceability.

Generally there is a lack of published material regarding traceability in medical device software in addition to a lack of guidance on how to implement traceability effectively in recognisations [21]. As traceability is central to the development of medical device software, a traceability assessment and improvement method called Med-Trace [23] has been developed (See section 5).

4 Medical device software regulations, standards and guidelines: requirements for traceability

4.1 Regulatory Requirements

Software traceability is central to medical device software development and essential for regulatory compliance. To understand the level of traceability required within the medical device domain, each of the medical device regulations, standards and guidelines were studied in detail for both explicit and implicit references to traceability through the software development lifecycle. The requirement for traceability is apparent in many of the medical device software standards and guidelines and is highlighted in Section 4.2. However the requirement for traceability through the software development lifecycle is not as obvious in the regulations.

The Medical Device Directive is the European Union's regulation for medical devices. Its main objective is to ensure that medical devices should provide patients, users and third parties with a high level of protection and attain the performance levels attributed to them by the manufacturer. It makes no direct reference to traceability through the software development lifecycle, only referencing traceability in Section 3.2e which specifies tracing of test equipment calibration; '*it must be possible to trace back the calibration of the test equipment adequately'*

Title 21 CFR Part 820 Quality System Regulation sets forth good manufacturing practice (CGMP) requirements in the United States and refers to traceability at the lot or batch level. Subpart F, Section 820.65 states '*Each manufacturer of a device … shall establish and maintain procedures for identifying with a control number each unit, lot, or batch of finished devices and where appropriate components. The procedures shall facilitate corrective action. Such identification shall be documented in the DHR'*. There is no reference to traceability within the software development lifecycle in this regulation.

While none of the above regulations directly refer to traceability throughout the SDLC, references are made to validating software according to the 'state of the art'. An example of this can be seen in the Medical Device Directive Annex I 12.1a:*'For devices which incorporate software or which are medical software in themselves, the software must be validated according to the state of the art taking into account the principles of development lifecycle, risk management, validation and verification'.*

Although 'state of the art' can be open to some interpretation it is generally accepted that for medical device software, ANSI/AAMI/IEC 62304:2006 and aligned standards (as noted in section 4.2) are 'state of the art'.

4.2 Standards and Guidelines

While the requirement for traceability is not transparent from the regulations, the standards and guidelines requirements for traceability are much more comprehensive. Detailed requirements for traceability, as expressed by the medical device standards and guidelines, are summarised in this section. Table 1 details the number of times (including section numbers for each instance) each standard identifies traceability. Table 2 provides an example of two of these references.

Table 1: Number of times (and section numbers) each standard impacts traceability

Standard Title	No.	Section Numbers
ANSI/AAMI/IEC 62304:2006 Medical device software—Software life cycle processes (2006)	**6**	5.1.1; 5.2.6; 5.7.4; 7.3.3; 8.2.4; B.6.2;
General Principles of Software Validation ; U.S. Food and Drug Administration 2002	6	3.1.2; 3.2; 5.2.2; 5.2.3; 5.2.4; 5.2.5;
Guidance for the Content of Premarket Submissions for Software Contained in Medical Devices : US FDA 2005	**2**	Page 11; Page 16;
Off-The-Shelf Software Use in Medical Devices: US FDA 1999	**1**	5.5.1
ISO 14971:2007(E) - Medical devices — Application of risk management to medical devices	**1**	3.5
IEC/TR 80002-1:2009 - Medical device software Part 1: Guidance on the application of ISO 14971	**8**	3.5; 6.3; Table C; Table D
ISO 13485 (2003) Medical devices — Quality management systems — Requirements for regulatory purposes	**2**	7.5.3.2.1; 7.5.3.2.2;

Table 2: An example of Practice content relating to traceability taken from two standards as referred to in Table 1

Standard Title	Process	Practice	Practice Content
ANSI/AAMI/IEC 62304:2006 Medical device software— Software life cycle processes (2006)	Software development Process 5.0	Software development planning 5.1 Software development Plan 5.1.1	The manufacturer shall establish a software development plan which should ensure TRACEABILITY between SYSTEM requirements, software requirements, SOFTWARE SYSTEM test, and RISK CONTROL measures implemented in software;
ANSI/AAMI/IEC 62304:2006	Software configuration management process 8.0	Change control 8.2 Traceability change7.3.3	The MANUFACTURER shall create an audit trail whereby each: a) CHANGE REQUEST; b) relevant PROBLEM REPORT; and c) approval of the CHANGE REQUEST can be traced. [Class A, B, C]

Failure in medical device software can have fatal consequences. The gravity of these consequences is highlighted in the medical device standards through reiteration of the necessity to control risks. Traceability can control risk. For example, the **GPSV [13]** states that a software requirements traceability analysis should be conducted to trace software requirements to (and from) system requirements and to risk analysis results. Moreover **ISO 14971:2007 [28]** requires the manufacturer to establish and maintain a risk control file which shall provide traceability for each identified hazard to a) risk analysis, b) risk evaluation, c) implementation and verification of risk control measures and d) the assessment of the acceptability of any residual risks. The documentation of risk control measures is emphasised by **ANSI/AAMI/IEC 62304 [17]** which directs the manufacturer to document traceability of the software hazards: from hazard situation to software item; from the software item to the specific software cause; from the software cause to the risk control measure; and from the risk control measure to the verification of the risk control measure. The imperative for risk control is further called for in **Off-The-Shelf Software Use in Medical Devices [12], Guidance for the Content of Premarket Submissions for Software Contained in Medical Devices [14]** and **IEC/TR 80002-1 [29]** .

4.3 Variability in Requirements for Traceability

There is considerable variance in the level of traceability detail required within the standards and guidelines. Some of the standards provide very little detail in relation to which stages of the SDLC traceability should be provided (e.g. **ISO 13485** refers to traceability at a high level stating that an organisation is required to establish documented procedures for traceability and that such procedures shall define the extent of product traceability and the records required). However, other standards provide a greater level of required traceability detail such as **GPSV** which requires traceability from system requirements to software requirements and through each stage of the SDLC including design, code (including modules and functions) and test (traceability from test to detail design, high level design and to software requirements). Moreover the **Guidance for the Content of Premarket Submissions for Software Contained in Medical Devices** state that explicit traceability must exist among requirements, specifications, identified hazards and mitigations and among verification and validation testing.

Figure 1 summarises the traceability requirements from standards and guidelines through each stage of the software development lifecycle and includes risk and change management. The transition letters A to F are explained in Table 3 immediately following Figure 1.

Figure 1: Requirements for Traceability through the SDLC, Risk management and Change management processes as defined in International Standards

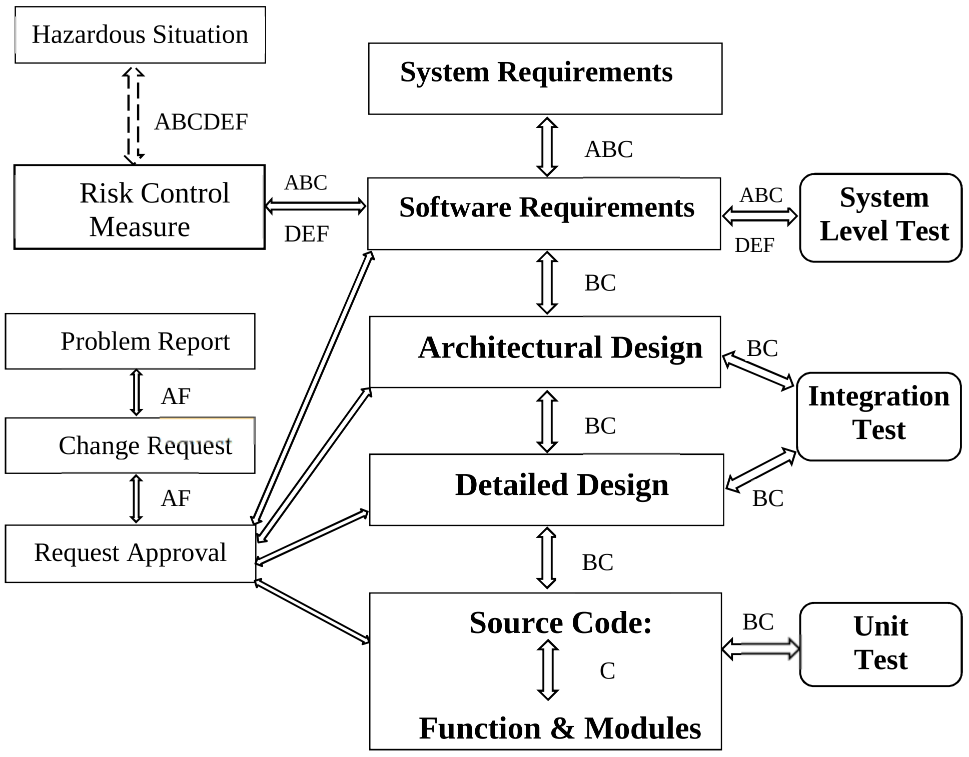

Table 3: Traceability link reference

A	IEC 62304:2006 Medical Device Software Lifecycle Processes
B	FDA 2005: Guidelines on the Content of Premarket Submissions
C	FDA 2002: General Principles of Software Validation (GPSV)
D	FDA 1999: Off the Shelf Software Use in Medical Devices
E	ISO 14971:2007 Application of Risk Management to Medical Devices
F	IEC/TR 80002-1:2009 Guidance on the application of ISO 14971 to Medical Device Software

Risk management helps ensure safe software. Risk analysis results must be converted into safety related requirements. Traceability ensures this conversion. A risk treatment is the treatment decision for a risk and can include risk reduction, risk avoidance, risk transfer and risk retention decisions [30]. The requirement for traceability through the risk management process from the hazardous situation through to risk control measures (including risk analysis and evaluation) and to verification of the risk control measures is clearly evident from each of the medical device standards and guidelines. The dashed bi-directional arrow between hazardous situation and risk control measure indicates that there are additional steps between these stages. These additional steps are given different names, depending on which standard you refer to. For example, IEC 62304 states that the manufacturer should document traceability a) from the hazardous situation to the software item; b) from the software item to the specific software cause; c) from the software cause to the risk control measure; and d) from the risk control measure to the verification of the risk control measure. On the other hand ISO 14971 states that the risk management file shall provide traceability for each identified hazard to a) the risk analysis; b) the risk evaluation; c) the implementation and verification of the risk control measures.
The ability to trace change is good practice for generic software development but of necessity for medical device software development. The necessity for change management is emphasised in **ANSI/AAMI/IEC 62304** when it states that the manufacturer shall create an audit trail whereby each a) Change request b) Problem report and c) Approval of change request, can be traced. It further requires that approved change requests are made traceable to the actual modification and verification of the software. While none of the other standards directly refer to a requirement for traceability through change management, some documents such as **IEC/TR 80002-1:2009** (6.6 Risks arising from Risk Control Measures) state that *'A rigorous software configuration management process, including rigorous change control (see Sub clause 8.2 of IEC 62304:2006), is essential'.*
Another interesting point arising from Figure 1 is that only the FDA's 'General Principles of Software Validation' and 'Guidance on the Content of Premarket Submission' documents directly refer to traceability from software requirements through design and code and onto test, with only the GPSV requiring traceability to

the function and module level. The GPSV document (5.2.4 Construction or Coding) states that '*a source code traceability analysis should be conducted and documented to verify that: modules and functions implemented in code can be traced back to an element in the software design specification and to the risk analysis.'*
As traceability is central to producing 'safe' software, we found it somewhat surprising that standards such as IEC 62304 do not stipulate the requirement for traceability through the design and implementation phases. So we asked ourselves the following question; 'By not stipulating the requirement for traceability through the design and implementation phases, does IEC 62304 imply that traceability through these phases is not necessary?' In attempting to answer this question we introduce some references taken from IEC 62304 which state that '*a standard is an important reference in responsible decision-making, but it should never replace responsible decision-making'* and '*a standard is truly useful only when it is used in conjunction with other sources of information and policy guidance and in the context of professional experience and judgment.* Therefore our understanding is that IEC 62304 details the minimum traceability requirements necessary and that although those minimum requirements don't detail traceability through the design and implementation phases, it is good software engineering practice to trace through these phases and the decision to implement tracing through these phases is left to responsible decision-making and professional experience and judgement of the developer. It is worth noting however that IEC 62304 does explicitly state the need for traceability between the software requirements and the system requirements, and also between the software requirements and the system level tests which ensures that the software component of the medical device is not developed in isolation.
Finally Figure 1 indicates that IEC 62304 does not require the level of traceability through the SDLC that the FDA guidance documents do. The primary reason for this is that 62304 is intended to be the minimum set of requirements considered necessary for safety of the medical device software. The FDA requirements are both for safety and efficacy. In other words, IEC 62304 is not particularly concerned about whether the software works as specified as long as it is safe, while the FDA is concerned about both whether the software works and whether it is safe. An example is that IEC 62304 does not require detailed design for software of class A, while FDA expects detailed design of all software.

5 Med-Trace assessments and findings

5.1 Development of the Med-Trace Assessment Method

Due to the safety critical nature of medical device software, a company must meet 'country specific' regulatory requirements in order to market their product in that country. An effective traceability process is a crucial requirement to achieving regulatory compliance. Due to a lack of specific guidance within the medical device standards and documentation, achieving an effective traceability process is problematic, resulting in many medical device companies engaging inefficient traceability processes [23]. Consequently, a method (known as Med-Trace [23]) of assisting medical device software companies to improve their traceability processes and to to adhere to the traceability aspects of the medical device software standards (as detailed in section 4) was developed. Med-Trace is a lightweight software traceability process assessment and improvement method for the medical device industry. Med-Trace is based on traceability best practices emanating from software engineering process models (CMMI_R, ISO/IEC 15504-5), software engineering traceability literature and medical device software standards and guidelines (i.e. the traceability practices that are referenced in table 1). Each of the standards and guidelines referenced in Table 3 were used to script the Med-Trace questions and the same questions were asked of both organisations. For a detailed description on the development of Med-Trace and on the observations of the case studies, the reader is referred to Casey and McCaffery [31]. A sample of the scripted MED-TRACE questions along with their source are listed in Table 4. For a greater sample of these questions the reader is referred to book chapter [21]**.**

Table 4: Sample Scripted Med-Trace Questions

Question	Source- Traceability Literature	Source- Medical Device Standards
Is there a documented procedure in place for traceability? Is training provided on traceability and to what extent is explicit knowledge made available on software traceability?	Ramesh (1998)	
How is traceability established between System Requirements, Software Requirements, and Software System testing?		Section 5.1.1 (ANSI/IEC 62304:2006,)
How is traceability undertaken from the software related hazards and the software risk control measures to the corresponding safety-related software requirements and the software items that satisfy those requirements?		Section 3.5 (ISO 14971:2007)

5.2 Med-Trace Implementation and observations

This section discusses how the Med-Trace assessment method was implemented in two medical device organis ations and the resulting observations. The objectives of the case studies were to demonstrate how Med-Trace could be used to assess the current status of the software traceability processes and to discover the main problems that medical device software development organisations face in terms of traceability. Med-Trace was implemented in two Small to Medium Sized (SME) medical device companies. Both companies (one based in Ireland and the other in the UK) developed electronic based medical devices that require compliance with both the FDA and the MDD.

The Med-Trace assessment method contains the following eight stages:

1. A preliminary meeting between the assessment team and the company wishing to undergo a Med-Trace assessment takes place;

2. The lead assessor provides an overview of the Med-Trace assessment to members of the organisation;

3. A review is undertaken of existing project documentation;

4. Staff from the organisation with responsibility for traceability are interviewed;

5. The assessors develop the findings report;

6. The findings report is presented to the assessed company;

7. The assessment team and the assessed organisation collaboratively develop an improvement pathway towards achieving highly effective and regulatory compliant traceability practices. The assessed organisation is responsible for implementing this pathway;

8. Three months after the initial assessment, the assessors reassess the company and review progress against the recommended improvement path. A final report is also produced.

As a result of the assessment on Medical Electronic (a pseudonym) the following recommendation was made; "The requirement for tracing open bugs/known issues to the safety/hazard/risk management system and linking them to the requirements will be addressed. This will be achieved by the introduction of an effective mechanism and a documented procedure "[31]. This recommendation implies no traceability during the risk management process which is a requirement of medical device standards and guidelines and is detailed in Figure 1. Additionally the second assessment, applied to North Medical UK (a pseudonym) recommended, "a documented procedure will be developed and implemented to facilitate mapping from the design documentation to the software code" [31]. This recommendation implies that no traceability exists between design and code phases, which is also a requirement of medical device standards and guidelines and is detailed in Figure 1. However, while tracing to code is an FDA requirement it is no surprise that the Med-Trace assessment has highlighted it's non implementation, as in communications we have had with the FDA they have stated that "In the almost 20 years I've been at FDA, I have never seen a mfr make links to source code. I used to do this working for xxxx 30 years ago".

The recommendations from both organisations indicate that neither was fully 'trace compliant'.

In more general terms the following observations were made across both organisations:

- A member of management was responsible for implementing traceability and its importance in medical device software development was recognised and understood;
- Tracing requirements and managing risk was recognised as difficult and complex;
- There is a lack of detailed guidance on how to implement traceability;
- Their process for software development with regard to traceability needed to be improved and formulised;
- The requirement for relevant training and the ability to record and leverage best practice with regard to traceability also emerged;
- The need for automated tools to manage traceability was recognised as was the serious limitation of using manual tools;
- Financial constraints needed to be considered when adapting automated tools.

Both organisations considered Med-Trace to be worthwhile and very relevant and appreciated the fact the Med-Trace is lightweight. The findings report addressed key areas where improvements were required and both organisations agreed to adapt the resultant traceability process improvement plan and to be reassessed.

6 Conclusions and Future Work

An effective traceability process is essential when developing medical device software due to its safety critical nature. Although the medical device regulations don't directly refer to requirements traceability through the SDLC, the requirement for effective traceability is mandated by the medical device standards and guidelines and its importance is evident from the number of times traceability is referred to in these standards and guidelines.
While the standards and guidelines mandate traceability, there is diversity in the level of traceability required. For example all of the above standards and guidelines require traceability through the risk management process from the hazardous situation through to risk control measures (including risk analysis and evaluation) and to verification of the risk control measures while only two of the standards explicitly require traceability through the change management process. In addition to this only two guidelines require traceability through each phase of the SDLC and to functions and modules in the source code. This diversity can make the implementation of an effective and compliant traceability process difficult and complex. IEC 62304 does not require the level of traceability through the SDLC that the FDA guidance documents do. The primary reason for this is that 62304 is intended to be the minimum set of requirements considered necessary for safety of the medical device software while the FDA requirements are both for safety and efficacy. Future work in this area is to develop a traceability process which will facilitate both organisations currently operating in the medical device software domain and organisations wishing to enter the medical device software domain to implement and maintain traceability in an efficient and compliant manner.
While effective traceability is mandated by the standards and its necessity was understood by the two organisations who participated in the Med-Trace assessment, there is a lack of detailed guidance in how to best implement an effective and compliant traceability process within the medical device software domain. There currently are a challenging number of standards governing medical device software development and to determine the exact traceability requirement from each of these standards can be time consuming. Med-Trace addresses these challenges by providing a light weight assessment method which may be used to diagnose an organisation's strengths and weaknesses in relation to traceability in their software development processes. To assess how compliant a company is in relation to traceability, Med-trace questions are based on the links indicated in Figure 1. Neither company who undertook the assessment were found to be fully compliant with tracing to code and tracing through the risk management process highlighted as issues needing to be addressed.
To-date, Med-Trace has been applied in two SME organisations and has been well received. It is envisaged that Med-Trace will continue to be refined based on on-going research and feedback from future assessments.
Future plans include a tool to automate Med-Trace with the objective of facilitating its national and international roll out and to encourage its wider use.

7 Acknowledgement

This research is supported by the Science Foundation Ireland (SFI) Stokes Lectureship Programme, grant number 07/SK/I1299, the SFI Principal Investigator Programme, grant number 08/IN.1/I2030 (the funding of this project was awarded by Science Foundation Ireland under a co-funding initiative by the Irish Government and European Regional Development Fund), and supported in part by Lero - the Irish Software Engineering Research Centre (http://www.lero.ie) grant 10/CE/I1855

8 References

[1] M.McHugh, F.McCaffery, V.Casey, Software process improvement to assist medical device software development organisations to comply with the amendments to the medical device directive, The Institution of Engineering and Technology, 6 (2011) 431-437.
[2] S.R. Rakitin, Coping with Defective Software in Medical Devices, in: Computer, 2006, pp. 40-45.
[3] A. Bond, A. Hacking, Z. Milosevic, A. Zander, Specifying and building interoperable eHealth systems: ODP benefits and lessons learned, Computer Standards & Interfaces, 35 (2013) 313-328.
[4] J. Burton, F. McCaffery, I. Richardson, A risk management capability model for use in medical device companies, in: Proceedings of the 2006 international workshop on Software quality, ACM, Shanghai, China, 2006, pp. 3-8.

[5] F.M. Caffery, A. Dorling, V. Casey, Medi SPICE: An update, in: International Conference on Software Process Improvement and Capability Determinations (SPICE), Pisa, Italy, 2010, pp. 195-198.
[6] T. Tasić, U. Grottker, An overview of guidance documents for software in metrological applications, Computer Standards & Interfaces, 28 (2006) 256-269.
[7] E. Council, COUNCIL DIRECTIVE 93/42/EEC concerning medical devices, in: Official Journal of The European Communities, Luxembourg 1993.
[8] European Council, COUNCIL DIRECTIVE 2000/70/EC (Ammendment), in: Official Journal of the European Union, Luxembourg, 2000.
[9] European Council, COUNCIL DIRECTIVE 2001/104/EC (Ammendment), in: Official Journal of the European Union, Luxembourg, 2001.
[10] European Council, COUNCIL DIRECTIVE 2003/32/EC (Ammendment), in: Official Journal of the European Union, Luxembourg, 2003.
[11] European Council, COUNCIL DIRECTIVE 2007/47/EC (Ammendment), in: Official Journal of the European Union, Luxembourg, 2007.
[12] FDA, Off-The-Shelf Software Use in Medical Devices; Guidance for Industry, FDA Reviewers and Compliance, in, CDRH, Rockville, 1999.
[13] FDA, General Principles of Software Validation; Final Guidance for Industry and FDA Staff, in, CDRH, Rockville, 2002.
[14] FDA, Guidance for the Content of Premarket Submissions for Software Contained in Medical Devices, in, CDRH, Rockville, 2005.
[15] ISO/IEC, 12207:1995 Systems and software engineering — Software life cycle processes, in, ISO, Geneva, Switzerland, 1995.
[16] AAMI, AAMI SW68 Medical device software - Software life cycle processes, in, 2001.
[17] ANSI/AAMI/IEC, 62304:2006 Medical device software—Software life cycle processes, in, AAMI, Arlington, VA, 2006.
[18] O. Gotel, P. Mader, Acquiring Tool Support for Traceability, in: J. Cleland-Huang, O. Gotel, A. Zisman (Eds.) Software and Systems Traceability, Springer, 2012.
[19] O.C.Z. Gotel, C.W. Finkelstein, An analysis of the requirements traceability problem, in: Requirements Engineering, 1994., Proceedings of the First International Conference on Requirements Engineering, 1994, pp. 94-101.
[20] O. Gotel, J. Cleland-Huang, A. Zisman, Software and Systems Traceability, Springer, London Dordrecht Heidelberg New York, 2012.
[21] F. McCaffery, V. Casey, M. Sivakumar, G. Coleman, P. Donnelly, J. Burton, Medical Device Software Traceability, in: J. Cleland-Huang, O. Gotel, A. Zisman (Eds.) Software and Systems Traceability, Springer, 2012.
[22] A. Espinoza, J. Garbajosa, A Proposal for Defining a Set of Basic Items for Project-Specific Traceability Methodologies, in: Software Engineering Workshop, 2008. SEW '08. 32nd Annual IEEE, Madrid, 2008, pp. 175-184.
[23] V. Casey, F.M. Caffery, Med-Trace: Traceability Assessment Method for Medical Device Software Development, in: EuroSPI 2011, Denmark, 2011.
[24] A. Kannenberg, D.H. Saiedian, Why Software Requirements Traceability Remains a Challenge, CrossTalk The Journal of Defense Software Engineering, (July/Aug 2009).
[25] P. Mason, On Traceability for Safety Critical Systems Engineering, in: Proceedings of the 12th Asia-Pacific Software Engineering Conference, IEEE Computer Society, 2005, pp. 272-282.
[26] FDA, Medical & Radiation Emitting Device Recalls, in, 2012.
[27] I. Lee, G.J. Pappas, R. Cleaveland, J. Hatcliff, B.H. Krogh, P. Lee, H. Rubin, L. Sha, High-Confidence Medical Device Software and Systems, in: Computer Society, IEEE Computer Society, 2006, pp. 33-38.
[28] ISO, ISO 14971:2007 Medical devices — Application of risk management to medical devices, in, ISO, Switzerland, 2007.
[29] IEC, IEC/TR 80002-1:2009 Medical Device Software Part 1: Guidance on the application of ISO 14971 to medical device software, in, ISO, Switzerland, 2009.
[30] R. Matulevi˘cius, H. Mouratidis, E.D. Nicolas Mayer, P. Heymans, Syntactic and Semantic Extensions to Secure Tropos to Support Security Risk Management, Journal of Universal Computer Science,, 18 (2012).
[31] V. Casey, F.M. Caffery, A lightweight traceability assessment method for medical device
software, JOURNAL OF SOFTWARE EVOLUTION AND PROCESS, (2011).

Author CVs

Gilbert Regan

Gilbert Regan is a PhD student at Dundalk Institute of Technology. He is a member of the Regulated Software Research Group in Dundalk Institute of Technology and a member of Lero. His main research subject is Software Process Improvement with particular focus on Traceability within Medical Device Software.

Fergal Mc Caffery

Dr Fergal Mc Caffery is a Lecturer with Dundalk Institute of Technology. He is the leader of the Regulated Software Research Group in Dundalk Institute of Technology and a member of Lero. He has been awarded Science Foundation Ireland funding through the Stokes Lectureship, Principal Investigator and CSET funding Programmes to research the area of software process improvement for the medical device domain. Additionally, he has received EU FP7 and Enterprise Ireland Commercialisation research funding to improve the effectiveness of embedded software development environments for the medical device industry

Kevin McDaid

Dr Kevin Mc Daid is a lecturer in Computing at Dundalk Institute of Technology. He leads research in the area of Spreadsheet Engineering in the Software Technology Research Centre. His background is in Mathematics and Statistics and his research focus is on the application of statistical methods to decision problems in software development.

Derek Flood

Dr Derek Flood is a postdoctoral research fellow with the Regulated Software Research Group at Fundal Institute of Technology (DkIT). Derek has recently worked as a post-doctoral research assistant at Oxford Brookes University on the PACMAD project, examining the usability issues associated with mobile applications. His current research interests include Software process improvement for Medical Devices, Spreadsheet risk, Usability of mobile applications, and the psychological impact of mobile applications on end users.