Traceability Research: Taking the Next Steps

Jane Cleland-Huang
Systems and Requirements Engineering Center
School of Computing
DePaul University
jhuang@cs.depaul.edu

ABSTRACT

This keynote highlights areas of significant accomplishments in traceability research and asks the question of "where next?" It describes forward looking projects of the Center of Excellence for Software Traceability (CoEST) and raises some of the difficult questions related to building a shared research infrastructure in the traceability community.

Categories and Subject Descriptors

D.2.7 [**Software Engineering**]: Distribution, Maintenance, and Enhancement; D.2.1 [**Requirements/Specifications**]: Tools

General Terms

Experimentation, Measurement, Documentation

Keywords

Traceability, Benchmarks, Metrics, Tools

1. THE TRACEABILITY CHALLENGE

Seminal work in 1995 by Gotel and Finkelstein [11] brought traceability issues to the foreground of requirements engineering and spawned a new generation of traceability solutions. However, despite major advances over the past fifteen years in both the state of practice and the state of art, the challenges are still daunting. This was illustrated in a recent report commissioned by the US Committee for Advancing Software-Intensive Systems Producibility for the defense industry [7], which highlighted the critical lack of "chains of evidence" in the form of traceability links between executable code and the functional and non-functional requirements that drive the code. Such chains of evidence would support evolution of requirements, architectural models, designs, code, and test cases, and facilitate higher level activities such as software assurance, requirements validation, and compliance verification. Despite significant advances in our field, it is evident that we have not yet succeeded in "solving" the traceability problem. For every success story [19] there are hundreds of cases in which complex and even safety-critical projects fail to use traceability effectively to support the goals and requirements of their projects [20, 7]. Furthermore, traceability is still generally perceived as an important, yet costly, arduous, and error-prone activity.

2. MAJOR ADVANCES

On the positive side, a quick survey of traceability research over the past fifteen years, reveals a research community which is grappling with some of the trickier challenges, and making significant yet incremental headway in solving them. To provide a few illustrative examples, numerous researchers have tackled the problem of automated trace retrieval, and have developed techniques for reducing manual effort by using information retrieval methods to generate candidate traceability links on-demand [18, 2, 17, 13, 15, 5]. Other researchers have looked at the role of the human in the traceability process. They have addressed questions such as "how should the human analyst filter candidate traceability links to produce a correct traceability matrix?", or "how does traceability support a software engineer in the specific engineering tasks he or she must perform?" [13, 16, 8, 19]. Others have investigated more specific problems such as the application of traceability in product line development or in specific industries such as the health-care industry. Still others have explored new ways to create or evolve traceability links in an ongoing and constantly changing project environment [14, 3]. Although it is impossible to cite all the work in this area, it is evident that creativity and innovation are endless, and it is clear that our community has made significant headway in addressing specific components of the traceability problem.

3. WHERE NEXT?

Nevertheless it is also fair to say that difficult problems don't disappear overnight. Such problems take long-term effort to solve, and sometimes span the careers of multiple generations of researchers and practitioners, each one building incrementally on the work of those who came before. With such long-term research efforts, it is especially important to keep our eye on the end goal [10].

Recognizing the challenges faced by our community, members of the Center of Excellence for Software Traceability [1] have worked collaboratively on several forward looking projects. The first project was launched in 2005 with funding from NASA and NSF and involves identifying and specifying the Grand Challenges of Traceability [6]. Once completed, this project will provide a roadmap for traceability research and will include challenges, goals, and specific tasks for advancing traceability research and practice. The second project is the Tracy project, which seeks to build shared infrastructure for our community in the form of shared datasets, metrics, and research tasks that are formulated into specific benchmarks [4, 12]. The benchmark project is intended to help researchers compare results across research groups while encouraging innovation and creativity. Finally the TraceLab project [9], which is also funded by NSF under a major research instrumentation grant, seeks to build an experimental environment, which will encourage new researchers to enter the field, and will promote collaboration between research groups.

4. PIPE DREAM OR REALITY?

Ongoing efforts to build shared infrastructure change the landscape of the research environment. On one hand they provide tools and datasets which are intended to increase the productivity of researchers, but on the other hand they introduce benchmarks which implicitly require higher levels of accountability. In recent discussions with researchers from other Software Engineering disciplines, we have heard concerns that developing benchmarks might have the undesired effect of creating a research silo that is even harder for new researchers to enter, and furthermore, that expecting researchers to selflessly share tools and datasets is an unrealistic expectation.

So far this has not been our experience. Nevertheless, these are real concerns which need to be addressed. We need to ensure that research infrastructure encourages and facilitates engagement in traceability research. We must address head-on the fairness issues of asking researchers to share and exchange datasets and tools, and identify ways to recognize and reward researchers for their contributions. Finally, we need to keep our end goals in mind, and develop new ways to assess, evaluate, and track our progress. Although it is clear that the traceability challenges will not be solved overnight, ongoing collaborations, continuing innovation, incremental improvements, and occasional moments of brilliance will keep us moving in the right direction and will ultimately lead to holistic and effective traceability solutions.

5. ACKNOWLEDGMENTS

The Tracy and TraceLab projects described in this paper were primarily funded by the U.S. National Science Foundation grant #CNS-0959924.

6. REFERENCES

[1] Center of Excellence for Software Traceability, http://www.traceabilitycenter.org, March 2008.

[2] G. Antoniol, G. Canfora, G. Casazza, A. D. Lucia, and E. Merlo. Recovering traceability links between code and documentation. *IEEE Trans. Software Eng.*, 28(10):970–983, 2002.

[3] H. U. Asuncion, A. Asuncion, and R. N. Taylor. Software traceability with topic modeling. In *ICSE (1)*, pages 95–104, 2010.

[4] J. Cleland-Huang, A. Czauderna, A. Dekhtyar, O. Gotel, J. Huffman Hayes, E. Keenan, G. Leach, J. Maletic, D. Poshyvanyk, Y. Shin, A. Zisman, G. Antonio, B. Berenbach, A. Egyed, and P. Maeder. Grand challenges, benchmarks, and tracelab: Developing infrastructure for the software traceability research community. 2011.

[5] J. Cleland-Huang, A. Czauderna, M. Gibiec, and J. Emenecker. A machine learning approach for tracing regulatory codes to product specific requirements. In *ICSE (1)*, pages 155–164, 2010.

[6] J. Cleland-Huang, A. Dekhtyar, J. Huffman Hayes, G. Antoniol, B. Berenbach, A. Egyed, S. Ferguson, J. Maletic, and A. Zisman. *Grand Challenges of Traceability.* Center of Excellence for Software Traceability, http://www.coest.org, 2006.

[7] Committee for Advancing Software-Intensive Systems Producibility. *Critical Code: Software Producibility for Defense.* National Research Council, USA, 2011.

[8] D. Cuddeback, A. Dekhtyar, and J. Hayes. Automated requirements traceability: The study of human analysts. *Requirements Engineering, IEEE Intn'l Conference on*, 0:231–240, 2010.

[9] A. Czauderna, M. Gibiec, G. Leach, Y. Li, Y. Shin, E. Keenan, and J. Cleland-Huang. Gtraceability challenge 2011: Using tracelab to evaluate the impact of local versus global idf on trace retrieval. 2011.

[10] A. Davis. Requirements: A textbook example of goals displacement. *Requirements Engineering, IEEE Intn'l Conference on*, 0:xx, 2010.

[11] O. Gotel and A. Finkelstein. Contribution structures (requirements artifacts). In *RE*, pages 100–107, 1995.

[12] O. Gotel et al. The Grand Challenges of Traceability 2.0. *Software and Systems Traceability*, Springer Verlag, November, 2011.

[13] J. Huffman Hayes, A. Dekhtyar, S. K. Sundaram, and S. Howard. Helping analysts trace requirements: An objective look. In *RE*, pages 249–259, 2004.

[14] W. Jirapanthong and A. Zisman. Xtraque: traceability for product line systems. *Software and System Modeling*, 8(1):117–144, 2009.

[15] H. H. Kagdi, J. I. Maletic, and B. Sharif. Mining software repositories for traceability links. In *ICPC*, pages 145–154, 2007.

[16] P. Mäder, O. Gotel, and I. Philippow. Motivation matters in the traceability trenches. In *RE*, pages 143–148, 2009.

[17] A. Marcus, J. I. Maletic, and A. Sergeyev. Recovery of traceability links between software documentation and source code. *Intn'l Journal of Software Engineering and Knowledge Engineering*, 15(5):811–836, 2005.

[18] R. Oliveto, M. Gethers, D. Poshyvanyk, and A. De Lucia. On the equivalence of information retrieval methods for automated traceability link recovery. In *Proc. of 18th IEEE Intn'l Conference on Program Comprehension (ICPC'10)*, pages 68–71, 2010.

[19] M. C. Panis. Successful deployment of requirements traceability in a commercial engineering organization...really. *Requirements Engineering, IEEE Intn'l Conference on*, 0:303–307, 2010.

[20] B. Ramesh and M. Jarke. Toward reference models of requirements traceability. *IEEE Trans. Software Eng.*, 27(1):58–93, 2001.