Editor in Chief: **Ipek Ozkaya**
Carnegie Mellon Software Engineering Institute,
ipek.ozkaya@computer.org

The Future of Software Engineering Work

Ipek Ozkaya

SOFTWARE ENGINEERING WORK is inherently collaborative and requires coordination. Developing a shared understanding of the artifacts to be developed is an essential part of this process. While being distributed adds an additional layer of complexity to collaboration, distributed and remote work is not new to software engineering. The post-COVID-19 era is challenging many organizations, including those whose main business is software, to revisit their procedures and assumptions about how work is conducted. Despite the fact that software engineers are not foreign to remote collaborative work, the postpandemic changes to where work is conducted will require us to revisit assumptions about software engineering work, productivity, and collaboration. There are several questions to ask: How is work conducted most effectively? What best serves the well-being of software engineers? How do we assess productivity, cost, and benefit compared to traditional models of collaborative software engineering work?

In this article, I discuss the future of software engineering work from the perspective of potential challenges and opportunities in the post-COVID-19 era. Organizations are grappling with figuring out what a return to the office looks like after a significant proportion of the software engineering workforce worked remotely for more than a year and many continue to do so globally. Companies will need to manage the challenges of finding top software engineering talent, mentoring and growing them internally in their organizations, and striking a productivity balance as they provide work environment flexibility to their employees. Our understanding of which approaches can achieve the most effective collaboration, coordination, and communication will likely shift.

> While being distributed adds an additional layer of complexity to collaboration, distributed and remote work is not new to software engineering.

A New Normal: Hybrid Work

The ability to work from anywhere and the flexibility to do so are now emerging as an expected perk as top software engineering talent seeks employment opportunities. As early as 2020 May during the early months of the COVID-19 pandemic, global software engineering organizations,

Digital Object Identifier 10.1109/MS.2021.3089729
Date of current version: 20 August 2021

IEEE Software Mission Statement

To be the best source of reliable, useful, peer-reviewed information for leading software practitioners—the developers and managers who want to keep up with rapid technology change.

0740-7459/21©2021IEEE

CONTACT US

AUTHORS

For detailed information on submitting articles, visit the "Write for Us" section at www.computer.org/software

LETTERS TO THE EDITOR

Send letters to software@computer.org

ON THE WEB

www.computer.org/software

SUBSCRIBE

www.computer.org/subscribe

SUBSCRIPTION CHANGE OF ADDRESS

address.change@ieee.org
(please specify *IEEE Software*.)

MEMBERSHIP CHANGE OF ADDRESS

member.services@ieee.org

MISSING OR DAMAGED COPIES

contactcenter@ieee.org

REPRINT PERMISSION

IEEE utilizes Rightslink for permissions requests. For more information, visit www.ieee.org/publications/rights/rights-link.html

such as Twitter and Square, announced a permanent remote-work policy for their employees. Twitter's announcement was only accelerated by the pandemic. The organization had already observed increased productivity when software engineers worked from home or locations of their choice, and it had plans to offer this as an option to its workforce. Others, such as Spotify, followed with a model of "work from anywhere," providing flexibility of choice to their employees, including a benefit of paying for subscriptions to coworking spaces. However, assuming that all software engineering work will go all remote for all small and large organizations would be naive.

The new normal is not going to be all remote work, but it will be *hybrid*: while the flexibility of working remotely will be offered to all, an expectation of regular in-person interaction will not disappear. Defining consistently what hybrid work looks like for software engineering in general is not likely to happen anytime soon. Disparate local as well as organizational needs and requirements will influence the definition of different work models. For the purposes of this article, I define hybrid work as a work model where an engineer will have the flexibility to choose to work a number of days remotely and a number of days in the office with in-person communication. While many of the lessons learned in global and distributed software development coordination and collaboration will likely apply, the increased flexibility of hybrid work will likely introduce an added level of complexity.

Collaboration While Performing Hybrid Work: Challenges and Opportunities

The software engineering community has studied collaborative software development, particularly in the context of global and distributed software development.[1] Developers who are not collocated, working together to build large software systems where many software engineers must collaborate and deal with issues of geographical, temporal, cultural, and language diversity, are central to global and distributed software development. A key challenge in distributed software development is ensuring timely, on-task communication and coordination to avoid mismatches among the many artifacts that move through the engineering of software. Hybrid work will likely exacerbate the existing challenges of collaboration. Among other issues, they include the following.

More or Better Tools Will Not Be the Ultimate Solution

The solutions to address communication challenges inevitably focus on the development of coordination and communication tools: Zoom, Teams, Slack, Trello, Mural, Miro, Ideaflip, Google Suite, email, and text, to name just a few. None of these tools are software engineering task specific; they are general coordination and communication tools. By all means, the specialized software coordination, development, and version control tools such as GitHub fill some similar needs as well. However, more tools or tools with more features do not guarantee better coordination and communication.[2] This will be particularly true in hybrid work mode. We are yet to experience the new communication barriers that will be introduced when some workers are collocated while others are remote, and when the distribution of each changes on a constant basis if there is flexibility.

The Need to Reduce Dependencies Among Engineers Will Increase

By now we all have experienced the joy of small (two to five attendees), medium (10–15 attendees), large (15–25 attendees), and why-am-I-here (more than 25 attendees) online meetings. In a hybrid work model, online and asynchronous connection and coordination will be a default mode of communication. As organizations add the flexibility of hybrid work, they initially will be adding unavoidable complexity of managing coordination to their teams. Focusing the dependencies on those tasks where coordination is essential and unavoidable and eliminating the rest will improve productivity as well as system quality and sustainability. The ability to develop software where dependencies are minimized will require software architecture and design thinking. Consequently, we will look for software architecting and design competencies as desirable skill sets in all software engineers—significantly more so in this new hybrid work model than it is now.

Organizational Memory Will Increasingly Have to Be Retained in Artifacts Rather Than Team Members

Recording organizational memory is one of the outcomes of effective software engineering collaboration.[3] As we introduce flexibility to software engineers with new working models, we will be increasingly removing informal, yet essential, modes of retention of organizational memory, which include, but is not limited to, "chat a minute or two before the meeting starts," "catch up at the water cooler," or "grab coffee or lunch." We often underestimate the value of these exchanges for information sharing and retention. They are also essential for staff growth, mentoring, and peer learning. The unavoidable consequence of reducing these activities will be the need to develop processes, hand-offs, and tools that, as a natural consequence, record the decision rationale that is essential for a project's history and in determining how organizational knowledge is created and retained.

Opportunities

We also cannot ignore the opportunities that hybrid work is likely to offer to software engineering. First and foremost, the assumption is that the ability to offer flexibility to engineers will allow them to balance their priorities and, consequently, improve their well-being as well as their productivity. The data on this topic are mixed. For example, early studies found that work-from-home settings during COVID-19 promoted inequality, especially for individuals who typically took on more of the domestic responsibilities.[4] The ability to establish fair work structures that do not handicap any portion of the workforce will be critical. Our mental models have shifted significantly; we realize that some communication tasks can be conducted just as effectively remotely as in person. This increased flexibility will also increase our ability to include a different variety of stakeholders in the development process; a task that otherwise may have been harder to achieve due to travel, schedule, and other communication barriers. Opportunities range from improved user and customer collaboration during development to ability to more easily reach out to domain experts.

EDITORIAL STAFF

IEEE SOFTWARE STAFF

Managing Editor: Jessica Welsh, j.welsh@ieee.org
Cover Design: Andrew Baker
Peer Review Administrator: software@computer.org
Publications Staff Editor: Cathy Martin
Publications Operations Project Specialist: Christine Anthony
Content Quality Assurance Manager: Jennifer Carruth
Publications Portfolio Manager: Carrie Clark
Publisher: Robin Baldwin
IEEE Computer Society Executive Director: Melissa Russell
Senior Advertising Coordinator: Debbie Sims

CS PUBLICATIONS BOARD

M. Brian Blake (VP of Publications), David Ebert, Elena Ferrari, Hui Lei, Timothy Pinkston, Antonio Rubio Sola, Diomidis Spinellis, Tao Xie, Ex officio: Robin Baldwin, Sarah Malik, Melissa Russell, Forrest Shull

CS MAGAZINE OPERATIONS COMMITTEE

Diomidis Spinellis (MOC Chair), Lorena Barba, Irena Bojanova, Shu-Ching Chen, Gerardo Con Diaz, Lizy K. John, Marc Langheinrich, Torsten Möller, Ipek Ozkaya, George Pallis, Sean Peisert, VS Subrahmanian, Jeffrey Voas

IEEE PUBLICATIONS OPERATIONS

Senior Director, Publishing Operations: Dawn M. Melley
Director, Editorial Services: Kevin Lisankie
Director, Production Services: Peter M. Tuohy
Associate Director, Information Conversion and Editorial Support: Neelam Khinvasara
Senior Managing Editor: Geraldine Krolin-Taylor
Senior Art Director: Janet Dudar

Editorial: All submissions are subject to editing for clarity, style, and space. Unless otherwise stated, bylined articles and departments, as well as product and service descriptions, reflect the author's or firm's opinion. Inclusion in *IEEE Software* does not necessarily constitute endorsement by IEEE or the IEEE Computer Society.

To Submit: Access the IEEE Computer Society's Web-based system, ScholarOne, at http://mc.manuscriptcentral.com/sw-cs. Be sure to select the right manuscript type when submitting. For complete submission information, please visit the Author Information menu item under "Write for Us" on our website: www.computer.org/software.

IEEE prohibits discrimination, harassment and bullying: For more information, visit www.ieee.org/web/aboutus/whatis/policies/p9-26.html.

Digital Object Identifier 10.1109/MS.2021.3101062

A Call to Action

Despite the challenges, the ability to offer more flexibility to the tech workforce when the potential negative consequences are managed will benefit software engineers with an increased focus on tasks that require more of their attention. This will consequently result in better quality of our software products. Open questions to address for improving the future of software engineering work in this upcoming hybrid mode include the following:

More tools or tools with more features do not guarantee better coordination and communication.

- Are there unanticipated additional and exasperating hand-offs as software engineers shift constantly from remote to in-person mode in a hybrid working model?
- What are the drawbacks of permanent remote-work models?
- How are software developer and engineer productivity and well-being impacted in different models of work?
- Do our existing mental models of collaboration and coordination and the tools to support them continue to function in various modes of hybrid work?
- What are the potential opportunities for errors and ambiguities that hybrid work models may exacerbate or introduce?
- How can design approaches that involve independent development and deployment of software elements further assist new team distribution models?
- Will the mirroring assumptions of software and organization structures, known as *Conway's law*,[5] still hold with hybrid work modes?

In this article, I focused only on the collaboration and communication aspects of hybrid work modes for the future of software engineering work. There are many other aspects that I did not discuss, including whether our known concepts of productivity are likely to change. There will be new opportunities and challenges for onboarding team members with different levels of experience. Some skill sets will likely have a higher priority to be effective in this different working environment, including specialized software engineering skills, such as increased expected competencies in software architecture and design. In this issue, we focus in depth on software productivity. The focus articles range from questioning our current understanding of the relationship between automatically measured and self-reported productivity and the impact of time pressure in software development to clarifying the relationship between agile software development and project management as well as DevOps and organizational maturity. I encourage the readers to read these articles also with a perspective on whether their lesson learned potentially transfer to hybrid work settings.

References

1. J. D. Herbsleb and D. Moitra, "Global software development," *IEEE Softw.*, vol. 18, no. 2, pp. 16–20, 2001. doi: 10.1109/52.914732.
2. K. Constantino, S. Zhou, M. Souza, E. Figueiredo, and C. Kästner, "Understanding collaborative software development: An interview study," in *Proc. 15th Int. Conf. Global Softw. Eng. (ICGSE '20)*, New York, 2020, 55–65. doi: 10.1145/3372787.3390442.
3. J. Whitehead, "Collaboration in software engineering: A roadmap," in *Proc. Future Softw. Eng.*, 2007, 214–225. doi: 10.1109/FOSE.2007.4.
4. L. S. Machado, C. Caldeira, M. G. Perin, and C. R. B. de Souza, "Gendered experiences of software engineers during the COVID-19 crisis," *IEEE Softw.*, vol. 38, no. 2, pp. 38–44, 2021. doi: 10.1109/MS.2020.3040135.
5. J. D. Herbsleb and R. E. Grinter, "Splitting the organization and integrating the code: Conway's law revisited," in *Proc. 21st Int. Conf. Softw. Eng.*, 1999, pp. 85–95. doi: 10.1145/302405.302455.