1st International Workshop on Release Engineering (RELENG 2013)

Bram Adams*, Christian Bird†, Foutse Khomh‡ and Kim Moir§
*MCIS, École Polytechnique de Montréal, Montréal (Québec, Canada), bram.adams@polymtl.ca **[main contact]**
†Microsoft Research, Redmond (WA, United States), cbird@microsoft.com
‡SWAT, École Polytechnique de Montréal, Montréal (Québec, Canada), foutse.khomh@polymtl.ca
§Release Engineering Team, Mozilla, Ottawa (Ontario, Canada), kmoir@mozilla.com

***Abstract*—Release engineering deals with all activities in between regular development and actual usage of a software product by the end user, i.e., integration, build, test execution, packaging and delivery of software. Although research on this topic goes back for decades, the increasing heterogeneity and variability of software products along with the recent trend to reduce the release cycle to days or even hours starts to question some of the common beliefs and practices of the field. For example, a project like Mozilla Firefox releases every 6 weeks, generating updates for dozens of existing Firefox versions on 5 desktop, 2 mobile and 3 mobile desktop platforms, each of which for more than 80 locales. In this context, the International Workshop on Release Engineering (RELENG) aims to provide a highly interactive forum for researchers and practitioners to address the challenges of, find solutions for and share experiences with release engineering, and to build connections between the various communities.**

I. Workshop Outline

A. Theme

Software release engineering is the discipline of integrating [1], building [2], testing [1] [3], packaging [4] and delivering [5] qualitative software releases to the end user. These activities form the vital link between the design and development phases of a software product and the actual usage and maintenance of the finished product.

Whereas software used to be released in shrink-wrapped form, the advent of agile methodologies and the web has changed the landscape drastically. Deployment of modern applications often includes coordinating the release of applications on multiple mobile platforms, web platforms with centralized backend services, and native desktop clients. Furthermore, concepts like continuous delivery of software [6] are no longer curiosities, but essential to retain a competitive edge. For example, lean start-ups like IMVU release up to 50 times *per day* [7], while modern companies like Google [8] and Mozilla [9] only take a couple of weeks in between releases.

Coping with the dynamic nature of releasing ever-changing software and reducing the release cycle time of a software project require considerable process and technical changes, for example to automatically pick up new build system dependencies, speed up test execution without compromising confidence in the test results, or enable operators to roll back to a safe earlier release at the click of a button. Yet, the scope and nature of such changes are unknown to the majority of practitioners, for multiple reasons. First, the release engineering practices currently in use by successful software projects are mostly undocumented. The practices and infrastructure that have been studied are largely ad hoc or project-specific [6], and hence the underlying principles cannot easily be generalized. Second, since continuous delivery concepts like rolling releases have only recently become mainstream, not much is known about their long-term viability and deficiencies. Third, changing the release cycle of large, heterogeneous projects is still a major challenge companies are struggling with.

[1]I.e., test execution and test result reporting, not the actual creation of tests, since that typically occurs during actual development.

B. Goals

The RELENG workshop focuses on the state-of-the-art in release engineering, both on the major sub-fields of release engineering (integration, build, testing, packaging and delivery) as well as on their interaction with other activities like development and maintenance. The workshop goals are three-fold: (1) making researchers aware of the challenges and research opportunities for modern release engineering, (2) making practitioners (e.g., release engineers and developers) aware of the latest research results, and (3) building connections between the different communities involved in release engineering.

As such, the workshop does not aim to be a regular researcher-only workshop, but has been conceived from the ground up as a collaboration between practitioners and researchers, in the following sense:

- instead of only “regular” workshop papers, we also have a practitioner talk track specifically aimed to attract presentations by release engineers and other practitioners
- with 40% of the PC being an actual release engineer, each paper or talk submission received at least one review from a practitioner
- one of the co-organizers is a release engineer in a major open source organization (Mozilla)

C. Program

The workshop consisted of 2 keynotes, 6 practitioner talks, 10 paper presentations, interactive working groups and a fishbowl panel for semi-structured group discussions. The two keynotes by John O’Duinn, director of Release Engineering at

Mozilla Corporation, and Alan Grosskurth, Release Engineering consultant and ex-Release Engineer at VMWare, Inc. set the stage for the workshop, introducing the release engineering challenges of modern organizations. The topics discussed by this workshop included:

- best practices for code movement (branching/integration)
- continuous integration and testing
- build and configuration of software
- build system maintenance
- testing and reporting infrastructures
- package and dependency management
- legal signoff and bill-of-materials
- delivery and deployment of software
- code signing and certificate management
- continuous delivery, deployment, installation and software update
- cloud provisioning and management
- interaction with app stores
- principles and automated techniques for release planning
- release engineering for product line systems
- devops and interaction with developers, end users, etc.
- large-scale build and test farms
- multi-platform build and test

II. Program Committee

In order to provide feedback from both the academic and practitioner point of view, each paper was reviewed by 2 academic PC members and 1 practitioner. Below is the list of PC members.

PC members (researchers):

- Jan Bosch (Chalmers University of Technology)
- Arie van Deursen (TU Delft)
- Daniel M. German (University of Victoria)
- Michael Godfrey (University of Waterloo)
- Reid Holmes (University of Waterloo)
- Sarah Nadi (University of Waterloo)
- Mei Nagappan (Queen's University)
- Tien N. Nguyen (Iowa State University)
- Dewayne E. Perry (University of Texas at Austin)
- Adam Porter (University of Maryland)
- Slinger Roijackers Jansen (Utrecht University)
- Guenther Ruhe (University of Calgary)
- Andy Zaidman (TU Delft)
- Yuanyuan Zhang (University College London)

PC members (practitioners):

- Ray Cort (release engineer, Microsoft)
- Sonia Dimitrov (release engineer, IBM)
- Eelco Dolstra (cloud deployment, LogicBlox)
- Christina Ho (release engineer, Microsoft)
- Merijn de Jonge (chief software development, Sorama)
- John Ransier (release manager, Microsoft)
- Paul Reed (consultant, Releng Approaches)
- Hyrum Wright (software engineer, Google)
- Stefano Zacchiroli (project leader, Debian)

III. Organizers

Bram Adams (@mcis_lab) is an assistant professor at the École Polytechnique de Montréal (Canada). He obtained his PhD at the GH-SEL lab at Ghent University (Belgium), and was an adjunct assistant professor at SAIL, Queen's University (Canada). His research interests include software release engineering in general, as well as software integration and software build systems [2] in particular.

Christian Bird is a researcher at Microsoft Research (Redmond, United States). He received his PhD from the University of California at Davis under advisor Prem Devanbu. His main research interest is empirical software engineering, predominantly examining collaboration and coordination in large software teams in both industrial and open source contexts. He has studied many aspects of release engineering at Microsoft, most recently code movement in SCMs [10] and coordination issues between builders and developers.

Foutse Khomh is an assistant professor at the École Polytechnique de Montréal (Canada). He received his Ph.D in Software Engineering from the University of Montreal in 2010, under the supervision of Yann-Gaël Guéhéneuc. His main research interest is in the field of empirical software engineering, with an emphasis on developing techniques and tools to improve software quality. He has studied many aspects of the release engineering process of large software companies such as RIM [11] and Mozilla [12].

Kim Moir (@kmoir) works at Mozilla as a release engineer. Her interests lie in build optimization, scaling large infrastructure and writing about the complexities of open source release engineering. Previously, Kim spent eight years working for IBM as a release engineer in the Eclipse community.

References

[1] W. B. Frakes and K. Kang, "Software reuse research: Status and future," *IEEE Trans. Softw. Eng.*, vol. 31, pp. 529–536, July 2005.

[2] S. McIntosh, B. Adams, Y. Kamei, T. Nguyen, and A. E. Hassan, "An empirical study of build maintenance effort," in *Proc. of Intl. Conf. on Software Engineering (ICSE)*, 2011, pp. 141–150.

[3] J. A. Whittaker, J. Arbon, and J. Carollo, *How Google Tests Software*. Addison-Wesley Professional, April 2012.

[4] R. DeLine, "Avoiding packaging mismatch with flexible packaging," in *Proc. of Intl. Conf. on Software Engineering (ICSE)*, 1999, pp. 97–106.

[5] A. van der Hoek and A. L. Wolf, "Software release management for component-based software," *Softw. Pract. Exper.*, vol. 33, pp. 77–98, January 2003.

[6] J. Humble and D. Farley, *Continuous Delivery*, 1st ed. Addison Wesley, August 2010.

[7] T. Fitz, "Continuous deployment at IMVU: Doing the impossible fifty times a day," http://goo.gl/qPT6, February 2009.

[8] S. Shankland, "Google ethos speeds up Chrome release cycle," http://goo.gl/vNvlr, July 2010.

[9] ——, "Rapid-release Firefox meets corporate backlash," http://goo.gl/wgDQl, June 2011.

[10] C. Bird and T. Zimmermann, "Assessing the value of branches with what-if analysis," in *Proc. of the Intl. Symp. on Foundations of Software Engineering (FSE)*, 2012.

[11] T. Dhaliwal, F. Khomh, Y. Zou, and A. E. Hassan, "Recovering commit dependencies for selective code integration in software product lines," in *Proc. of the Intl. Conf. on Software Maintenance (ICSM)*, 2012.

[12] F. Khomh, T. Dhaliwal, Y. Zou, and B. Adams, "Do faster releases improve software quality? an empirical case study of mozilla firefox," in *Proc. of the Working Conf. on Mining Software Repositories (MSR)*, 2012.