A Traceability Process Assessment Model for the Medical Device Domain

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

{gilbert.regan, fergal.mccaffery, kevin.mcdaid,
derek.flood}@dkit.ie
Miklos.biro@sch.at

Abstract. Traceability of requirements through the software development lifecycle (including supporting processes such as risk management and change management) is a difficult and expensive task. The implementation of effective traceability allows organizations to leverage its many advantages, such as impact analysis, product verification and validation, and facilitation of code maintenance. Traceability is conducive to producing quality software.

Within the medical device domain, as in other safety critical domains, software must provide reliability, safety and security because failure to do so can lead to injury or death. However, despite its criticality most software systems don't employ explicit traceability between artefacts. Numerous barriers hamper the effective implementation of traceability such as cost, complexity of relationship between artefacts, calculating a return on investment, different stakeholder viewpoints, lack of awareness of traceability and a lack of guidance as to how to implement traceability.

To assist medical device organisations in addressing the lack of guidance on how to implement effective traceability, this paper aims to present the development of a traceability process assessment model and how traceability process assessment and maintenance could be fully automated using the Open Services for Lifecycle Collaboration (OSLC) initiative. The process assessment model will allow organisations to identify strengths and weaknesses in their existing traceability process and pinpoint areas for improvement.

Keywords: Requirements traceability, Traceability assessment, Medical device, Safety critical, Process assessment, Automation

1 Introduction

Medical device software is considered safety critical, meaning that failure in the software can result in loss of life, significant environmental damage, or major financial loss [1], therefore medical device software must provide reliability, safety and security. Manufacturers must ensure their software is safe and establish effective software development processes that are based on recognized engineering principles

adfa, p. 1, 2011.

appropriate for safety critical systems. At the heart of such processes, they must incorporate traceability.

Traceability is the ability to establish links (or traces) between source artefacts and target artefacts [2]. In addition to tracing requirements through each phase of the software development lifecycle (SDLC) the medical device standards and guidelines also require traceability through the supporting processes of risk management and change management. Implementing traceability through risk management helps ensure that risk control measures for identified hazards have been implemented and tested. Similarly, implementing traceability through the change management process helps ensure that changes in the software, agreed as a result of problem reports or user requests, have been implemented and tested.

Traceability is a requirement of many regulatory bodies such as the Federal Aviation Administration who specify in their DO-178C standard [3] that "software developers must be able to demonstrate traceability of designs against requirements" at each stage of the development. The Food and Drug Administration (FDA) state that documentation provided in a submission for approval should "provide traceability to link together design, implementation, testing, and risk management" [4]. The automobile safety standard ISO 26262:2011 [5] states that "safety requirements shall be traceable…to: each source of a safety requirement at the upper hierarchical level, each derived safety requirement at a lower hierarchical level, or to its realization in the design, and the specification of verification".

However despite its many benefits and regulatory requirements, most existing software systems lack explicit traceability links between artefacts [6]. Numerous reasons have been identified for reluctance in implementing traceability including cost, complexity, building a requirements trace matrix (RTM) is time consuming, arduous and error prone [7], stakeholders having differing perceptions as to the benefits of traceability [1], developers may fear that traces could be used to monitor their work [8], and difficulties with trace tools [9]. Finally almost no guidance is available for practitioners to help them establish effective traceability in their projects and as a result, practitioners are ill-informed as to how best to accomplish this task [10, 11].

To assist medical device organisations in addressing the lack of guidance on how to implement effective traceability, this paper presents the development and validation of a traceability process assessment model (PAM). To be effective, organisations need to know how well their current traceability process helps them achieve their goals. Additionally an assessment of a process will lead to an increased understanding of the actual performance and management of activities, and the potential for improvement.

The remainder of this paper is structured as follows: Section 2 outlines current assessment models' relationship to traceability and the need to automate the assessment and maintenance of traceability. Section 3 outlines current assessment of traceability in medical device standards and guidelines and assessment models such as ISO 15504 [12]. Section 4 outlines the methodology used to develop the PAM while section 5 details the structure of the developed PAM. Section 6 discusses how traceability assessment and maintenance could be automated using the Open Services for Lifecycle Collaboration (OSLC) initiative. Finally section 7 concludes the paper.

2 Related Work

A literature review was conducted to determine what other traceability assessment models were available in the general, safety critical or medical device domains. This review returned only one model on traceability compliance/ capability assessment called Med-Trace [10]. Med-trace is a lightweight traceability assessment method, completed in 8 stages, whose goal is to assist medical device organizations to improve their software development traceability process. The authors completed assessments on two medical device companies and were able to identify areas for improvement in each company's traceability process.

There are a number of process assessment models which provide common frameworks for assessing software process capability. These models include ISO 15504 SPICE , Automotive SPICE [13], SPICE 4 SPACE [14], and the Capability Maturity Model CMMI [15] among others. These frameworks assess processes such as software design process, software construction process, software testing process etc. However the frameworks do not include a dedicated traceability assessment process. The frameworks do include traceability assessment but it is spread out across a lot of processes and sometimes difficult to interpret (as detailed in section 3-1) e.g. base practice 4 of the software construction process (Eng. 6) in SPICE states;

"*Verify software units. Verify that each software unit satisfies its design requirements by executing the specified unit verification procedures and document the results*".

Explicit traceability is not required in the above statement but it may be implied. It is open to interpretation.

It is important to highlight that traceability has been considered as a key issue by the agile community as well. Scott Ambler, one of the key personalities of the agile movement, states in 1999 that "My experience shows that a mature approach to requirements traceability is often a key distinguisher between organizations that are successful at developing software and those that aren't. Choosing to succeed is often the most difficult choice you'll ever make—choosing to trace requirements on your next software project is part of choosing to succeed." [16]

The same Scott Ambler's advice in 2013 [17]:

"Think very carefully before investing in a requirements traceability matrix, or in full lifecycle traceability in general, where the traceability information is manually maintained. When does maintaining traceability information make sense?

- Automated tooling support exists
- Complex domains
- Large teams or geographically distributed teams
- Regulatory compliance"

While the above view reflects the reluctance in implementing traceability as discussed in the introduction, it also shows its importance in the case of the medical device domain being both complex and subject to regulatory compliance requirements.

Considering all of the above discussion, the need for the automation of assessing and maintaining traceability is imminent. It is this automation to which the Open Services

for Lifecycle Collaboration (OSLC) initiative opens the way also as discussed in this paper.

3 Software Process Assessment

A Software process provides a framework for the key activities of software development. Good management of the process should provide for a sustained orderly improvement of the process. Software process assessment assist organizations in understanding the current state of their software process by identifying strengths and weaknesses in their process and thus providing focus on areas for improvement. In addition to assessing their own process an organization can use software process assessment to determine the state of a supplier's process.

3.1 Traceability assessment

To understand how traceability is currently assessed, four software process improvement frameworks, and the medical device standards and guidelines, have been analysed for their requirements for traceability through the SDLC.

The results of this analysis are shown overleaf in Table 1. Figure 1 is a depiction of the SDLC, with the numbered double head arrows indicating bi-directional traceability between the different phases and between the phases and test. These numbers are represented in the first column of Table 1. The assessment models and documents analysed were;

- Capability Maturity Model Integration (CMMI)
- ISO/IEC 15504-5 Process assessment model
- Automotive SPICE Process assessment model
- SPICE 4 SPACE Process assessment model
- Medical device standards and guidelines documents
    A. IEC 62304 - Medical device Software-Software lifecycle processes
    B. FDA - General Principles of Software Validation (GPSV)
    C. FDA - Guidance for Premarket Submissions for Software in Medical Devices
    D. FDA - Guidance on Off-The-Shelf Software Use in Medical Devices
    E. ISO 13485 - Medical devices — Quality management systems
    F. ISO 14971 - Application of risk management to medical devices

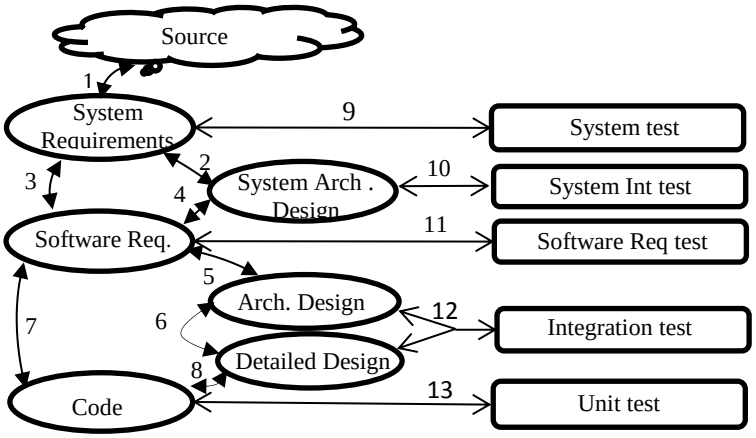

Fig. 1. SDLC and links between phases @Annex E of the Automotive SPICE® PAM

Table 1. Traceability links across different assessment models

Link	Medical Device Standards	15504 SPICE	Auto SPICE	SPICE 4 SPACE	CMMI
1	E - 7.3.2	ENG 2 BP 5	ENG 2 BP 6	ENG2 BP 5	RD – SG2
2	E - 7.1(d)	ENG 3 BP 6	ENG 3 BP 6	ENG3 BP 6	REQM - SP 1.4
3	A	ENG 4 BP 4	ENG 4 BP 6	ENG 4 BP 4	REQM - SP 1.4
4	E	ENG 9 BP 6	ENG 4 BP 7	ENG 9 BP6	REQM - SP 1.4
5	B	ENG 5 BP 5	ENG 5 BP 9	ENG 5 - BP 6	REQM - SP 1.4
6	A	ENG BP3/5	ENG 5 BP10	ENG 5 - BP 3	REQM - SP 1.4
7	C	ENG 6 BP 3	ENG 6 BP 9	ENG 6 - BP 3	REQM - SP 1.4
8	B	ENG 6 BP 3	ENG 6 BP 8	ENG 6 - BP 3	REQM - SP 1.4
9	E - 7.3.5	ENG 1 BP 1	ENG 10 BP 5	ENG 10-BP 1	REQM - SP 1.4
10	E - 7.3.5	ENG 9 BP 2	ENG 9 BP 7	ENG 9 - BP 2	REQM - SP 1.4
11	A	ENG 8 BP 1	ENG 8 BP 5	ENG 8 - BP 1	REQM - SP 1.4
12	B	ENG 7 BP 2	ENG 7 BP 7	ENG 7 - BP 2	REQM - SP 1.4
13	B	ENG 6 BP 4	ENG 6 BP10	ENG 6 - BP 4	REQM - SP 1.4

Table 1 indicates that each of the traceability links are required through an assortment of the medical device standards and guidelines and that each of the assessment models requires traceability for each link. However a difficulty arises with understanding the clarity of the requirement for traceability, with some models somewhat open to interpretation. For example Automotive SPICE is very definite and clear about the tracea-

bility links required whereas CMMI is more general. This point can be illustrated by looking at the requirement for link 4;

Base practice 4 of the Software requirements analysis process (ENG 4) in Automotive SPICE states '*Ensure consistency and bilateral traceability of system architectural design to software requirements*' whereas CMMI states '*Maintain requirements traceability from a requirement to its derived requirements and allocation to functions, interfaces, objects, people, processes, and work products'*. This CMMI statement takes some interpretation and it is the view of this study that this statement covers all links from 2 to 13.

The difficulty with understanding the requirements for traceability in the frameworks is further compounded by the fact that the traceability requirements in each of the assessment models are spread out across many processes so extracting the requirements is a time consuming task. A point of note from Table 1 is that the medical device standards' requirement for traceability is matched by the traceability requirements from the improvement frameworks, therefore it is envisaged that the assessment model developed as part of this study, with slight modifications should be easily transferable to other domains.

4 Research Methodology

The traceability process assessment model is based on the ISO 15504-2 [18]. It was decided to base the traceability assessment model on ISO/IEC 15504 as this improvement and capability determination model was derived from ISO/IEC 12207 [19] and since ANSI/AAMI/IEC 62304:2006 (Software lifecycle processes for medical device software) is derived from ISO/IEC 12207 it was determined that there was good synergy between ANSI/AAMI/IEC 62304:2006 and ISO/IEC 15504. Additionally, 15504 is used extensively in other safety critical industries such as the automotive industry (Automotive SPICE), space industry (SPICE 4 SPACE) and the medical device industry (Medi SPICE).

The first stage was to develop a traceability PRM. The PRM was developed using the requirements from traceability (taken from the medical device standards and guidelines), and ISO 15504-2 section 6.2 which sets out the requirements for a Process Reference Model. While ISO 15504-2 details the minimum requirements that a PRM and a PAM should meet, it provides no guidance on how to develop the models i.e. it does not tell you how to transform requirements into a PRM or PAM. To address this issue, this study based the development of the PAM on the Tudor IT Service Management Process Assessment (TIPA) transformation process. The TIPA transformation process complies with the requirements for PRMs and PAMs as expressed in ISO/IEC 15504-2. The transformation process contains the following steps [20];

1. Identify elementary requirements in a collection of requirements
2. Organise and structure the requirements
3. Identify common purposes upon those requirements and organize them towards domain goals

4. Identify and factorize outcomes from the common purposes and attach them to the related goals
5. Group activities together under a practice and attach it to the related outcomes
6. Allocate each practice to a specific capability level
7. Phrase outcomes and process purpose
8. Phrase the Base Practices attached to Outcomes
9. Determine Work Products among the inputs and outputs of the practices

5 Structure of Traceability PAM

The traceability assessment framework, illustrated in Figure 2, consists of 4 traceability processes which are Change Management (CM) traceability, Risk Management (RM) traceability, Software Development Lifecycle (SDLC) traceability, and Best Practice traceability.

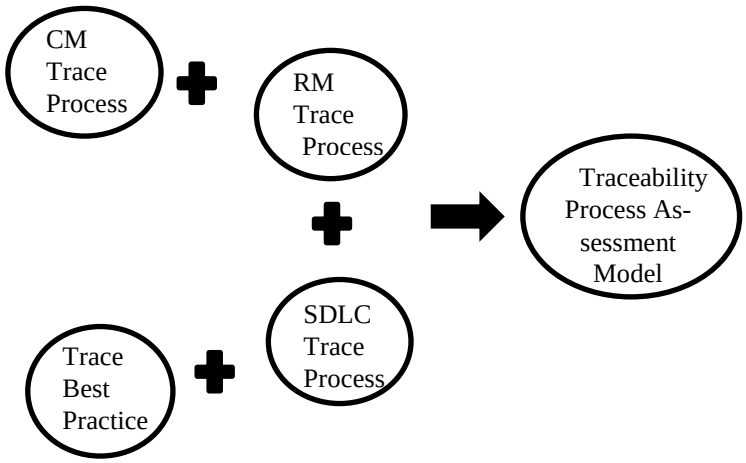

Fig. 2. Traceability Process Assessment Framework

Each of the processes contains: (i) Title; (ii) Purpose, which contains the unique functional objectives of the process when performed in a particular environment; (iii) Outcomes, which are a list of expected positive results of the process performance; (iv) Base practices, whose performance provides an indication of the extent of achievement of the process purpose and process outcomes; and (v) Work Products (WPs) are either used or produced (or both), when performing the process.

The CM traceability process: The purpose of this process is to ensure that traceability is adequately addressed throughout all stages of the Change management/Problem resolution process by assessing the following application of bi-directional traceability: between each Problem Report (PR) and Change Request (CR); between each CR and its analysis and evaluation; between approval of CR and identification of software modification; between each denial of CR/PR and reason for denial; between each identified software modification and its implementation and verification; and between each modification implementation and regression testing.

The RM traceability process: The purpose of this process is to ensure that traceability is adequately addressed throughout all stages of the risk management process by assessing the following application of bi-directional traceability: between analysis of risk to the identification of hazards; between hazardous situation and software item; between software item and specific software cause; between each hazard to estimation of risk of each hazard; between each risk estimation to evaluation of acceptability of the risk; between hazards and identification and implementation of risk control measures; between implementation and verification of risk control measures; and between residual risk to assessment of acceptability of those risks.

The SDLC traceability process: The purpose of the SDLC Traceability Process is to ensure that traceability is adequately addressed throughout all stages of the SDLC process by assessing the following application of bi-directional traceability: between software requirements and system requirements; between software requirement and software architectural and software detailed design; between software detailed design and source code; between software requirements and source code; and between each phase of the SDLC and test for that phase.

Traceability best practice process: The purpose of the Traceability Best Practices process is to ensure that traceability best practices are established when implementing traceability through the SDLC and the supporting processes of risk management and change management. This is achieved by assessing if a company policy and a standard operating procedure for traceability have been developed, the resources required for successful traceability implementation are made available, and the appropriate techniques for successful implementation are deployed.

6 Automation of Traceability Assessment and Maintenance: the Future of Traceability Best Practices

As discussed in section 2, there is imminent need for the automation of traceability assessment and maintenance. Considering the clear definition cited in the introduction, traceability is the ability to establish links (or traces) between source artefacts and target artefacts [2]. According to the state of the art of web technology, we have today the means to identify and to establish links between immense numbers of artifacts which can even be seamlessly traced on the basis of these links.

Our vision is that the processes defined in the Traceability Process Assessment Model of this paper could be executed using a system accessing all of the necessary artifacts which would be accessible on the web (internet or intranet). By consequent, this system would ultimately have full traceability assessment and also resulting traceability maintenance capability.

Application Lifecycle Management (ALM) tool vendors are perfectly aware of this need, and some of the tools [21] contain features supporting a given level of automation. However, current ALM tools have following inherent weaknesses:

- Traceability is basically restricted to the closed ALM system. APIs are available for providing internal data, however, no standardized open form of exchange was made possible before the below discussed OSLC initiative.

- Useful traceability reports can be generated, but they are static while requirements and identified defects are very dynamically changing artefacts, and may even originate from outside the ALM system.
- Assessors and users may be easily confused by the complexity of the set of widgets, such as buttons, text fields, tabs, and links which are provided to access and edit all properties of resources at any time.
- Assessors and users need to reach destinations such as web pages and views by clicking many links and tabs whose understanding is not essential for the assessment.

Open Services for Lifecycle Collaboration (OSLC) is the recently formed cross-industry initiative aiming to define standards for compatibility of software lifecycle tools. Its aim is to make it easy and practical to integrate software used for development, deployment, and monitoring applications. This aim seems to be too obvious and overly ambitious at the same time. However, despite its relatively short history starting in 2008, OSLC is the only potential approach to achieve these aims at a universal level, and is already widely supported by industry.

The unprecedented potential of the OSLC approach is based on its foundation on the architecture of the World Wide Web, which is unquestionably proven to be powerful and scalable, and on the generally accepted software engineering principle to always focus first on the simplest possible things that will work.

The elementary concepts and rules are defined in the OSLC Core Specification which sets out the common features that every OSLC Service is expected to support using the terminology and generally accepted approaches of the World Wide Web Consortium (W3C). One of the key approaches is Linked Data being the primary technology leading to the Semantic Web which is defined by W3C as providing a common framework that allows data to be shared and reused across application, enterprise, and community boundaries.

The OSLC Core Specification is actually the core on which all lifecycle element (domain) specifications must be built upon. Examples of already defined OSLC Specifications include:

- Architecture Management
- Asset Management
- Automation
- Change Management
- Quality Management
- Requirements Management

Let us focus for example on the Change Management Specification which is of particular interest in the Traceability PAM discussed in this paper. Its version 3.0 is under development in 2014, and builds of course on the Core, briefly mentioned above, to define the resource types, properties and operations to be supported by any OSLC Change Management (OSLC CM) provider.

Examples of possible OSLC CM Resources include defect, enhancement, task, bug, activity, and any application lifecycle management or product lifecycle man-

agement artifacts. Resource types are defined by the properties that are allowed and required in the resource.

The properties defined in the OSLC Change Management Specification describe these resource types and the relationships between them and all other resources. The relationship properties describe in most general terms for example that

- the change request affects a plan item
- the change request is affected by a reported defect
- the change request tracks the associated Requirement
- the change request implements associated Requirement
- the change request affects a Requirement

7 Conclusion

To assist medical device organizations improve their traceability, a traceability assessment model has been developed. This model, which consists of four processes, is based on the ISO 15504 structure and used the TIPA transformation process for development. By assessing for all traceability requirements from the medical device standards and guidelines and by assessing for traceability implementation best practices, this traceability assessment model will assist medical device organisations understand their actual traceability performance and management of activities, and the potential for improvement. It will also allow an organisation assess the state of a supplier's traceability process.

If our envisioned system, based on the processes defined in the Traceability Process Assessment Model of this paper, could seamlessly access the resources and their relationships using OSLC across all tools applied in the entire software development lifecycle (SDLC), then traceability process assessment and maintenance could be fully automated.

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.

References

1. Kannenberg, A., Saiedian, D.H.: Why Software Requirements Traceability Remains a Challenge. CrossTalk The Journal of Defense Software Engineering 5 (2009)

2. Gotel, O., Mader, P.: Acquiring Tool Support for Traceability. In: Cleland-Huang, J., Gotel,

O., Zisman, A. (eds.) Software and Systems Traceability. Springer, London Dordrecht Heidelberg New York (2012)

3. FAA: DO-178C, Software Considerations in Airborne Systems and Equipment Certification. RTCA (2012)
4. FDA: Guidance for the Content of Premarket Submissions for Software Contained in Medical Devices. CDRH, Rockville (2005)
5. ISO: 26262: Road Vehicle. Functional Safety. (2011)
6. Lucia, A.D., Marcus, A., Oliveto, R., Poshyvanyk, D.: Information Retrieval Methods for Automated Traceability Recovery. In: Cleland-Huang, J., Gotel, O., Zisman, A. (eds.) Software and Systems Traceability, pp. 88 - 111. Springer (2012)
7. Cleland-Huang, J.: Just Enough Requirements Traceability. Proceedings of the 30th Annual International Computer Software and Applications Conference - Volume 01, pp. 41-42. IEEE Computer Society (2006)
8. Jarke, M.: Requirements tracing. Commun. ACM 41, 32-36 (1998)
9. Regan, G., Mc Caffery, F., Mc Daid, K., Flood, D.: The Barriers to Traceability and their Potential Solutions: Towards a Reference Framework. 38th Euromicro Conference on Software Engineering and Advanced Applications, pp. 319- 322. IEEE, Cesme, Turkey (2012)
10. McCaffery, F., Casey, V.: Med-Trace: Traceability Assessment Method for Medical Device Software Development. EuroSPI pp. 1.1 - 1.8, Denmark (2011)
11. Mader, P., Gotel, O., Philippow, I.: Motivation Matters in the Traceability Trenches. Proceedings of the 2009 17th IEEE International Requirements Engineering Conference, RE, pp. 143-148. IEEE Computer Society (2009)
12. ISO/IEC: 15504-5 : An exemplar Process Assessment Model. ISO, Switzerland (2006)
13. SIG, A.: Automotive SPICE® Process Assessment Model. (2010)
14. ECCS: Space Product Assurance- Software process assessment and improvement – Part 2: Assessor instrument. ESA Requirements and Standards Division, Netherlands (2010)
15. Institute, S.E.: CMMI® for Development, Version 1.3. Improving processes for developing better products and services, (2010)
16. Ambler, S.: Tracing Your Design. Dr.Dobb's Journal: The World of Software Development (1999)
17. Ambler, http://www.agilemodeling.com/essays/agileRequirementsBestPractices.
18. ISO/IEC: 15504-2: Process assessment — Performing an assessment. ISO, Switzerland (2003)
19. ISO/IEC: 12207: Systems and software engineering — Software life cycle processes. ISO, Geneva, Switzerland (2008)
20. Barafort, B., Renault, A., Picard, M., Cortina, S.: A transformation process for building PRMs and PAMs based on a collection of requirements – Example with ISO/IEC 20000.

SPICE, Nuremberg, Germany (2008)

21. Gartner, http://www.techostan.com/docs/quadrant.pdf