Pathways to Technology Transfer and Adoption: Achievements and Challenges (Mini-Tutorial)

Dongmei Zhang
Microsoft Research Asia
Beijing, 100080, China
Email: dongmeiz@microsoft.com

Tao Xie
North Carolina State University
Raleigh, NC 27695, USA
Email: xie@csc.ncsu.edu

***Abstract*—Producing industrial impact has often been one of the important goals of academic or industrial researchers when conducting research. However, it is generally challenging to transfer research results into industrial practices. There are some common challenges faced when pursuing technology transfer and adoption while particular challenges for some particular research areas. At the same time, various opportunities also exist for technology transfer and adoption.**

This mini-tutorial presents achievements and challenges of technology transfer and adoption in various areas in software engineering, with examples drawn from research areas such as software analytics along with software testing and analysis. This mini-tutorial highlights success stories in industry, research achievements that are transferred to industrial practice, and challenges and lessons learned in technology transfer and adoption.

I. Introduction

Producing industrial impact (such as producing successful technology transfer and adoption) has often been one of the important goals of academic or industrial researchers when conducting research. However, it is generally challenging to transfer research results into industrial practices. Some research areas within software engineering may have more successful stories of technology transfer and adoption than some other areas. There are some common challenges faced when pursuing technology transfer and adoption while particular challenges for some particular research areas. At the same time, various opportunities also exist for technology transfer and adoption.

Built upon a mini-tutorial on *Software Analytics in Practice* [26] that we presented in the ICSE 2012 Software Engineering in Practice (SEIP) track, this mini-tutorial presents achievements and challenges of technology transfer and adoption in various areas in software engineering, with examples drawn from research areas such as software analytics [25] along with software testing and analysis [20]. In particular, the objective of this mini-tutorial is to allow the attendees to gain an overview of successful technology transfer and adoption, learn about challenges and opportunities in technology transfer and adoption, and acquire knowledge needed to carry out technology transfer and adoption. The mini-tutorial covers the topic in the domain of software engineering and includes examples from software engineering, such as code-clone detection for reliability, security, and maintenance [16], [8], [9], mining runtime callstacks [15] or logs [11], [12] for performance diagnosis, static defect detection [16], [4], [2], [27], [3], and test generation [21], [22], [5], [18], [23], [10], [1], [14], [19].

II. Target Audience

The mini-tutorial is targeted at academic researchers, industrial researchers, and software practitioners who have interest in technology transfer and adoption.

Academic researchers. The mini-tutorial gives an overview of collaborations between academic researchers (from universities) and industrial researchers (from industrial research labs) or software practitioners (from product teams at companies). Academic researchers are expected to acquire knowledge needed to carry out collaborations with industrial researchers or software practitioners to strive for successful technology transfer and adoption.

Industrial researchers. The mini-tutorial gives an overview of collaborations between industrial researchers (from industrial research labs) and academic researchers (from universities) or software practitioners (from product teams at companies). Industrial researchers are expected to acquire knowledge needed to carry out collaborations with academic researchers or software practitioners to strive for successful technology transfer and adoption.

Software practitioners. The mini-tutorial gives an overview of collaborations between software practitioners (from product teams at companies) and industrial researchers (from industrial research labs) or academic researchers (from universities). Software practitioners are expected to acquire knowledge needed to carry out collaborations with industrial researchers or academic researchers to strive for successful technology transfer and adoption.

The reasons why the topic is timely and relevant are primarily three folds. First, a set of promising research results have been produced by the research community and demonstrated to be useful on various real-world open source projects. There are huge opportunities for exploiting these research results to improve industrial practices of software engineering. Second, there are substantial demands from software practitioners to address their urgent and critical issues in software engineering practices. Third, the research community has already realized gaps between academic research and industrial practices [17], and has called for training and education of researchers and practitioners in conducting successful technology transfer and adoption. For example, gaps between academic research and industrial practices were discussed in both Carlo Ghezzi's ICSE 2009 keynote speech [13] along with Lionel Briand's ICSM 2011 keynote speech [6] and his recent article [7]. Furthermore, successful experiences on technology transfer were shared by Yuanyuan Zhou in her MSR 2011 keynote speech [27] and Dongmei Zhang in her MSR 2012 keynote speech [24].

Acknowledgment

Tao Xie's work is supported in part by NSF grants CCF-0845272, CCF-0915400, CNS-0958235, CNS-1160603, an NSA Science of Security Lablet grant, a NIST grant, a Microsoft Research Software Engineering Innovation Foundation Award, and National Science Foundation of China No. 61228203.

References

[1] Microsoft's protocol documentation program: Interoperability testing at scale. *Queue*, 9(6):20:20–20:27, June 2011.

[2] N. Ayewah and W. Pugh. The Google FindBugs fixit. In *Proc. ISSTA*, pages 241–252, 2010.

[3] T. Ball, V. Levin, and S. K. Rajamani. A decade of software model checking with SLAM. *Commun. ACM*, 54(7):68–76, July 2011.

[4] A. Bessey, K. Block, B. Chelf, A. Chou, B. Fulton, S. Hallem, C. Henri-Gros, A. Kamsky, S. McPeak, and D. R. Engler. A few billion lines of code later: using static analysis to find bugs in the real world. *Commun. ACM*, 53(2):66–75, 2010.

[5] M. Boshernitsan, R. Doong, and A. Savoia. From Daikon to Agitator: lessons and challenges in building a commercial tool for developer testing. In *Proc. ISSTA*, pages 169–180, 2006.

[6] L. C. Briand. Useful software engineering research - leading a double-agent life. In *Proc. ICSM, Keynote Speech*, page 2, 2011.

[7] L. C. Briand. Embracing the engineering side of software engineering. *IEEE Software*, 29(4):96, 2012.

[8] Y. Dang, S. Ge, R. Huang, and D. Zhang. Code clone detection experience at Microsoft. In *Proc. IWSC*, pages 63–64, 2011.

[9] Y. Dang, D. Zhang, S. Ge, C. Chu, Y. Qiu, and T. Xie. XIAO: Tuning code clones at hands of engineers in practice. In *Proc. ACSAC*, pages 369–378, 2012.

[10] J. de Halleux and N. Tillmann. Moles: tool-assisted environment isolation with closures. In *Proc. TOOLS*, pages 253–270, 2010.

[11] R. Ding, Q. Fu, J.-G. Lou, Q. Lin, D. Zhang, J. Shen, and T. Xie. Healing online service systems via mining historical issue repositories. In *Proc. ASE*, pages 318–321, 2012.

[12] Q. Fu, J.-G. Lou, Q.-W. Lin, R. Ding, Z. Ye, D. Zhang, and T. Xie. Performance issue diagnosis for online service systems. In *Proc. SRDS*, pages 273–278, 2012.

[13] C. Ghezzi. Reflections on 40+ years of software engineering research and beyond: an insider's view. In *Keynote address at ICSE*, 2009.

[14] P. Godefroid, M. Y. Levin, and D. Molnar. SAGE: Whitebox fuzzing for security testing. *Queue*, 10(1):20:20–20:27, Jan. 2012.

[15] S. Han, Y. Dang, S. Ge, D. Zhang, and T. Xie. Performance debugging in the large via mining millions of stack traces. In *Proc. ICSE*, pages 145–155, 2012.

[16] Z. Li, S. Lu, S. Myagmar, and Y. Zhou. CP-Miner: Finding copy-paste and related bugs in large-scale software code. *IEEE Trans. Software Eng.*, 32(3):176–192, 2006.

[17] L. J. Osterweil, C. Ghezzi, J. Kramer, and A. L. Wolf. Determining the impact of software engineering research on practice. *IEEE Computer*, 41(3):39–49, 2008.

[18] N. Tillmann and J. de Halleux. Pex – white box test generation for .NET. In *Proc. TAP*, pages 134–153, 2008.

[19] N. Tillmann, J. D. Halleux, T. Xie, S. Gulwani, and J. Bishop. Teaching and learning programming and software engineering via interactive gaming. In *Proc. ICSE, Software Engineering Education (SEE)*, 2013.

[20] X. Xiao, S. Thummalapenta, and T. Xie. Advances on improving automation in developer testing. *Advances in Computers*, 85:165–212, 2012.

[21] T. Xie, D. Marinov, and D. Notkin. Rostra: A framework for detecting redundant object-oriented unit tests. In *Proc. ASE*, pages 196–205, 2004.

[22] T. Xie and D. Notkin. Tool-assisted unit-test generation and selection based on operational abstractions. *Automated Software Engineering Journal*, 13(3):345–371, July 2006.

[23] T. Xie, N. Tillmann, P. de Halleux, and W. Schulte. Fitness-guided path exploration in dynamic symbolic execution. In *Proc. DSN*, pages 359–368, 2009.

[24] D. Zhang. MSR 2012 keynote: Software analytics in practice - approaches and experiences. In *Proc. MSR*, page 1, 2012.

[25] D. Zhang, Y. Dang, J.-G. Lou, S. Han, H. Zhang, and T. Xie. Software analytics as a learning case in practice: Approaches and experiences. In *Proc. MALETS*, 2011.

[26] D. Zhang and T. Xie. Software analytics in practice: Mini tutorial. In *Proc. ICSE, SEIP, Mini Tutorial*, page 997, 2012.

[27] Y. Zhou. Connecting technology with real-world problems - from copy-paste detection to detecting known bugs (keynote abstract). In *Proc. MSR*, pages 2–2, 2011.