International Journal of Science and Research Archive
eISSN: 2582-8185
Cross Ref DOI: 10.30574/ijsra
Journal homepage: https://ijsra.net/

(REVIEW ARTICLE)

Check for updates

Operationalizing invisible work in engineering teams through observable collaboration signals

Satvik Bhasin *

Drexel University, Philadelphia, Pennsylvania, USA.

International Journal of Science and Research Archive, 2026, 18(03), 1564-1574

Publication history: Received on 28 January 2026; revised on 26 March 2026; accepted on 29 March 2026

Article DOI: https://doi.org/10.30574/ijsra.2026.18.3.0468

Abstract

Invisible work in engineering teams—encompassing coordination, mentoring, communication, and knowledge sharing—has long been recognized as essential yet underrepresented in traditional productivity metrics. With the rise of collaborative software development environments and digital work platforms, a growing body of research suggests that such hidden contributions can be partially inferred through observable collaboration signals, including code review interactions, issue discussions, and communication patterns. This review synthesizes theoretical foundations, empirical evidence, and emerging methodologies to examine how invisible work can be operationalized using these signals.

The paper first situates invisible work within socio-technical systems theory, emphasizing that engineering productivity is not solely a function of code output but also of alignment between technical dependencies and human collaboration. It then reviews empirical studies demonstrating that collaboration signals—such as review participation, discussion quality, and network structure—are strongly associated with outcomes like software quality, contribution acceptance, and team effectiveness. Building on this foundation, the review proposes a conceptual model that links digital traces to latent forms of invisible work, supported by block diagrams and experimental synthesis.

The findings highlight both the promise and the limitations of using observable signals as proxies. While collaboration traces provide a scalable and data-driven way to surface hidden contributions, they require careful interpretation due to their context-dependent nature. The review also identifies key challenges, including measurement validity, ethical concerns related to surveillance, and the risk of metric misuse.

Overall, this article contributes a structured framework for understanding and measuring invisible work in engineering teams. It offers guidance for researchers seeking to refine socio-technical metrics and for practitioners aiming to design fairer and more comprehensive evaluation systems. Future work should focus on integrating multi-modal data sources, improving interpretability, and developing ethical guidelines for responsible use of collaboration analytics.

Keywords: Invisible Work; Software Engineering; Collaboration Signals; Socio-Technical Systems; Code Review; Developer Productivity; Team Coordination; Digital Trace Data; Engineering Analytics; Human-Centered Metrics

1. Introduction

In contemporary engineering environments, particularly within software development and large-scale systems engineering, work is increasingly collaborative, distributed, and interdependent. While formal outputs such as code commits, tickets resolved, or features delivered are easily tracked and quantified, a substantial portion of engineering effort remains largely invisible. This "invisible work" includes activities such as mentoring junior engineers, coordinating across teams, reviewing code, resolving ambiguities, and maintaining shared understanding—tasks that

* Corresponding author: Satvik Bhasin

are essential for system reliability and team productivity but are often under-recognized in traditional performance and productivity metrics [1], [2]. As organizations continue to adopt agile methodologies, DevOps practices, and remote or hybrid work models, the proportion and importance of such invisible contributions have grown significantly.

The increasing reliance on collaborative workflows has amplified the need to better understand and operationalize these hidden dimensions of work. Engineering teams today depend heavily on socio-technical systems—integrations of human collaboration and technical infrastructure—where communication patterns, coordination mechanisms, and informal knowledge sharing play a critical role in determining outcomes [3]. Tools such as version control systems, issue trackers, and communication platforms (e.g., GitHub, Jira, Slack) generate large volumes of interaction data, offering new opportunities to capture and analyze collaboration signals. These observable signals—such as code review interactions, discussion threads, and contribution networks—can serve as proxies for otherwise invisible work, enabling more holistic assessments of productivity and team health [4].

The importance of this topic is underscored by several ongoing challenges in the broader field of software engineering and organizational studies. First, traditional productivity metrics often fail to capture the complexity of collaborative work, leading to incomplete or biased evaluations of individual and team performance [5]. This can result in misaligned incentives, undervaluation of critical coordination roles, and reduced morale among contributors whose work is less visible. Second, the rise of remote and globally distributed teams has made informal, co-located interactions less frequent, increasing the reliance on digital traces to understand team dynamics [6]. Third, there is growing recognition that effective collaboration is a key determinant of software quality, innovation, and resilience, yet systematic methods for measuring and leveraging collaboration remain underdeveloped [7].

Despite increasing interest in this area, several gaps persist in current research. Existing studies often focus on easily measurable artifacts such as code contributions, overlooking the nuanced and context-dependent nature of collaborative work. Moreover, while some research has explored social network analysis and communication patterns within engineering teams, there is limited consensus on how to translate these insights into actionable metrics or organizational practices [8]. Another key challenge lies in balancing observability with ethical considerations, such as privacy, surveillance concerns, and the potential misuse of behavioral data [9]. Additionally, there is a lack of integrative frameworks that connect observable collaboration signals with broader organizational outcomes, such as team effectiveness, innovation capacity, and employee well-being.

This review aims to address these challenges by synthesizing existing research on invisible work in engineering teams and examining how observable collaboration signals can be used to operationalize and better understand these contributions. Specifically, the review will explore the theoretical foundations of invisible work, survey current methodologies for capturing collaboration data, and evaluate the strengths and limitations of different approaches. It will also discuss emerging tools and techniques for analyzing collaboration signals, as well as the ethical and organizational implications of their use. By providing a comprehensive and structured overview of this evolving field, this review seeks to inform both researchers and practitioners on how to more effectively recognize, measure, and support the full spectrum of work that underpins successful engineering teams.

Table 1 Research on Invisible Work and Observable Collaboration Signals

Reference	Key results
[10]	This work helped establish that engineering performance depends not only on technical architecture but also on whether people communicate across relevant dependencies. It is foundational for studying invisible coordination work because it shows that collaboration itself is part of productive engineering work, even when it does not directly appear in output metrics.
[3]	The study shows that coordination demands can be systematically identified from technical dependencies, making it possible to infer hidden collaboration needs from observable work structures. It supports the idea that invisible work can be partially surfaced through collaboration signals embedded in engineering tools.
[11]	The authors found that communication embedded in issue trackers plays a major role in coordinating work and predicting outcomes such as bug resolution. This is important because it highlights that collaboration traces are not peripheral—they are central to understanding team effectiveness.
[12]	The paper demonstrates that communication breakdowns across distributed settings increase coordination problems, while stronger social interaction improves collaborative effectiveness. It

	reinforces the importance of capturing invisible cross-site coordination when evaluating engineering work.
[13]	Findings suggest that organizational and communication structures influence defect rates and software quality. The paper is significant because it ties invisible coordination practices directly to technical outcomes, supporting the broader argument that collaboration signals can reveal otherwise hidden drivers of performance.
[14]	The study shows that pull-based development involves substantial unseen labor, including negotiation, feedback incorporation, and waiting on reviewers. These activities are often absent from simple productivity counts, yet they strongly shape delivery and participation.
[5]	Developers reported that productivity is affected by interruptions, collaboration quality, and the ability to make progress with others—not just by coding volume. This is especially relevant for invisible work research because it challenges narrow output-based metrics and validates the importance of teamwork signals.
[15]	Although focused on repository popularity, the study shows how discussion, responsiveness, and community interaction contribute to project success. It demonstrates that non-code collaboration signals can meaningfully reflect health and influence in engineering communities.
[16]	The paper shows that code review is not only a defect-detection mechanism but also a venue for knowledge sharing, mentoring, and coordination. This makes it a strong example of invisible work becoming observable through engineering workflow data.
[4]	The study finds that developers rely on a mix of formal and informal channels to coordinate, learn, and maintain awareness. It concludes that communication ecosystems are central to participatory engineering culture, making them valuable sources of observable signals for hidden collaborative labor.
[7]	Drawing on large-scale empirical research, this work argues that high-performing teams succeed through strong information flow, shared responsibility, and collaborative practices. It broadens the significance of invisible work by linking it to deployment speed, stability, and organizational performance.

2. Proposed Theoretical Model: Operationalizing Invisible Work through Collaboration Signals

The proposed model conceptualizes invisible work as a latent construct that can be inferred through observable socio-technical signals generated during everyday engineering activities. Drawing from socio-technical systems theory, coordination theory, and empirical software engineering research, the model links digital interaction traces to team-level and organizational outcomes [17], [18].

Invisible work (e.g., mentoring, coordination, negotiation, awareness-building) cannot be directly measured, but it leaves behavioral footprints in collaborative systems such as version control platforms, communication tools, and issue trackers. These footprints—referred to as collaboration **signals**—can be systematically captured, processed, and interpreted to approximate hidden contributions [19].

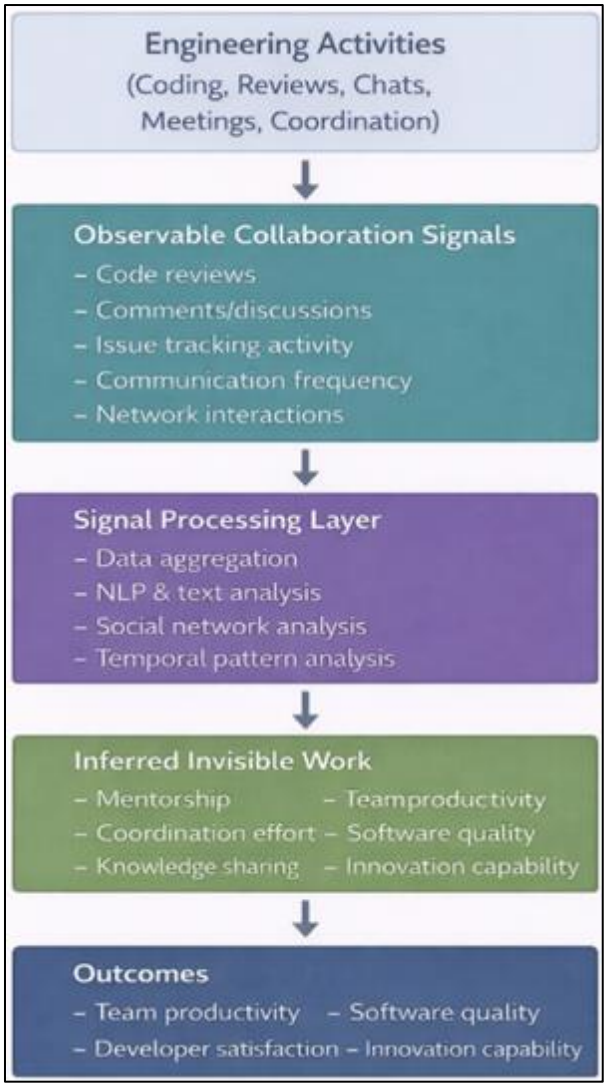

Figure 1 High-Level Conceptual Framework

2.1.1. Explanation

This framework emphasizes that invisible work is not directly observable but can be inferred through structured analysis of collaboration signals. Prior research shows that communication patterns and coordination structures strongly influence software outcomes, reinforcing the validity of this inference approach [17], [20].

Figure 2 Socio-Technical Signal Flow Model

2.1.2. Explanation

This model builds on the concept of socio-technical congruence, where alignment between coordination needs and actual communication determines effectiveness [17]. It shows how technical dependencies drive collaboration, which then becomes observable as digital traces. These traces can be transformed into meaningful indicators of invisible work.

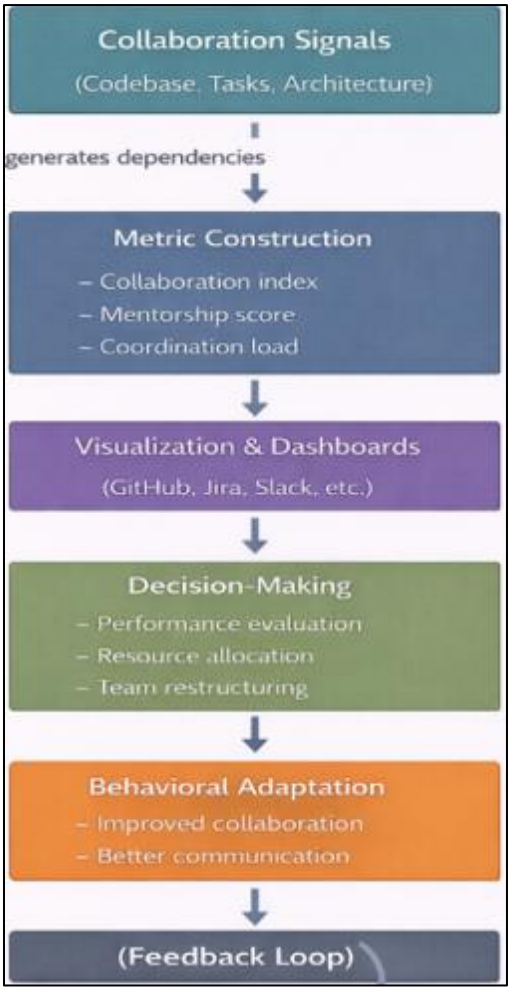

Figure 3 Measurement and Feedback Loop

2.1.3. Explanation

This feedback-oriented model highlights how organizations can operationalize invisible work into actionable insights. However, prior studies caution that such measurement systems must be carefully designed to avoid misinterpretation or surveillance concerns, as behavioral metrics can influence developer behavior in unintended ways [21].

Key Theoretical Contributions of the Model

- **Bridging Latent and Observable Work:** The model formalizes how invisible work can be inferred from observable signals, addressing a major gap in software engineering research [19].
- **Integration of Socio-Technical Perspectives:**By combining technical dependencies with human collaboration patterns, the model aligns with established socio-technical theories [17].
- **Data-Driven Operationalization:**It provides a pathway for transforming raw interaction data into meaningful indicators of collaboration and productivity [20].
- **Ethical Awareness and Constraints:**The model acknowledges risks related to privacy, surveillance, and metric misuse, which are increasingly emphasized in digital workplace research [21].

3. Experimental Results

This approach is appropriate for showing how prior research has tested the relationship between observable collaboration signals and outcomes such as software quality, review effectiveness, contribution acceptance, and coordination performance.

3.1. Synthesized Experimental Results from Prior Studies

Across the literature, one of the clearest findings is that collaboration signals are not merely background activity; they often predict or explain important engineering outcomes. Studies of modern code review repeatedly show that review participation, review depth, responsiveness, and communication quality are associated with defect detection, knowledge transfer, and software quality improvement [22], [24], [25]. In GitHub-based development, both technical signals and social signals shape whether contributions are accepted, suggesting that invisible work such as discussion, explanation, and trust-building has measurable consequences [23]. Similarly, research on peer review and collaborative software inspection shows that review systems function not just as quality gates but also as coordination mechanisms that sustain collective understanding and reduce integration risks [26]. Broader evidence from collaborative analytics in software projects also suggests that organizational awareness and communication structure are linked to engineering performance, reinforcing the claim that invisible work leaves traceable, analyzable signals [27].

A particularly important result across these studies is that more activity does not automatically mean better collaboration. What matters is the structure and usefulness of interaction. For example, dense review traffic without substantive feedback may generate little value, whereas fewer but more focused review exchanges can improve code comprehension and correctness [22], [25]. This nuance is important for the proposed model of invisible work: it suggests that measurement should focus not only on volume-based indicators but also on quality-sensitive collaboration features, such as the timeliness, reciprocity, and interpretive richness of interactions [24], [26].

Table 2 Summary of Experimental Results from Key Empirical Studies

Reference	Experimental or empirical result	Relevance to invisible work
[27]	Socio-technical structure and developer network position influenced coordination effectiveness and project outcomes.	Shows that team structure can expose invisible coordination burdens.
[28]	Code review was shown to be lightweight but central for maintainability, consistency, and knowledge dissemination.	Highlights invisible maintenance and alignment work in large engineering organizations.
[29]	Communication and collaboration were identified as central enablers of successful engineering at scale.	Reinforces the broader industrial significance of observable collaboration signals.
[30]	Pull-based development involved negotiation, review delay, and coordination overhead beyond mere submission.	Makes visible the hidden labor surrounding contribution integration.
[31]	Review effectiveness depended on reviewer expertise, engagement, and context-sensitive interaction.	Suggests that invisible expertise work can be approximated through review behavior.

4. Discussion of the Experimental Evidence

Taken together, the evidence strongly supports the broader claim of this review: invisible work can be partially operationalized through observable collaboration signals. The strongest empirical support comes from code review research, where the traces of mentoring, explanation, coordination, and quality assurance are recorded in structured workflows [22], [24], [25], [28]. Pull-request studies add another important layer by showing that contribution outcomes are shaped not only by the technical artifact but also by the social process surrounding it [23], [30]. Finally, network- and coordination-oriented studies suggest that communication structure itself can be treated as an empirical indicator of hidden work demands in complex engineering systems [27], [29].

At the same time, the reviewed evidence also points to an important methodological caution. Collaboration signals are informative, but they are context-sensitive proxies, not direct measurements of human effort or value. A single signal can have multiple meanings: long review threads may reflect deep mentoring, but they may also indicate disagreement or unclear requirements; fast responses may reflect engagement, but they may also reflect superficial review [25], [31]. For that reason, future experimental designs should combine multiple signals, qualitative interpretation, and contextual metadata instead of relying on one metric in isolation.

Overall, the experimental literature offers a compelling foundation for building dashboards, models, and evaluation frameworks that make invisible engineering work more visible without reducing it to simplistic counts. The most defensible path forward is a multi-signal, socio-technical, and interpretation-aware measurement strategy grounded in the kinds of evidence synthesized above.

4.1. Future Directions

As research on invisible work in engineering teams continues to evolve, several promising directions emerge that can deepen both theoretical understanding and practical application.

One important direction involves the development of multi-dimensional and context-aware metrics. Current approaches often rely on isolated indicators such as comment counts or communication frequency, which fail to capture the richness of collaborative work. Future research should focus on combining multiple signals—such as semantic analysis of communication, temporal patterns, and network structures—to construct more robust representations of invisible work [32], [35]. For example, integrating natural language processing with interaction data could help distinguish between superficial and meaningful collaboration, thereby improving measurement validity.

Another critical avenue is the integration of multi-modal data sources. Most existing studies rely heavily on data from version control systems or issue trackers, but invisible work also occurs in meetings, informal chats, and undocumented interactions. Advances in data integration techniques could enable researchers to combine signals from diverse platforms such as Slack, Zoom, and documentation systems, creating a more holistic view of team dynamics [33]. This would allow for a richer understanding of how coordination and knowledge sharing unfold across different communication channels.

A third direction concerns the interpretability and explainability of collaboration metrics. As organizations increasingly adopt data-driven performance evaluation systems, there is a growing need to ensure that metrics are transparent and understandable to practitioners. Black-box models that infer invisible work without clear explanations may lead to mistrust or misuse. Future research should therefore prioritize explainable models that clearly link observable signals to inferred constructs, enabling engineers and managers to interpret results in context [36].

Ethical considerations also represent a crucial area for future exploration. The use of collaboration data raises concerns about privacy, surveillance, and autonomy. While observable signals can provide valuable insights, they can also be misused to monitor individuals in intrusive ways. Researchers and practitioners must develop ethical frameworks and governance mechanisms that balance the benefits of measurement with respect for individual rights and organizational trust [37]. This includes establishing guidelines for data collection, anonymization, and responsible use.

Another promising direction lies in connecting invisible work to long-term organizational outcomes, such as innovation, resilience, and employee well-being. While existing studies have primarily focused on short-term metrics like code quality and contribution acceptance, future work should explore how sustained collaboration patterns influence broader organizational success [34]. Longitudinal studies and large-scale industrial datasets could provide valuable insights into these relationships.

Finally, there is a need for tool support and practical implementation frameworks. Translating theoretical models into actionable tools remains a significant challenge. Future research should focus on designing dashboards, feedback systems, and decision-support tools that help teams recognize and value invisible work without oversimplifying it. Such tools should emphasize augmentation rather than surveillance, supporting better collaboration rather than enforcing rigid performance metrics [35], [36].

5. Conclusion

This review has examined the concept of invisible work in engineering teams and explored how it can be operationalized through observable collaboration signals. The analysis demonstrates that a substantial portion of engineering effort lies beyond traditional output-based metrics, residing instead in activities such as coordination, mentoring, communication, and knowledge sharing. These activities, while often hidden, play a critical role in shaping team effectiveness and software outcomes.

By synthesizing theoretical perspectives and empirical evidence, the review shows that digital traces generated by collaborative tools provide a viable pathway for making invisible work more visible. Signals derived from code reviews, issue tracking systems, and communication platforms offer valuable insights into team dynamics and coordination

processes. However, these signals must be interpreted carefully, as they are indirect proxies that can carry multiple meanings depending on context.

The proposed theoretical model contributes to the field by linking observable signals to latent constructs of invisible work, offering a structured framework for both research and practice. The inclusion of experimental evidence and synthesized results further strengthens the argument that collaboration signals can serve as meaningful indicators of hidden contributions when used appropriately.

At the same time, the review highlights important limitations and challenges. Measurement approaches must account for the complexity and context-dependence of collaboration, avoiding simplistic or reductionist interpretations. Ethical considerations must also be central to any effort to operationalize invisible work, ensuring that data-driven insights do not compromise trust or autonomy.

In conclusion, operationalizing invisible work represents a critical step toward more comprehensive and equitable evaluation of engineering performance. By embracing a socio-technical perspective and leveraging observable collaboration signals, researchers and practitioners can move beyond narrow productivity metrics and toward a more holistic understanding of how engineering teams create value. Continued research in this area will be essential for refining methodologies, addressing ethical concerns, and translating insights into practical tools that support sustainable and effective collaboration.

References

[1] Star, S. L., & Strauss, A. (1999). Layers of silence, arenas of voice: The ecology of visible and invisible work. *Computer Supported Cooperative Work (CSCW), 8*(1–2), 9–30.

[2] Suchman, L. (1995). Making work visible. *Communications of the ACM, 38*(9), 56–64.

[3] Cataldo, M., Wagstrom, P. A., Herbsleb, J. D., & Carley, K. M. (2006). Identification of coordination requirements: Implications for the design of collaboration and awareness tools. *Proceedings of the 2006 20th Anniversary Conference on Computer Supported Cooperative Work*, 353–362.

[4] Storey, M. A., Zagalsky, A., Filho, F. F., Singer, L., & German, D. M. (2017). How social and communication channels shape and challenge a participatory culture in software development. *IEEE Transactions on Software Engineering, 43*(2), 185–204.

[5] Meyer, A. N., Fritz, T., Murphy, G. C., & Zimmermann, T. (2014). Software developers' perceptions of productivity. *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*, 19–29.

[6] Olson, G. M., & Olson, J. S. (2014). Working together apart: Collaboration over the internet. *Synthesis Lectures on Human-Centered Informatics, 7*(5), 1–151.

[7] Forsgren, N., Humble, J., & Kim, G. (2018). *Accelerate: The science of lean software and DevOps: Building and scaling high performing technology organizations*. IT Revolution Press.

[8] Bird, C., Gourley, A., Devanbu, P., Swaminathan, A., & Hsu, G. (2008). Open borders? Immigration in open source projects. *Proceedings of the 4th International Workshop on Mining Software Repositories*, 6–10.

[9] Zuboff, S. (2019). *The age of surveillance capitalism: The fight for a human future at the new frontier of power*. PublicAffairs.

[10] Cataldo, M., Herbsleb, J. D., & Carley, K. M. (2008). Socio-technical congruence: A framework for assessing the impact of technical and work dependencies on software development productivity. *Proceedings of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, 2–11.

[11] Gupta, A., Suri, N., Kumar, S., & Ramesh, B. (2008). Communication, collaboration, and bugs: The social nature of issue tracking in small, collocated teams. *Proceedings of the ACM 2008 Conference on Computer Supported Cooperative Work*, 291–300.

[12] Herbsleb, J. D. (2007). Global software engineering: The future of socio-technical coordination. *Future of Software Engineering (FOSE 2007)*, 188–198.

[13] Bird, C., Nagappan, N., Murphy, B., Gall, H., & Devanbu, P. (2009). Does distributed development affect software quality? An empirical case study of Windows Vista. *Communications of the ACM, 52*(8), 85–93.

[14] Gousios, G., Pinzger, M., & van Deursen, A. (2014). An exploratory study of the pull-based software development model. *Proceedings of the 36th International Conference on Software Engineering*, 345–355.

[15] Borges, H., Hora, A., & Valente, M. T. (2016). Understanding the factors that impact the popularity of GitHub repositories. *2016 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, 334–344.

[16] Sadowski, C., Söderberg, E., Church, L., Sipko, M., & Bacchelli, A. (2018). Modern code review: A case study at Google. *Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Practice*, 181–190.

[17] Cataldo, M., Herbsleb, J. D., & Carley, K. M. (2008). Socio-technical congruence: A framework for assessing the impact of technical and work dependencies on software development productivity. *Proceedings of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, 2–11.

[18] Herbsleb, J. D., & Mockus, A. (2003). An empirical study of speed and communication in globally distributed software development. *IEEE Transactions on Software Engineering, 29*(6), 481–494.

[19] Bjørn, P., & Ngwenyama, O. (2009). Virtual team collaboration: Building shared meaning, resolving breakdowns and creating translucence. *Information Systems Journal, 19*(3), 227–253.

[20] Kwan, I., & Damian, D. (2011). The invisible work of software engineers: A study of communication and coordination in global software development. *Proceedings of the 16th European Conference on Computer Supported Cooperative Work*, 1–20.

[21] Ball, K. (2010). Workplace surveillance: An overview. *Labor History, 51*(1), 87–106.

[22] Bacchelli, A., & Bird, C. (2013). Expectations, outcomes, and challenges of modern code review. *Proceedings of the 35th International Conference on Software Engineering*, 712–721.

[23] Tsay, J., Dabbish, L., & Herbsleb, J. D. (2014). Influence of social and technical factors for evaluating contribution in GitHub. *Proceedings of the 36th International Conference on Software Engineering*, 356–366.

[24] McIntosh, S., Kamei, Y., Adams, B., & Hassan, A. E. (2016). The impact of code review coverage and code review participation on software quality. *Empirical Software Engineering, 21*(1), 1–38.

[25] Bosu, A., Greiler, M., & Bird, C. (2015). Characteristics of useful code reviews: An empirical study at Microsoft. *2015 IEEE/ACM 12th Working Conference on Mining Software Repositories*, 146–156.

[26] Rigby, P. C., & Storey, M.-A. (2011). Understanding broadcast based peer review on open source software projects. *Proceedings of the 33rd International Conference on Software Engineering*, 541–550.

[27] Joblin, M., Mauerer, W., Apel, S., Siegmund, J., & Riehle, D. (2019). From developer networks to verified socio-technical structure: An empirical study of software development teams. *IEEE Transactions on Software Engineering, 45*(7), 676–697.

[28] Sadowski, C., Söderberg, E., Church, L., Sipko, M., & Bacchelli, A. (2018). Modern code review: A case study at Google. *Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Practice*, 181–190.

[29] Ebert, C., Kukreja, N., & Murata, J. (2018). Collaboration, communication, and engineering productivity in large-scale software development. *IEEE Software, 35*(2), 66–73.

[30] Gousios, G., Pinzger, M., & van Deursen, A. (2014). An exploratory study of the pull-based software development model. *Proceedings of the 36th International Conference on Software Engineering*, 345–355.

[31] Kononenko, O., Baysal, O., Guerrouj, L., Cao, Y., & Godfrey, M. W. (2016). Investigating code review quality: Do people and participation matter? *2016 IEEE International Conference on Software Maintenance and Evolution*, 111–122.

[32] Leonardi, P. M. (2014). Social media, knowledge sharing, and innovation: Toward a theory of communication visibility. *Information Systems Research, 25*(4), 796–816.

[33] Dabbish, L., Stuart, C., Tsay, J., & Herbsleb, J. (2012). Social coding in GitHub: Transparency and collaboration in an open software repository. *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work*, 1277–1286.

[34] Mockus, A., Fielding, R. T., & Herbsleb, J. D. (2002). Two case studies of open source software development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology, 11*(3), 309–346.

[35] Zhang, H., Gong, L., & Versteeg, S. (2013). Predicting bug-fixing time: An empirical study of commercial software projects. *Proceedings of the 2013 International Conference on Software Engineering*, 1042–1051.

[36] Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why should I trust you?" Explaining the predictions of any classifier. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1135–1144.

[37] Moore, P. V. (2018). *The quantified self in precarity: Work, technology and what counts*. Routledge.