Investigation of Traceability within a Medical Device Organization

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

Dundalk Institute of Technology , Dundalk , Ireland
{gilbert.regan, fergal.mccaffery, kevin.mcdaid, derek.flood}@dkit.ie

Abstract. Requirements traceability helps to ensure software quality. It supports quality assurance activities such as impact analysis, regression test selection, compliance verification and validation of requirements. Its implementation has long been promoted by the research and expert practitioner communities. However, evidence indicates that few software organizations choose to implement traceability processes, in the most part due to cost and complexity issues. Organizations operating within the safety critical domains are mandated to implement traceability, and find the implementation and maintenance of an efficient and compliant traceability process a difficult and complex issue.

Through interviews with a medical device SME, this paper seeks to determine how traceability is implemented within the organization, the difficulties it faces in implementing traceability, how compliant it is with the medical device standards and guidelines, and what changes could be made to improve the efficiency of their traceability implementation and maintenance.

Keywords: traceability, requirements traceability, software traceability, medical device, software process improvement

1 Introduction

The importance and role of traceability in supporting software development have been long recognised [1]. Requirements tracing helps ensure that the customers' requirements are being met and that the quality of the final product is maximised while minimising costly rework due to errors in the requirements. Traceability through the software development and risk management process is particularly important in safety critical industries such as the medical device domain, where it is incumbent on manufacturers to produce safe software [2].

The effect of poor quality medical device software can be seen from the recent announcement by Johnson & Johnson that some of its Animas insulin pumps will become defunct at the stroke of midnight on December 31, 2015. The company has warned European regulators that the pumps will stop delivering insulin at the start of 2016 and will generate a "call service alarm" due to a software fault. During November 2012 the company began to warn patients of the impending malfunction, offering to

replace devices that have warranties that will still be in effect when the software glitch is expected to hit [3].

Organizations building safety critical systems are often legally required to demonstrate that all parts of the code trace back to valid requirements. Laws such as the US Sarbanes-Oxley Act 2002 [4] require organizations to implement change management processes with explicit traceability coverage for any parts of a software product that potentially impact the balance sheet [5].

Regulations and guidelines exist to assist medical device organisations to produce quality software. The documents which medical device manufacturers must adhere to are IEC 62304 [6] (which is endorsed by the European Union and the U.S.), FDA Guidance for the Content of Pre-market Submission for Software in Medical Devices [7], FDA General Principles of Software Validation(GPSV) [8], FDA Off-the Shelf Software Use in Medical Devices [9] and ISO 14971 [10]. These documents emphasise to different degrees the requirement for traceability through the software development lifecycle (SDLC), risk management and change management processes. Understanding the different degrees of requirements for traceability and implementing those requirements is a difficult and complex task for a medical device small to medium enterprise (SME). A lack of detailed guidance and direction on how to implement and maintain traceability could lead to many medical device SMEs implementing inefficient and/or non-compliant traceability processes [11].

The objective of this paper is to analyse the requirements for traceability as detailed in the medical device standards and guidelines documents and to highlight the varying requirements for traceability between these documents. The authors conducted detailed interviews with a medical device SME in order to determine what processes are being used to implement traceability, difficulties faced in implementing traceability, how compliant the organisation is, and what changes could be made to improve the efficiency of traceability implementation and maintenance.

This paper has been divided into 7 sections. Related work in Section 2 explains the concept of traceability and reveals why its implementation is important. Section 3 details the requirements for traceability as prescribed by the medical device standards and guidelines. Section 4 list the aims of the study and the approach used to achieve those aims and also details the organisation profile. Section 5 summarises the implementation of traceability within the organisation and highlights difficulties faced by the organisation in implementing traceability. Section 6 discusses the difficulties that the organisation face in implementing traceability and ways in which it might overcome these difficulties and become more efficient.

2 Related Work

In engineering terms a trace is comprised of a source artifact, a target artifact and the link between them [12]. 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 "*[13].

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 many important aspects of a project such as; are the customers' requirements being met, the specific requirements that an artefact relates to, the origins and motivation of a requirement, and what are the requirements associated with this test case. Traceability supports critical activities such as compliance verification, impact analysis, and regression test case selection[14].Traceability helps ensure that 'quality' software is developed.

Traceability is about linking requirements to artefacts in the software development environment. This environment includes technical aspects (e.g. specifications, diagrams and code) and social aspects (e.g. people, policies, decisions etc.). 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. Traceability links represent an important source of information for project managers, analysts, designers, maintainers, and end users. Increasingly software development is globally distributed across multiple teams and sites which makes traceability even more important [11]. As traceability provides an essential support for developing high quality software systems [15] it is vital to engage an efficient traceability process.

Unfortunately, establishing and maintaining traceability links between software artefacts is a time consuming, error prone, and labour intensive task. Consequently, despite the advantages that can be gained, explicit traceability is rarely established unless there is a regulatory reason for doing so [16]. In safety critical domains such as the medical device domain traceability information can also be used when certifying a safety-critical product to show that all requirements were implemented and covered by specific tests.

3 Traceability Requirements for Compliance within Medical Device Standards

The requirements for traceability through the SDLC are not transparent from the medical device regulations; in fact the regulations make little or no reference to traceability. However the medical device standards and guidelines mandate traceability throughout the SDLC and within supporting processes such as change management and risk management.

Figure 1 indicates the requirements for traceability through the SDLC, Risk and Change management processes. The letters A to E refer to the medical device standards and guidelines and are listed in Table 1. The obvious point of note is the

variance in the requirements between the standards with only the FDA's General Principles of Software Validation document requiring traceability to the module and function level

Figure 2 indicates the medical device standards and guidelines requirements for traceability within the Change management and Risk management processes. While each of the standards require full traceability through the risk management process, only IEC 62304 requires traceability within the change management process

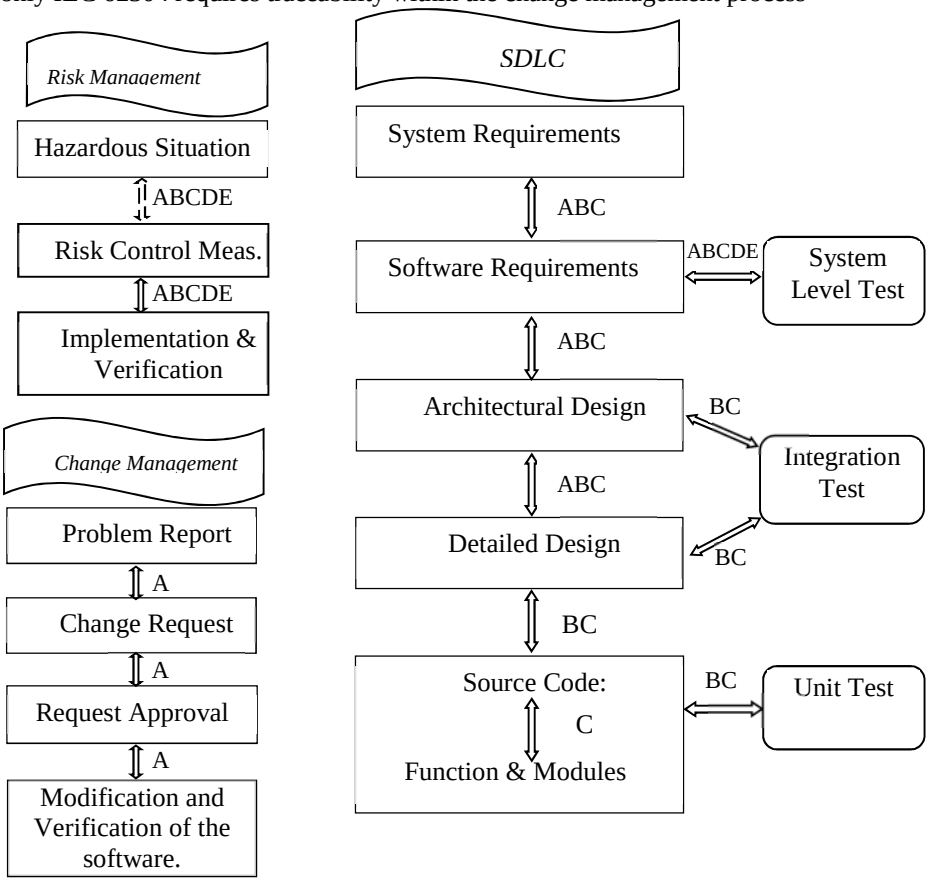

Figure 1: Medical device standards requirements for traceability through the software development lifecycle, Risk management process and Change management process

Table 1: Standards and Guidelines which inform traceability requirements of medical device domain

A	IEC 62304: 2006 Medical Device Software Lifecycle Processes
B	FDA 2005: Guidance for the Content of Pre-market Submission for Software in Medical Devices
C	FDA 2002: General Principles of Software Validation
D	FDA 1999: Off-The-Shelf Software Use in Medical Devices
E	ISO 14971 2007: Application of Risk Management in Medical Devices

4 Methodology

4.1 Aims and method.

The aim of our case study was to answer the following research questions:

1. What is the traceability practice within a medical device SME?
2. Is this practice compliant with medical device standards and guidelines?
3. Is the process used to implement and maintain traceability efficient and compliant?
4. How could this process be improved?

To be able to answer these questions a qualitative study was conducted in which 2 people from the organization were interviewed. These 2 people were chosen as the organization thought they were the most knowledgeable people in the organization with regards to traceability implementation and maintenance. The job titles of the two people involved were Chief Technical Officer and Software Development Manager. Each interview comprised 60 questions and lasted between 2 hours and, 2 hours and 30 minutes; the interviews were recorded and later transcribed and summarized. The questions were developed in such a way as to determine the following:

1. Between what stages of the software development lifecycle did the organisation trace e.g. Do you trace from software requirements to design?
2. How do they implement traceability e.g. How do you trace from software requirement to design?
3. What are the stages of the risk management process and how do they provide traceability between each stage e.g. How do you ensure that each hazard has a corresponding risk control measure, if required?
4. What are the stages of the change management process and how do provide traceability between each stage e.g. IEC 62304 requires the manufacturer to create an audit trail where each change specification, problem report or change request, and each approval of the change request can be traced. Can you explain how you meet this requirement?
5. Difficulties the organisation encountered in implementing and maintaining traceability e.g. What difficulties do you have with implementing or maintaining traceability?
6. Any ideas for improvements the organisation had with regard to implementation and maintenance of traceability e.g. how do you think your present traceability process could be improved?
7. Any process improvement initiatives the organization were currently or were planning to undertake e.g. What process improvements are you working on or plan to work on?

4.2 Organization profile

The organization, who's headquarters are based in the UK, has a research and development and manufacturing facility based in Ireland. This study was carried out in the Ireland facility. The organization employs 60 to 70 people and sometimes employs contractors on a part time basis so the numbers can fluctuate. The products are marketed globally into the primary care market, secondary care, occupational health, sports medicine and clinical trials. Their products are rated as software safety classification II, meaning non-serious injury to the patient or operator of the device is possible due to a defect in the device. The organization uses the V model for their software development. (The V model, like the waterfall model is a sequential path for the execution of processes. Each phase must be completed before the next phase begins. Testing of the product is planned in parallel with a corresponding phase of development. The V model is popular with medical device software development organizations). The organization's main software development process covers class II (non-serious injury is possible e.g. x-ray systems, gas analyzers, pump) & class III (Class III medical devices have the most stringent regulatory controls). They have a separate process for any class I medical device (have the least amount of regulatory control and present minimal potential harm to the user development e.g. tongue depressors, arm slings) they produce.

5 Findings

This section presents the major findings from the interviews held with the organization, beginning with the strategy the organization uses to implement traceability through the SDLC, risk management and change management processes. Additionally the difficulties the organization faces in implementing traceability and any improvements proposed by the organization are unfolded

5.1 Traceability Strategy

The organization have a traceability standard operating procedure which basically is a document detailing how to fill in their traceability matrix (template for matrix is provided which provides consistency). The traceability matrix is 'user requirement' driven i.e. user requirements are listed in the first column then all software requirements that satisfy that user requirement are listed in the second column. Subsequent columns list architectural design, detailed design, risk analysis and test cases). Their quality management procedures say who should complete the matrix, when and how often. In a recent project the CTO updated the user requirement column in the traceability matrix, the development team updated the software requirement, architectural and detailed design columns and QA updated test. The software development manager is ultimately responsible for ensuring the matrix

gets done as and when it should get done and CTO is responsible for approving it at the end.

Bi-directional traceability is implemented through a mixture of in-document tracing and by tracing through the traceability matrix. Bi-directional traceability exists from user requirements to software requirements (through the traceability matrix and through the documents, e.g. the software requirement spec will indicate the corresponding user requirement related to each software requirement). Bi-directional traceability *somewhat* exists between software requirements and architectural and detailed design). The organization "*does not do 'within document' tracing in the design documents. The only way to trace back from design is to search through the traceability matrix for each design component, and, as a design component might satisfy more than one user requirement, this might be a long and laborious task*". The organization has traceability between requirements and test and between design and test, through the traceability matrix. They do not have traceability between code and test.

Traceability from each specification document to the author of that document was evident because each document has the author, reviewer and approver with their signatures and date and each document has an issue control sheet.

Establishing traceability from test to test equipment and its calibration records is documented within the organization's software test plan and not in the risk analysis document.

5.2 Traceability through Risk management

The organization has a Product Risk Management document and a Software Risk Management document. Risk analysis is documented in the risk management document. Risk analysis starts at the User requirements phase and is done at every subsequent phase. Every software requirement is reviewed for potential cause of harm and given a risk classification and may also be given a prioritization based on potential for harm and how easy it is to detect. Based on the risk classification and/or prioritization the organization decides on what mitigations to implement (e.g. change to design, product labeling, training, and user manuals). Software QA validate the mitigations after they have been implemented and then the risks get re-classified hopefully to an acceptable level. *"Each step of risk analysis, risk control implementation and validation is traceable through the risk management document"*.

The software risk analysis has different categories. Usability is an example of a category. "*You might see a number of requirements under the Usability heading but you might also see the same requirement under a number of different headings. So the organization has traceability at the risk analysis level and also the top level requirements traceability matrix*". If there is a change to core product (e.g. introduction of a new feature) the risk analysis document gets updated to a new issue specifying what was added and why.

5.3 Traceability through Change Management

Changes can happen for a number of reasons e.g. a bug, a customer request, the natural progression and maintenance of the software or from a complaint.

If it's a bug they document the issue, the investigation and the results of the investigation, what corrective action if any is required and if the corrective action requires the software to be updated. For any change that requires a change to the software the organization use a Software Changes Specification (SCS). The SCS records any changes to the software, risk associated with the change, impact in terms of other requirements. Traceability is maintained throughout by linking each change to risk and impact. The traceability matrix gets updated during a mini software lifecycle enacted for this change. For any change QA have to do a test plan and test cases, do a test summary to show what tests didn't pass and provide the relevant traceability in a test report. On the other hand a customer may request a change for a clinical trial. The organization use a Customer Requirement Specification (CRS) for this instead of a URS and that drives SCS and a mini lifecycle where the traceability matrix gets updated. Software change requests (from marketing or from a customer for example) are logged in an Excel Spreadsheet recording who logged it, when and why it was logged, and any attachments. For the natural progression and maintenance of software the URS gets updated with new or removed features (along with the traceability matrix) and that goes through the full lifecycle again which means new test plans, new test cases etc. The software requirements, architectural and detailed design specifications also get updated. Complaints are logged in SmartSolve (a document control system) along with investigations, results, and corrective actions. SmartSolve is 21CFR Part 11 compliant so records full electronic signatures. The rationales for customer complaints and for product modifications are recorded (because they have to justify the reason for change). The rationale for requirements in general is not recorded.

5.4 Traceability benefits

The organization recognized two benefits of traceability i.e. "*it is easy to see if all the requirements have been tested*" and "*a matrix is useful when getting audited*". The organization does not use the traceability matrix for instruments such as impact analysis, because; (i) with their experience they can tell what the impact of a requirement change will be from a review of the requirement specification and (ii) the fact that the matrix tends to contain requirement numbers which don't give any detail of what the requirement does so reference to the requirement specification is still required. Putting the required detail into the traceability matrix would make it absolutely massive.

5.5 Downside to traceability and their process

As traceability is manually implemented in an Excel spreadsheet, the organization found this task to be complex and burdensome. This complexity, coupled with the fact

that the organizations felt that the real benefit of having traceability is to satisfy auditors, means that traceability matrix is often not filled out until the project is at or near completion. In fact the general attitude is that the organization "*probably would not concern themselves with traceability but for the fact that they are mandated to do so*".

In a manual system you have to maintain a consistent numbering scheme for the duration of the product's life, so for example if someone releases a new issue of a requirements specification and re-numbers everything then you've lost your traceability; an electronic tool would automatically update everything and traceability wouldn't get lost.

In general, staff was trained in their organization's processes and procedures (including traceability procedures). However there was a lack of understanding of the benefits of traceability and there were no plans in place to address this.

Largely the organization was compliant with the medical device standards and guidelines requirements for traceability (as detailed in Section 3) but there were areas which they were unsure about and would require some guidance e.g. the organization did not trace to code level which is an FDA requirement.

5.6 Traceability tools

Traceability was implemented through a traceability matrix using Excel (along with some in-document tracing). The organization admits that using Excel is not very efficient and painful to update. The organization does not use a dedicated traceability tool for the following reasons:

a) **Time constraints**: It was felt that a lot of time was needed to investigate the suitability of the range of tools on the market and no-one was given the responsibility for doing this

b) **Cost**: It was thought that the cost of purchasing and implementing a traceability tool might not be something that they could justify. One tool (although it was a full application lifecycle management tool) they had considered "*was 100,000 euro and that, even if they had the money, they could not justify spending that amount"*.

c) **Compliance**: issues over the tools not being able to output a full traceability matrix and also failure to meet Title 21 CFR Part 11 requirements which details the FDA's requirements for electronic records and electronic signatures to be trustworthy, reliable and essentially equivalent to paper records and handwritten signatures.

d) **Tool stability**: The organization felt that tools that might be affordable to them would likely not provide the necessary stability and support that they would require e.g. after considering another tool they didn't have confidence in it as there seemed to be only a small group of people involved in the organization and they were concerned about future support and stability issues. Another concern was that the medical device process within the tool is a third part add-on. *"It is a big decision for an organization to move their process to an electronic format, one which we would not be prepared to take unless we had an affordable, stable and fully compliant tool"*.

5.7 Process improvement

At present the organization has a process improvement initiative in place. The organization has hired a person to gather metrics for software QA (metrics to highlight what are the functional areas that are causing issues, and why are they causing issues etc.). It was something they had done in the past but had fallen away. Presently the organization feels that too much of the quality issues are being left to QA and that development needs to take more responsibility; this was highlighted recently when contractors were hired and their code was absolutely solid because they were writing unit test cases for absolutely everything they did. The organization's process states that they will do unit testing on functional areas that QA cannot test via their functional testing, just to make sure they have coverage of everything. So at present every requirement will have a test case or a unit test and code review, or maybe both if it was that critical a requirement.

The organization feel that their traceability process could be improved by use of a traceability tool, "*making it part of everyday work, ensuring traceability gets done and kept up to date*", as at the moment it seems that the traceability matrix only gets filled in towards the end of a project and is mainly used to satisfy auditors. From the organization's perspective if they were not required to do traceability then they might not do it as they find it laborious and they don't get the full benefit of it.

6 Discussion

The organization finds traceability implementation to be a laborious and complex task and feel that the use of a traceability tool to help automate the process would be of great benefit. Using an Excel spreadsheet to create a traceability matrix has several disadvantages such as static and sometimes outdated information, hours wasted in creating the document and no proactive notification capabilities. Effective traceability tools will output up-to-date information, impact analysis reports and a traceability matrix and reduce the complexity involved. It is somewhat surprising then that multiple studies have found the level of commercial traceability tool adoption to be around 50 percent throughout industry. The majority of the remaining companies utilize manual methods and a small percentage develop their own in-house traceability tools [17].

The organization sees the main benefit of traceability as being one of satisfying auditors and only implements traceability because they are mandated to do so. The organization doesn't fully understand the potential benefits of traceability and, although they have a standard operating procedure in place, its implementation is not being driven by management. An organizational policy stating the organization's

policy with regard to traceability along with an education program on the benefits of traceability can help developers and management understand the importance of implementing and maintaining traceability when changes occur.

7 Conclusion

The organization uses two manual approaches to implement traceability i.e. in-document tracing and an excel traceability matrix. A traceability standard operating procedure outlines how to complete the traceability matrix. It is the software engineering manager's responsibility to ensure that the matrix gets completed and it is then signed off by the chief technical officer.

From an auditors perspective the organization would seem to be reasonably compliant, as they implement traceability through the risk management and change management processes and in the most part through the SDLC, although there may be a compliance issue as the organization does not meet the FDA requirement for tracing to the level of code as detailed in Section 3.

However the organization acknowledges that the implementation and maintenance of traceability is a burdensome, inefficient, complex and an inadequately executed task. It is felt that the introduction of a traceability tool would improve efficiency and diminish the burdensome nature of the task. An education program on the benefits of implementing and maintaining traceability would help both management and developers to understand the importance of traceability to producing quality software. The probable result thus being a traceability process adhered to by all relevant personnel, and not just a traceability matrix to be completed at the end of the project for inspection purposes.

Acknowledgment: 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. Ramesh, B., *Factors influencing requirements traceability practice.* Commun. ACM, 1998. **41**(12): p. 37-44.
2. Rakitin, S.R., *Coping with Defective Software in Medical Devices*, in *Computer*. 2006. p. 40-45.
3. Technology, M.D. *Certain J&J insulin pumps destined to fail over software issue*. 2013 [cited 2013 23/01/2013]; Available from:

http://www.massdevice.com/news/diabetes-certain-jj-insulin-pumps-destined-fail-over-software-issue.

4. Congress, S.a.H.o.R.o.t.U.S.o.A.i., *Sarbanes-Oxley Act of 2002''*. P.L. 107–204, Editor. 2002: Washington.
5. Cleland-Huang, J., et al., *Best Practices for Automated Traceability*. Computer, 2007. **40**(6): p. 27-35.
6. ANSI/AAMI/IEC, *62304:2006 Medical device software—Software life cycle processes*. 2006, AAMI: Arlington, VA.
7. FDA, *Guidance for the Content of Premarket Submissions for Software Contained in Medical Devices*. 2005, CDRH: Rockville.
8. FDA, *General Principles of Software Validation; Final Guidance for Industry and FDA Staff*. 2002, CDRH: Rockville.
9. FDA, *Off-The-Shelf Software Use in Medical Devices; Guidance for Industry, FDA Reviewers and Compliance*. 1999, CDRH: Rockville.
10. ISO, *ISO 14971:2007 Medical devices — Application of risk management to medical devices*. 2007, ISO: Switzerland.
11. McCaffery, F., et al., *Medical Device Software Traceability*, in *Software and Systems Traceability*, J. Cleland-Huang, O. Gotel, and A. Zisman, Editors. 2012, Springer.
12. Gotel, O. and P. Mader, *Acquiring Tool Support for Traceability*, in *Software and Systems Traceability*, J. Cleland-Huang, O. Gotel, and A. Zisman, Editors. 2012, Springer.
13. Gotel, O.C.Z. and C.W. Finkelstein. *An analysis of the requirements traceability problem*. in *Requirements Engineering, 1994., Proceedings of the First International Conference on Requirements Engineering*. 1994.
14. Cleland-Huang, J. *Requirements Traceability - When and How does it Deliver more than it Costs?* in *Requirements Engineering, 14th IEEE International Conference*. 2006.
15. Espinoza, A. and 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*. 2008. Madrid.
16. Lucia, A.D., et al., *Information Retrieval Methods for Automated Traceability Recovery*, in *Software and Systems Traceability*, J. Cleland-Huang, O. Gotel, and A. Zisman, Editors. 2012, Springer. p. 88 - 111.
17. Kannenberg, A. and D.H. Saiedian, *Why Software Requirements Traceability Remains a Challenge.* CrossTalk The Journal of Defense Software Engineering, July/Aug 2009.