Ethics of Care for Software Engineering

Alexander Serebrenik
a.serebrenik@tue.nl
Eindhoven University of Technology
The Netherlands

Sebastian Baltes
sebastian.baltes@uni-heidelberg.de
Heidelberg University
Germany

Abstract

Software engineering researchers repeatedly argue that the impact of their research on industrial practice, while desired and intended, is rarely achieved. We believe that a possible explanation of this phenomenon is the opposition of "caring about" and "caring for", based on the *ethics of care*. Indeed, while software engineering is collaborative and hence builds on interpersonal relations, researchers tend to care about "industrial impact" and "practitioners" in abstract terms, but rarely care for specific individuals working in specific contexts facing specific challenges. In this position paper, we advocate for the adoption of ethics of care in software engineering and discuss the implications of this adoption for researchers and conference organizers.

Keywords

software engineering, academia-industry collaboration, ethics of care

ACM Reference Format:
Alexander Serebrenik and Sebastian Baltes. 2018. Ethics of Care for Software Engineering. In *Proceedings of FoSE (Conference acronym 'XX)*. ACM, New York, NY, USA, 2 pages. https://doi.org/XXXXXXX.XXXXXXX

A recent survey of software engineering researchers conducted by Margaret-Anne Storey and Andre van der Hoek as part of the Future of Software Engineering track [19] surfaced multiple concerns about the divergence between academic research in software engineering and industrial practice. When asked what aspect or aspects of the software engineering research community do not work well and why, respondents lamented "*minimal impact on industry and almost no relevance to adjacent fields like AI*", "*lack of industrial relevance*", the research community being too "*far from real issues and problems found by practitioners*", and engaging in "*too much navel gazing <...> and not enough focused on translating results to industry and impact*". This concern is not unique to the survey respondents. "*Software engineering research has had as much impact on programmers as astronomy has had on stars,*" as Greg Wilson has aptly put it.[1] However, at the same time, survey respondents regret that, compared to other computer science researchers, software engineering researchers are "*often perceived as less relevant, less impactful, and overly focused on industrial application*" and remind that as researchers "*we are not competing with industry, we are doing something different than them, and we often forget this.*" The fundamental question is the role of the industry vs. academia relation in software engineering. To what extent is engagement with the industry fundamental for our discipline or is it merely the way it is currently being organized?

A possible answer is that, in return for rewarding careers, good salaries, and public recognition, we as academics have the duty to generate useful knowledge for the respective state and to train a workforce capable of applying it [14]. Given that most universities are paid by the government, a close connection to the needs of the state, which is paying the salaries and other expenses might, however, threaten the validity of scientific results. In extreme cases, this might lead to the suppression of research areas, for example, the suppression of genetics in the Soviet Union and the creation of pseudoscience such as Lysenkoism [8].

Another angle is that (applied) research can be judged on its application in practice. This is particularly true in software engineering, where the usefulness of knowledge becomes equated with the impact on industrial practice. However, while science can be "*the pacemaker of technological progress*" [2], it has also been argued that the value of a scientific inquiry cannot be reduced to the set of prospective yet uncertain technological applications (cf. discussion of purism by Kitcher [10]).

However, Sommerville, in his classic software engineering textbook, argues that "*software engineering is intended to support professional software development, rather than individual programming*" [17]. This means that software engineering as a discipline can only exist in relation to the practice of software development. As this practice is carried out by people, we interpret Sommerville's definition as a recognition that **software engineering as a discipline is defined through its relation with and for people**. The ethics of care considers relationships not from a legalistic point of view but as having an intrinsic unconditional and non-negotiable value [18]. This, together with our interpretation of software engineering as a discipline, means that **software engineering should be subject to the ethics of care** [7, 12].

The ethics of care is based on three ideas [15]. First, people are fundamentally dependent on each other. Empirical software engineering researchers inherently depend on practitioners, as researchers study the process enacted by practitioners or artifacts created by them. Second, the ethics of care calls for attention to the most vulnerable when making decisions. There is increasing attention to making software development more diverse and inclusive [9]. Finally, the ethics of care differentiates between a more abstract notion of "caring about" and more immediate "caring for" [12], and argues that our moral choices should attend and respond to the immediate conditions of our context.

[1] https://third-bit.com/talks/to-dont/#3

Conference acronym 'XX, 978-1-4503-XXXX-X/2018/06

ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
https://doi.org/XXXXXXX.XXXXXXX

It is particularly the last opposition between "caring about" and "caring for" contains promise of resolving the opposition between software engineering researchers trying to impact the industry but reaching a limited success. Therefore, we pose the following question:

> *Do software engineering researchers really "care for" software practitioners, that is, specific individuals developing software or do they rather "care about" practical impact on an abstract level?*

This opposition calls for the replacement of generic "developers", "companies", and "projects" that we care about in our research with specific developers, companies, and projects that we care for. Trying to impact software development practice in all its diversity might not be feasible; however, we can and should aim to make software development a better place for individuals.

In addition, individual researchers should continue to work with people from vulnerable or underprivileged populations within software engineering. Recent studies have considered the experiences of software developers from minoritized groups such as women [3], neurodiverse developers [6, 11], racial and ethnic minority developers [4], LGBTIQ+ developers [5], older developers [1], developers with disabilities [16], developers from the Global South [13] and intersections of these identities [20, 21]. This list is by no means exhaustive and should not be limited to demographic diversity. Perspectives of developers from small companies that do not necessarily have direct connections to universities or academic research, of scientists trained outside of computer science and developing software to support their research, or end-users creating websites with Generative AI should be explored and included in our view of software engineering. The "caring for" principle requires deep and prolonged engagement with those populations, and acknowledgment of the right of these individuals to be who they are [18].

Similarly, conference organizers should prioritize topics that affect practitioners as opposed to those that simply improve technology, and studies conducted in a specific context and having had a positive impact in this context. **We call for each and every software engineering paper to present a story of a real person** who develops software (whether seeing themselves as a software engineer or not), or a person who teaches or researches software development. This presentation should include a description of their specific context and the specific challenges they face, e.g., related to the technology they are using, their way of working, or their workplace environment. Such a story—anonymized or not—can strengthen our connection to practice, answer the fundamental question why are we doing software engineering research (because we care), and hopefully help to achieve the impact sought after by the survey respondents.

Acknowledgments

The authors thank Mary Shaw and Titus Winters for sharing their thoughts on the topic on the Discord channel.

References

[1] Sebastian Baltes, George Park, and Alexander Serebrenik. 2020. Is 40 the New 60? How Popular Media Portrays the Employability of Older Software Developers. *IEEE Softw.* 37, 6 (2020), 26–31.

[2] Vannevar Bush. 1945. *Science—The Endless Frontier: A Report to the President.* United States Government Printing Office.

[3] Claudia Maria Cutrupi, Letizia Jaccheri, and Alexander Serebrenik. 2026. Gender Diversity Interventions in Software Engineering: A Comprehensive Review of Existing Practices. *Comput. Sci. Rev.* 59 (2026), 100812. doi:10.1016/J.COSREV.2025.100812

[4] Ella Dagan, Anita Sarma, Alison Chang, Sarah D'Angelo, Jill Dicker, and Emerson R. Murphy-Hill. 2023. Building and Sustaining Ethnically, Racially, and Gender Diverse Software Engineering Teams: A Study at Google. In *FSE*, Satish Chandra, Kelly Blincoe, and Paolo Tonella (Eds.). ACM, 631–643.

[5] Ronnie Edson de Souza Santos, Cleyton V. C. de Magalhães, and Paul Ralph. 2023. Benefits and Limitations of Remote Work to LGBTQIA+ Software Professionals. In *ICSE SEIS*. IEEE, 48–57. doi:10.1109/ICSE-SEIS58686.2023.00011

[6] Kiev Gama, Grischa Liebel, Miguel Goulão, Aline Lacerda, and Cristiana Lacerda. 2025. A Socio-Technical Grounded Theory on the Effect of Cognitive Dysfunctions in the Performance of Software Developers with ADHD and Autism. In *ICSE*. IEEE, 1–12. doi:10.1109/ICSE-SEIS66351.2025.00006

[7] Carol Gilligan. 1993. *In a Different Voice: Psychological Theory and Women's Development.* Harvard University Press.

[8] Sandra Harding. 1991. *Whose Science? Whose Knowledge?: Thinking from Women's Lives.* Cornell University Press.

[9] Sonja M. Hyrynsalmi, Sebastian Baltes, Chris Brown, Rafael Prikladnicki, Gema Rodríguez-Pérez, Alexander Serebrenik, Jocelyn Simmonds, Bianca Trinkenreich, Yi Wang, and Grischa Liebel. 2025. Making Software Development More Diverse and Inclusive: Key Themes, Challenges, and Future Directions. *ACM Trans. Softw. Eng. Methodol.* 34, 5 (2025), 134:1–134:23. doi:10.1145/3711904

[10] Philip Kitcher. 2001. *Science, Truth, and Democracy.* OUP USA.

[11] Kaia Newman, Sarah Snay, Madeline Endres, Manasvi Parikh, and Andrew Begel. 2025. "Get Me in the Groove": a Mixed Methods Study on Supporting Adhd Professional Programmers. In *ICSE*. IEEE, 1217–1229. doi:10.1109/ICSE55347.2025.00242

[12] Nel Noddings. 2003. *Caring: A Feminine Approach to Ethics and Moral Education.* University of California Press. https://books.google.nl/books?id=vkMkLi6pnMYC

[13] Chaiyong Ragkhitwetsagul, Morakot Choetkiertikul, Srisupa Palakvangsa-Na-Ayudhya, Thanwadee Sunetnanta, and Nattanee Satchanawakul. 2025. The Impact of COVID-19 and Remote Work on Software Development in Thailand. In *International Conference on Information Technology*. 265–272. doi:10.1109/InCIT66780.2025.11276124

[14] Reese A. K. Richardson, Spencer S. Hong, Jennifer A. Byrne, Thomas Stoeger, and LuÃs A. Nunes Amaral. 2025. The entities enabling scientific fraud at scale are large, resilient, and growing rapidly. *Proceedings of the National Academy of Sciences* 122, 32 (2025), e2420092122. arXiv:https://www.pnas.org/doi/pdf/10.1073/pnas.2420092122 doi:10.1073/pnas.2420092122

[15] Andrew L. Russell and Lee Vinsel. 2019. Make Maintainers: Engineering Education and an Ethics of Care. In *Does America Need More Innovators?*, Matthew Wisnioski, Eric S. Hintz, and Marie Stettler Kleine (Eds.). The MIT Press, Chapter 13.

[16] Clark Saben, Jessica Zeitz, and Prashant Chandrasekar. 2024. Enabling Blind and Low-Vision (BLV) Developers with LLM-Driven Code Debugging. *J. Comput. Sci. Coll.* 40, 3 (2024), 204–215. doi:10.5555/3722479.3722531

[17] Ian Sommerville. 2011. *Software Engineering.* Pearson.

[18] Robert J. Starratt. 1991. Building an Ethical School: A Theory for Practice in Educational Leadership. *Educational Administration Quarterly* 27, 2 (1991), 185–202.

[19] Margaret Storey and Andre van der Hoek. 2025. *Community Survey for ICSE 2026 Future of Software Engineering: Toward a Healthy Software Engineering Community.* doi:10.5281/zenodo.18217799

[20] Anna Szlávi, Marit Fredrikke Hansen, Sandra Helen Husnes, Tayana Uchôa Conte, and Letizia Jaccheri. 2024. Designing for Intersectional Inclusion in Computing. In *Universal Access in Human-Computer Interaction - 18th International Conference, UAHCI 2024, Held as Part of the 26th HCI International Conference, HCII 2024, Washington, DC, USA, June 29 - July 4, 2024, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 14696)*, Margherita Antona and Constantine Stephanidis (Eds.). Springer, 122–142. doi:10.1007/978-3-031-60875-9_9

[21] Sterre van Breukelen, Ann Barcomb, Sebastian Baltes, and Alexander Serebrenik. 2023. "STILL AROUND": Experiences and Survival Strategies of Veteran Women Software Developers. In *ICSE*. IEEE, 1148–1160.