MUD 2014 Keynote

Lessons and Insights from Tech Transfers at Microsoft

Christian Bird, Microsoft Research, USA

Abstract

"Research without practice is like building castles in the air. Practice without research is building castles on slippery grounds." Kader Parahoo, 1997.

As a basic industrial research lab, Microsoft Research expects its members to both conduct and publish basic research. However, another expectation is to transfer our research techniques, practices, tools, and ideas into products and practice elsewhere within the company. The goal of turning research into practice is not limited to MSR. Companies such as Siemens, IBM, and ABB invest heavily in basic research in an effort to improve their software development teams. At the same time, researchers in other domains, including university professors and government lab scientists, endeavor to have their research impact practice.

Unfortunately, moving from a validated technique or model in a published paper to a state where that same technique is being used by and providing value to software development projects on a regular basis is a time consuming, fraught, and difficult task. We have attempted to make this transition, which we call "Tech Transfer", many times in the empirical software engineering group (ESE) at Microsoft Research. Much like research in general, while there have been successes and failures, each experience has provided valuable insight and informed our next effort.

Development teams need data that is continually fresh for them to make informed decisions. Developers won't trust models that they can't understand or that is based on assumptions that they disagree with. Recommendation systems can't take minutes to make recommendations and must adapt quickly to changes in team processes, company reorganizations, and new tools. Just because a new website exists with valuable risk analytics doesn't mean that testers will begin visiting it regularly. The developer user experience is just as critical as the performance of the algorithm behind it. Techniques that work great for hundred thousand line codebases break down when facing tens of millions of lines. Two teams may exist in the same company and use different version control systems, build infrastructure, test processes, and reporting tools, making a "one size fits all" approach all but impossible. These are just a few of the challenges to tech transfer we have encountered over the past four years of ESE's existence.

Fortunately, we have found ways to overcome or mitigate many of these obstacles. In this talk, I will share share the lessons that we have learned and the insights we have gained in our efforts to turn our research into tools and processes that are being used on a daily basis within Microsoft. While the concrete implementations were specific to our company, the ideas and solutions can be used in other organizations and domains of software

engineering by anyone hoping to put their research into the hands of development teams. We believe they can be of value to any researcher that wants to see their work used by others.

I will share how a risk assessment system went from a web- site with a few visits per day to a web-service service thousands of requests per day. I will discuss the tradeoffs we faced and the design decisions we made to a system that automatically adds high expertise reviewers to code reviews (and describe the mistakes we made along the way in addition to how developers responded to them!). In addition, I will also discuss what we learned from those research projects that never succeeded at Tech Transfer even though we tried. Finally, I will propose what I believe to be open problems in tech transfer and the way the research community interacts with software development practitioners today.

Biography

Christian Bird is a researcher in the empirical software engineering group at Microsoft Research. He is primarily interested in the relationship between software design, social dynamics, and processes in large development projects and in developing tools and techniques to help software teams. He has studied software development at Microsoft, IBM, and in the Open Source realm, examining the effects of distributed development, ownership policies, code review and the ways in which teams complete software tasks. He has published in the top Software Engineering venues including three ACM SIGSOFT Distinguished papers and Communications of the ACM. Christian received his B.S. in computer science from BYU and his Ph.D. from U.C. Davis. Christian has organized and run many events, including the Workshop on Release Engineering, the workshop on Data Analysis Patterns in Software Engineering, and the workshop on Replications in Software Engineering Research. He has served on the PCs of ICSE, FSE, ASE, MSR, ICSME, and ICPC, was the Industrial track PC co-chair of ICPC 2013, and is the PC co-chair of ICPC 2015.