The Barriers to Traceability and their Potential Solutions: Towards a Reference Framework

Gilbert Regan, Fergal McCaffery, KevinMc Daid and Derek Flood

Regulated Software Research Group,
Department of Computing & Mathematics,
Dundalk Institute of Technology & Lero,
Dundalk, Ireland
{gilbert.regan, fergal.mccaffery, kevin.mcdaid, derek.flood} @dkit.ie

***Abstract*—Traceability of software artifacts, from requirements to design and through implementation and quality assurance, has long been promoted by the research and expert practitioner communities. However, evidence indicates that, with the exception of those operating in the safety critical domain, few software companies choose to implement traceability processes, often due to associated cost and complexity issues. This paper presents a review of traceability literature including the implementation of traceability in real organizations. Through both analyzing case studies and research published by leading traceability researchers, this paper synthesizes the barriers faced by organizations while implementing traceability, along with proposed solutions to the barriers. Additionally, given the importance of traceability in the regulated domain of safety critical software, the paper compares the barriers for organizations operating inside and outside of this domain.**

Keywords-traceability, barriers, safety-critical

I. INTRODUCTION

Software systems are becoming increasingly complex. Artefacts such as test cases, requirements documents, source code, design documents, bug reports 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 artefacts [1]. Though the importance and role of traceability in supporting systems development has been long recognised, there are wide variations in the quality and usefulness of the practice of traceability [2]. 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 relevant [3].

Good traceability information is fundamental for a number of reasons including change impact analysis, requirements validation and regression testing. Often the quality of this information is poor, or out of date due to improper maintenance [4].

Traceability techniques and tools are not widely used in industry [5, 6]. Companies who do adopt traceability techniques often adopt inefficient manual traceability methods and tools despite semi-automated and automated approaches becoming available [6].

This paper considers the barriers to implementing traceability and proposes solutions to those barriers. This paper will also consider if there is any difference in barriers between the general and safety critical domains. To achieve this, a literature review including eight case studies was conducted for both generic and safety critical software domains.

This paper is structured as follows: Section II presents the research methodology used in this work. Section IIII presents the findings of this work. Section IV proposes a framework for barriers/solutions. Our conclusion and future work is provided in Section V .

II. RESEARCH METHODOLOGY

The purpose of this research is to answer the following research question:

> What are the barriers to an organisation implementing traceability and how can these barriers be overcome?

In addition to examining the barriers and potential solutions, we also decided to investigate any difference in barriers between the general and safety critical domains.

Traceability literature was surveyed to gain insight from both research experts and industry practitioners regarding the potential barriers to implementing traceability and possible solutions to these barriers.

The following steps were used during the performance of the literature review:

- Collect all publications from identified sources using keywords as detailed below
- Identify only full or short papers; excluding workshops, posters etc.
- Examine the abstracts of each paper to determine if they meet the inclusion criteria
- Extract all pertinent information from each publication.

The inclusion criteria were: Software traceability implementation, Problems/barriers to implementing traceability, Case studies focusing on software traceability implementation and dated within the last 8 years

The portals that facilitated this research were IEEE Xplore digital library, ACM digital library and Google Scholar. Key search words used were: *traceability, software+traceability, requirements+traceability, traceability + case + study, traceability+survey, traceability+barriers*. These searches returned more than 150 publications. Each abstract was scanned for relevance to the research topic (detail on barriers to traceability) and 32 were selected to inform the research. 12 of these were case studies from which we selected 8 of the most recent and relevant (the other 4 did not have much detail).

In addition to the above , a recently published book entitled Software and Systems Traceability [1] informed a great deal of our research. Some of the chapters from this book are referred to throughout this paper.

The case studies report on traceability practices in and experiences of implementing and using traceability. One study carried out an assessment of two organisations with regard to their practice of traceability. The case studies are referenced as follows:**Klimpke: 2009** [11] , **Panis: 2010** [12] , **Arkley and Riddle: 2006** [13] , **Neumuller and Grunbacher: 2006** [6], **Mc Caffery: 2011** [14] , **Heindl and Biffl: 2005** [15 , **Born, Favaro and Kath: 2010** [16] , **Mader, Gotel and Philippow: 2009** [17]

III. FINDINGS

The barriers and their solutions are categorised as either Management issues, Social issues or Technical issues. This section synthesises industry viewpoints on barriers to traceability (taken from the case studies) with those published by established researchers. This section presents a reference framework of the barriers to traceability and the proposed solutions to these barriers.

5.1 Management Issues

The Management issues are considered to be Cost, Return on Investment, Traceability decay, Lack of guidance and Data collection issues.

Cost. There are three main costs associated with the implementation of traceability. These are the costs of purchasing or developing traceability tools, overhead in terms of training and labour[18]. Three suggestions for mitigating these costs is to **a)** design the user interface of these tools with usability in mind to help minimise the amount of user training required [19], **b)** consider the use of value based requirements tracing to reduce the amount of time spent in performing traceability tasks [15], and **c)** consider the use of a general purpose tool such as Excel or Word or to use custom built tools [7].

Lack of Guidance. Almost no guidance is available for practitioners to help them establish traceability in their projects and as a result, practitioners are ill-informed as to how best to accomplish this task [14, 17].

An industry or organisational guideline as to which artefacts to trace to and at what level of granularity etc. is required. Practicing value based requirements traceability is a possible solution as to which requirements to trace from.

Return on Investment (ROI). There are few metrics for measuring the return on investment [21] for traceability and there is little comparative data available on the cost effectiveness of various traceability techniques, methods and tools. Many of the benefits of traceability may only be realised during maintenance or after delivery of the product [6].Education and training will promote the value perception of traceability [5] and help mitigate ROI issues. Management should be educated on benefits of traceability such as financial gains which are as a result of productivity and quality.

Traceability Decay. Traceability decays over time and can only be prevented by traceability maintenance [12]. Currently, automatic traceability is not fully trusted and therefore manual feedback from users is still required which is expensive, hence it is not feasible to recover trace links anew every time a model changes [19]. Making someone responsible for trace maintenance [21] and providing them with the required training, along with better design of tools to return more precise traces should help ensure traces are maintained.

Data Collection. Collecting all possible data is likely to result in an unmanageably large dataset and increased project cost. A cost trade-off exists between collecting data and ensuring appropriate data is available [9].

'Trace for a purpose' [22] (ensures that no link gets established that does not serve a clear purpose), practise value based requirements tracing or ensuring that the best available techniques are selected for each individual link (this means that a single requirement could be traced to one artefact using one technique and to another artefact using a different technique [10]) are possible solutions.

5.2 Social Issues

Different stakeholder viewpoints. Some sectors within an organisation can perceive traceability as an optional extra (and of low priority), so the allocation of time, staff, and resources is often insufficient. Current best practice [21] is to consider the views of different stakeholders by creating an organizational policy for traceability that may be applied uniformly to all projects within the organization.

Internal politics. The need for documentation can cause resentment among developers who may fear that traces could be used to monitor their work [23]. It is also important to integrate traceability tasks with the developers existing work to ensure that such tasks are performed [6, 24]. It is

suggested to 'Make only small changes to work practices' as traceability has to support developers in their daily tasks [6].

Lack of communication between groups [24] and/or unclear understanding of artefact ownership [18]. To reduce the effects of these issues it is proposed that projects should have: clear visibility of responsibilities and knowledge areas; clarity of working structures; and team commitment and ownership [8]. Regular reviews and meetings between groups should help to alleviate communication difficulties.

5.3 Technical Issues

The technical issues are considered to be associated with tools, storage and versioning, complexity, and other technical issues

Issues with tools. There are numerous issues with tools which hinder the establishment and maintenance of traceability. Choosing a tool for traceability is a difficult and time consuming task [3, 6, 11, 14]. Challenges include:

- the lack of standalone traceability tools;
- selecting between available tools, configuring a general purpose tool or developing a custom tool;
- tools not being maintained;
- traces may not integrate across tool and organisational boundaries;
- automatic tools require validation which is time consuming and costly;
- automated tools lack accountability which possibly disqualifies it from safety critical domain;
- Tools have integration issues with other tools.

Further development of standalone, better designed traceability tools is one option for mitigating the issues [16].

Storage and Versioning.. An integrated artefact repository that holds at least the major models of the development process would make the storage and versioning of traceability relations less problematic. This requires an agreed format among tool vendors which is something that is unlikely to be popular [17].

Complexity. Technical issues such as the sheer number of artefacts to be traced and the complex relationship between artefacts hamper traceability [18, 25, 26]. Non-functional requirements tracing is particularly complex. Techniques such as 'Practising value-based requirements tracing' [15] or 'Bound the problem space' (limit the level of tracing: too much tracing can cause more harm than good) help with alleviating complexity, in addition to automating 'the right set of features' (being selective) [6]. However tracing non-functional requirements remains difficult.

IV. FRAMEWORK

Table 1: Categories of barriers to traceability and their proposed solutions

Category	Barriers	Proposed Solutions
Management Issues	Cost	Tool design - usability. VBRT. Use general purpose tool Build tool from scratch.
	Lack of Guidance	Industry/domain specific guidelines
	Return on Investment	Management education
	Traceability decay	Taking ownership. Training/Education. Better tool design for precise trace return.
	Data collection	Trace for a purpose. Value based requirements tracing. Best available technique per link.
Social Issues	Different stakeholder viewpoints	Organisational policy
	Internal politics	Organisational policy, Incentive schemes. Integrate traceability tasks into existing work practices.
	Lack of communication/ understanding	Group meetings/reviews /training. Clear visibility of responsibilities.
Technical Issues	Issues with tools	Make person/dept responsible. Develop custom tools. Semi-automated approach.
	Storage and versioning	An integrated artefact repository.
	Complexity	Use of various techniques and tools. VBRT. Bound the problem space.

V. CONCLUSIONS

According to the literature, traceability is beneficial. However its implementation is inconsistent at best, with most companies either not implementing it or implementing it in a haphazard manner. Our findings show that there are many barriers to a company implementing and using traceability (e.g. cost, complexity, lack of guidance, political issues, and tool issues). However, these barriers can be overcome to some extent with solutions such as: value based requirements tracing; industry/organisational policies; customised tools; using different approaches and techniques including a semi-automated approach; training/education; incentive schemes; and assigning responsibilities. Some of

the challenges surrounding tools and tracing of non functional requirements remain difficult to overcome.

All of the above barriers are important to both the general and safety critical domains. However, for the safety critical domain, there are further complexities with using automated tracing as fully automated tracing alone does not ensure accountability which possibly disqualifies it from the safety critical domain. Using manual checking of automated traces however could overcome this issue.

Future work will be performed to gain industry based feedback in relation to the framework, to determine how effective the suggestions for overcoming the barriers to implementing traceability are in practice.

ACKNOWLEDGEMENTS

This work was supported, in part, by Science Foundation Ireland grant 10/CE/I1855 to Lero - the Irish Software Engineering Research Centre

REFERENCES

1. Gotel, O., J. Cleland-Huang, and A. Zisman, *Software and Systems Traceability*, ed. J. Cleland-Huang, O. Gotel, and A. Zisman. 2012, London Dordrecht Heidelberg New York: Springer.
2. Ramesh, B., *Factors influencing requirements traceability practice*. Commun. ACM, 1998. **41**(12): p. 37-44.
3. McCaffery, F., et al., *Medical Device Software Traceability*, in *Software and Systems Traceability*, O.G.e. al, Editor. 2012, Springer.
4. A.Dekhtar and J.H. Hayes, *Studying the role of Humans in the Traceability Loop*, in *Software and Systems Traceability*. 2012, Springer.
5. Gotel, O., et al., *The Grand Challenge of Traceability*, in *Software and Systems Traceability*. 2012, Springer.
6. Neumuller, C. and P. Grunbacher. *Automating Software Traceability in Very Small Companies: A Case Study and Lessons Learned*. in *Automated Software Engineering, 2006. ASE '06. 21st IEEE/ACM International Conference on*. 2006.
7. Gotel, O. and P. Mader, *Acquiring Tool Support for Traceability*, in *Software and Systems Traceability*. 2012, Springer.
8. 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. P. 94-101.
9. Ingram, C. and S. Riddle, *Cost Benefits of Traceability*, in *Software and Systems Traceability*. 2012, Springer.
10. Cleland-Huang, J., G. Zemont, and W. Lukasik. *A heterogeneous solution for improving the return on investment of requirements traceability*. in *Requirements Engineering Conference, 2004. Proceedings. 12th IEEE International*. 2004.
11. Klimpke, L. and T. Hildenbrand, *Towards End-to-End Traceability: Insights and Implications from Five Case Studies*, in *Proceedings of the 2009 Fourth International Conference on Software Engineering Advances*. 2009, IEEE Computer Society. p. 465-470.
12. Panis, M.C. *Successful Deployment of Requirements Traceability in a Commercial Engineering Organization...Really*. in *Requirements Engineering Conference (RE), 2010 18th IEEE International*. 2010.
13. Arkley, P. and S. Riddle. *Tailoring Traceability Information to Business Needs*. in *Requirements Engineering, 14th IEEE International Conference*. 2006.
14 Caffery, F.M. and V. Casey, *Med-Trace* in *SPICE 2011*. 2011.
15. Heindl, M. and S. Biffl, *A case study on value-based requirements tracing*, in *Proceedings of the 10th European software engineering conference held jointly with 13th ACM SIGSOFT international symposium on Foundations of software engineering*. 2005, ACM: Lisbon, Portugal. p. 60-69.
16. Born, M., J. Favaro, and O. Kath, *Application of ISO DIS 26262 in Practice*, in *CARS*. 2010: Valencia,Spain.
17. Mader, P., O. Gotel, and I. Philippow, *Motivation Matters in the Traceability Trenches*, in *Proceedings of the 2009 17th IEEE International Requirements Engineering Conference, RE*. 2009, IEEE Computer Society. p. 143-148.
18. Asuncion, H.U., et al., *An end-to-end industrial software traceability tool*, in *Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering*. 2007, ACM: Dubrovnik, Croatia. p. 115-124.
19. Mader, P. and O. Gotel, *Ready-to-Use Traceability on Evolving Projects*, in *Software and Systems Traceability*, J. Cleland-Huang, OrlenaGotel, and A. Zisman, Editors. 2012, Springer: London-Dordrecht-Heidelberg-New York.
20. Mirakhorli, M. and J. Cleland-Huang, *Tracing architectural concerns in high assurance systems (NIER track)*, in *Proceedings of the 33rd International Conference on Software Engineering*. 2011, ACM: Waikiki, Honolulu, HI, USA. p. 908-911.
21. Kannenberg, A. and D.H. Saiedian, *Why Software Requirements Traceability Remains a Challenge.* CrossTalk The Journal of Defense Software Engineering, July/Aug 2009.
22. Cleland-Huang, J., et al., *Best Practices for Automated Traceability.* Computer, 2007. **40**(6): p. 27-35.
23. Jarke, M., *Requirements tracing.* Commun. ACM, 1998. **41**(12): p. 32-36.
24. Asuncion, H.U. and R.N. Taylor, *An end-to-end industrial software traceability tool*, in *Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering*. 2007, ACM: Dubrovnik, Croatia. p. 115-124.
25. Arkley, P. and S. Riddle, *Overcoming the Traceability Benefit Problem*, in *Proceedings of the 13th IEEE International Conference on Requirements Engineering.* 2005, IEEE Computer Society. p. 385-389.
26.. Anderson, K., S. Sherba, and W. Lepthien, *Towards large-scale information integration*, in *ICSE 02*. 2002: Orlando, Florida.