Jan Jürjens, Kurt Schneider (Hrsg.): Software Engineering 2017,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2017 55

Factors Influencing Code Review Processes in Industry

Tobias Baum[1] Olga Liskin[2] Kai Niklas[3] Kurt Schneider[4]

Abstract: Code review is known to be an efficient quality assurance technique. Many software companies today use it, usually with a process similar to the patch review process in open source software development. However, there is still a large fraction of companies performing almost no code reviews at all. And the companies that do code reviews have a lot of variation in the details of their processes. We have performed a grounded theory study to clarify process variations and their rationales. The study is based on interviews with software development professionals from 19 companies. These interviews provided insights into the reasons and influencing factors behind the adoption or non-adoption of code reviews as a whole as well as for different process variations. We have condensed these findings into six hypotheses and a classification of the influencing factors. Our results show the importance of cultural and social issues for review adoption. They trace many process variations to differences in development context and in desired review effects.

Keywords: Code reviews, Empirical software engineering

There is a lot of evidence that Inspections and other types of code review are efficient software quality assurance techniques. Nevertheless, they are not adopted throughout the whole industry. In many companies that do code reviews, the review process is converging towards a lightweight process that is based on the regular review of changes [RB13] and that is similar to the processes used by many open source projects. Looking beyond the convergence in the general structure of the review process, we found a lot of variation in the details [Ba16a], a fact also observed in earlier studies [HTH05]. Which part of these variations is accidental, and which factors influence the choices taken by industrial practitioners?

To assess these and other questions, we performed a Grounded Theory [GS67] study based on interviews with 24 software engineering professionals from 19 companies. They described 22 different cases of code review use. The interviews were semi-structured, using open-ended questions. The broad research questions treated in the article are: *Why are code reviews used (or not used) by industrial software development teams?* and *Why are code reviews used the way they are?*

Of the studied companies, eleven have a regular code review process. The remaining eight only do irregular code reviews, i.e. code review is not codified in the team's process and used only on a personal basis, if at all.

We condensed our findings on the adoption of reviews into the following hypotheses: (1) Code review processes are mainly introduced or changed when a problem, i.e. a gap

[1] Leibniz Universität Hannover, FG Software Engineering, Hannover, tobias.baum@inf.uni-hannover.de
[2] Leibniz Universität Hannover, FG Software Engineering, Hannover, olga.liskin@inf.uni-hannover.de
[3] Leibniz Universität Hannover, FG Software Engineering, Hannover, kai.niklas@inf.uni-hannover.de
[4] Leibniz Universität Hannover, FG Software Engineering, Hannover, kurt.schneider@inf.uni-hannover.de

between some goal and reality, is perceived. (2) When code reviews are not used at all, this is mainly due to cultural and social issues. Needed time and effort are another important, but secondary, factor. (3) The importance of negative social effects decreases with time when reviews are in regular use. (4) Code review is most likely to remain in use if it is embedded into the process (and its supporting tools) so that it does not require a conscious decision to do a review.

Like previous studies [BB13], we found that teams use code review to reach a range of intended effects: Findings defects and better code quality, but for example also better knowledge distribution, forming of a shared culture and finding better solutions.

The interviewees mostly relied on their own or colleagues' experiences and considerations to gain knowledge on review processes. Web pages and practitioners' journals and conferences were also used, while scientific journals or conferences were not used at all.

We identified several contextual factors that influence the review process and grouped them into five categories: "Culture", "development team", "product", "development process" and "tool context". They influence the review process directly, mainly by making certain process variants infeasible or unattractive. But a major influence is also exerted indirectly, mediated through the intended combination of review effects. This does not mean that every process choice is made thoughtfully: Many minor process aspects just stay the way they were first tried. We formulated the following hypotheses for this subject area: (5) The intended and acceptable levels of review effects are a mediator in determining the code review process. (6) Model processes known from other teams or projects or coming from review tools have a large influence on many minor decisions shaping the code review process. Further information on this study can be found in [Ba16b].

References

[Ba16a] Baum, Tobias; Liskin, Olga; Niklas, Kai; Schneider, Kurt: A Faceted Classification Scheme for Change-Based Industrial Code Review Processes. In: Software Quality, Reliability and Security (QRS), 2016 IEEE International Conference on. IEEE, 2016.

[Ba16b] Baum, Tobias; Liskin, Olga; Niklas, Kai; Schneider, Kurt: Factors Influencing Code Review Processes in Industry. In: Proceedings of the ACM SIGSOFT 24th International Symposium on the Foundations of Software Engineering. ACM, 2016.

[BB13] Bacchelli, Alberto; Bird, Christian: Expectations, outcomes, and challenges of modern code review. In: Proceedings of the 2013 International Conference on Software Engineering. IEEE Press, pp. 712–721, 2013.

[GS67] Glaser, B.G.; Strauss, A.L.: The Discovery of Grounded Theory: Strategies for Qualitative Research. Aldine, 1967.

[HTH05] Harjumaa, Lasse; Tervonen, Ilkka; Huttunen, Anna: Peer reviews in real life-motivators and demotivators. In: Quality Software, 2005.(QSIC 2005). Fifth International Conference on. IEEE, pp. 29–36, 2005.

[RB13] Rigby, Peter C; Bird, Christian: Convergent contemporary software peer review practices. In: Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering. ACM, pp. 202–212, 2013.