

# Improving Software Engineering in Academia

Samuel Grayson

## Related studies

Eisty and Carver [2] study “What is the effect of peer code review on research software engineering? What are common positive and negative experiences? What are difficulties and barriers? What improvements can be made?” They found its practitioners believe that its effect is to improve code quality and facilitate knowledge sharing. These effects are also “positive experiences.” Negative experiences include review taking too long and developers misunderstanding the feedback. Difficulties include the challenge of understanding the code, while barriers include finding the time to do reviews. Potential improvements include formalizing process, using more software tools, and funding more people to do reviews.

Collberg and Proebsting [1] study “How much of published computer systems research in 2012 is repeatable?” The researchers found that roughly 2/3 had source code, and of those, about 1/2 were repeatable out-of-the-box. The results has been questioned in follow up work by Krishnamurthi [3] because perhaps those tasked with reproducing were not knowledgeable enough.

Vandewalle et al. [5] study “Does releasing source code correlate with increased citation-count for image processing in 2004?” The researchers found a correlation, but leave causality to future work. About 1/10 of their samples had source code in image processing journals in 2004; this is strikingly different from Collberg [1], perhaps because the domain and time were different.

Murphy-Hill et al. [4] study “What makes software developers productive in 2019?” They found that the most important are: job enthusiasm, peer support for new ideas, useful feedback about job performance.

## Software Development Practices

- **Peer code review:** studied by [2]

## Methods

Eisty and Carver [2] use survey to study how a process works, what are the positives, and what are the negatives. They do a pilot study first. They never directly assess efficacy of that process, just “do the practitioners find it important?”

Collberg [1] have undergraduate researchers attempt to repeat research experiments. Krishnamurthi [3] shows that they may not be knowledgeable enough and erroneously declare software not repeatable.

Vandewalle [5] uses citation count to measure research impact and searches for source-code by hand.

Murphy-Hill [4] uses surveys to quantify environmental variables and self-assessment to quantify productivity.

- [1] Christian Collberg and Todd A. Proebsting. 2016. Repeatability in computer systems research. *Communications of the ACM* 59, 62–69. DOI:<https://doi.org/10.1145/2812803>
- [2] Nasir U. Eisty and Jeffrey C. Carver. Developers Perception of Peer Code Review in Research Software Development. Retrieved from <https://arxiv.org/abs/2109.10971>
- [3] Shriram Krishnamurthi. Examining “Reproducibility in Computer Science.” Retrieved from <http://cs.brown.edu/~sk/Memos/Examining-Reproducibility/>
- [4] Emerson Murphy-Hill, Ciera Jaspan, Caitlin Sadowski, David Shepherd, Michael Phillips, Collin Winter, Andrea Knight, Edward Smith, and Matthew Jorde. What Predicts Software Developers’ Productivity? *IEEE Transactions on Software Engineering* 47. DOI:<https://doi.org/10.1109/TSE.2019.2900308>
- [5] Patrick Vandewalle, Jelena Kovacevic, and Martin Vetterli. *IEEE Singal Processing Magazine* 26. DOI:<https://doi.org/10.1109/MSP.2009.932122>