Remote and Hybrid Work in Software Engineering: Productivity, Collaboration, Wellbeing and Organisational Design

A critical synthesis of post-pandemic evidence (2023–November 2026)

E.Oghina

Abstract

Remote and hybrid work have become enduring features of software engineering, yet evidence on their consequences is difficult to interpret because work arrangements, research designs and performance measures differ substantially. This review synthesises selected academic and grey literature, prioritising publications from 2023 to November 2026 and using earlier pandemic studies as historical context. Four questions examine productivity, communication and collaboration, developer wellbeing, and management and organisational outcomes. The synthesis distinguishes self-reported performance from behavioural indicators and causal estimates, as well as individual outcomes from team and organisational outcomes. Evidence from a randomised hybrid-work trial supports improved retention and job satisfaction without detectable deterioration in measured performance in its particular setting. Software-engineering studies indicate that flexibility can support focused individual work, while distributed coordination, informal learning and feedback remain vulnerable to reduced shared context. Evidence on proximity and mentoring also challenges the assumption that short-term individual output adequately captures engineering effectiveness. A socio-technical framework is proposed in which task interdependence, coordination practices, access to mentorship and boundary management condition the relationship between work arrangements and sustainable performance. The evidence supports context-sensitive policy design rather than a universal preference for either remote or office-based work. The review identifies priorities for longitudinal evaluation, equitable career development and the interaction between generative AI and distributed work. Because the original search records were not provided for audit, findings are presented as a structured critical synthesis rather than a comprehensive systematic review.

Keywords: software engineering; hybrid work; remote work; developer productivity; collaboration; wellbeing; socio-technical systems; literature review.

1. Introduction

The COVID-19 pandemic accelerated an established transition towards digitally mediated software development. Distributed version control, cloud infrastructure, issue tracking and asynchronous communication made many engineering activities technically feasible outside a shared office. However, the emergency transition was not equivalent to an intentionally designed remote or hybrid operating model. Early research documented heterogeneous experiences among software developers working from home, while subsequent studies identified continuing demand for flexibility and the need to redesign workplace policies [1, 2, 3].

The contemporary policy question is no longer simply whether software can be developed remotely. Rather, it concerns which arrangements sustain quality, learning, collaboration and retention over time, for which employees, and under which organisational conditions. A developer may complete more implementation work at home while a team experiences longer review queues, slower knowledge transfer or reduced opportunities for junior staff. Conversely, a co-located team may

benefit from rapid informal clarification but sacrifice uninterrupted work through commuting and frequent interruptions. These outcomes can coexist; treating either as decisive evidence of an optimal work mode obscures the underlying trade-offs.

Existing reviews provide a foundation for this question. The 2024 mapping study by Khanna et al. examined the intersection of hybrid work and agile software development [4]. A subsequent systematic review, available online in November 2026, synthesised 42 studies across 48 organisational cases; benefits were most apparent at individual level, whereas team challenges dominated reported disadvantages [5]. The latter finding is particularly important: an individual advantage does not necessarily establish a collective advantage. The present paper complements that literature through a critical synthesis spanning software productivity, collaboration, wellbeing and organisational design, with explicit attention to causal credibility and the distribution of benefits and costs across organisational levels. It does not claim a novel comprehensive search or independent replication of those reviews.

1.1. Scope and definitions

Remote work refers here to work performed away from a common organisational office, typically at home. Fully remote arrangements involve routine geographical separation; hybrid work combines remote and on-site work, whether through scheduled office days, team-selected events or individual choice. Remote-first describes an operating principle in which processes and information access are designed for distributed participation, irrespective of occasional office attendance. These categories are analytically distinct: a team attending an office together twice weekly differs from a nominally hybrid team whose members attend on unrelated days.

Productivity is defined as the ability to produce valuable, high-quality software outcomes relative to the resources and time required. It is not synonymous with activity volume. The SPACE framework distinguishes satisfaction and wellbeing, performance, activity, communication and collaboration, and efficiency and flow [6]. Communication denotes information exchange; collaboration denotes joint work towards shared outcomes, including coordination and knowledge creation. Wellbeing includes perceived work–life balance, psychological and social functioning, and the sustainability of work demands. Psychological safety is narrower: it concerns whether team members can raise concerns, ask questions and acknowledge mistakes without fear of interpersonal penalty. These constructs are related but should not be treated as interchangeable.

1.2. Research questions

RQ1. How do remote and hybrid work arrangements affect perceived and measured software engineering productivity, and under what conditions do outcomes differ?

RQ2. How do these arrangements influence communication, collaboration, knowledge sharing and mentoring in software development teams?

RQ3. How do remote and hybrid arrangements influence developer wellbeing and work–life balance, including differences across employee circumstances and experience levels?

RQ4. How do management practices, tooling and organisational culture condition the effectiveness and sustainability of remote and hybrid software engineering?

2. Review approach

2.1. Design and scope

The original review was developed using a multivocal literature review (MLR) approach, informed by Garousi et al., which allows research evidence to be considered alongside relevant practitioner literature [7]. The original manuscript reports iterative searches in Google Scholar, IEEE Xplore and

thesis repositories, followed by title and abstract screening, duplicate removal, full-text screening and Zotero-based reference management. The reported example search strings were: “software development” AND “hybrid work” AND “communication”; “remote software teams” AND “productivity”; and “agile practices” AND “distributed teams”. The stated inclusion criteria concerned English-language sources, relevance to remote or hybrid software engineering, and empirical evidence or rigorous conceptual contributions. Methodologically opaque and predominantly opinion-based material was intended to be excluded.

The original manuscript does not provide dated search logs, complete search strings per database, screening counts, exclusion reasons, or a study-level extraction and quality-assessment record. Consequently, the completeness of retrieval and consistency of screening cannot be established. This revised review is therefore a structured narrative synthesis informed by MLR principles. It retains relevant original literature and incorporates selected additional sources identified through targeted citation checks and searches for directly relevant research, up to 24 November 2026. It is not presented as a newly executed systematic search or as a formal update of the 2024 mapping study. Earlier publications are used explicitly as historical or conceptual evidence; publication year is distinguished from the period in which observations were made. For example, a study published in 2024 using data collected in 2021 is not evidence of a 2024 workplace policy.

2.2. Evidence interpretation and synthesis

Evidence is organised by the four research questions rather than pooled into a single effect estimate. Three distinctions guide interpretation: (i) relevance to professional software engineering versus adjacent knowledge-work settings; (ii) whether evidence concerns forced remote work, deliberate hybrid arrangements or fully remote operating models; and (iii) the strength and type of inference supported by the research design. Randomised trials provide stronger evidence about a specified intervention and measured outcomes; quasi-experimental designs may support causal interpretation subject to identifying assumptions; surveys and case studies are useful for identifying experiences and mechanisms but generally cannot isolate the effect of work location. Reviews provide coverage of existing studies, but may overlap with primary papers cited here and are not counted as independent replications. Selected grey literature is retained only where it contributes otherwise relevant context or a clearly identified research question, not as the basis for general effect sizes.

No numerical quality scores or new pooled estimates are reported, because the underlying study-level audit data were not provided. The selected evidence in Table 1 illustrates the differences between study populations, methods and inferences; it is not an inventory of all retrieved or included records. The synthesis uses cautious language where evidence is observational, self-reported, limited to a single firm, or indirect. The conceptual framework in Section 7 is an interpretive contribution derived from the literature, not a statistically validated model.

3. Productivity and performance (RQ1)

3.1. Distinguishing perceived from measured performance

Research conducted during the pandemic established an important baseline: remote work did not produce a uniform change in developers’ perceived productivity. Šmite et al. synthesised 7,686 survey data points and found little average change alongside substantial differences between individuals [8]. The result should not be interpreted as a measured improvement or as proof of equivalence; it concerns respondents’ experiences during an exceptional period. Similarly, a 2023 study of programming teams reported favourable assessments of remote performance [9]. Self-selection into preferred work arrangements, recall bias and the possibility that participants equate longer hours with greater output limit the conclusions that can be drawn from such assessments.

Stronger causal evidence is available for one specific hybrid policy. Bloom et al. randomly assigned 1,612 graduate employees at a Chinese technology company to either two home-working days per week or full-time office attendance. Over the six-month experiment, attrition was 4.8% in the hybrid group and 7.2% in the office group, an absolute difference of 2.4 percentage points and a relative reduction of one-third ($p = 0.043$). The study also found no detectable adverse effect on the performance measures examined [10]. This supports the feasibility of that hybrid arrangement; it does not establish that fully remote work is equivalent, that all engineering roles benefit equally, or that every dimension of quality and innovation remains unchanged. Participants included engineers as well as employees in other business functions, and the experiment preceded the post-pandemic period despite being published in 2024.

An industrial case study of post-pandemic hybrid work illustrates the importance of individual preferences, commuting, project demands and team dynamics [11]. A broader 2026 online systematic review similarly identified perceived productivity advantages primarily at the individual level, while finding that the available literature did not establish corresponding team-level advantages [5]. Together, these observations caution against extrapolating individual satisfaction or concentration directly to collective delivery performance.

3.2. Focus, flow and the productivity denominator

Software development requires sustained cognitive effort, but it also depends on frequent collaboration with others. Research on developer flow and interruptions highlights the importance of task context and the costs of disruption [12, 13]. Remote work may protect implementation, debugging and design tasks from unplanned office interruptions. It can also introduce distractions from domestic responsibilities, asynchronous messages and technical constraints. The relevant mechanism is not home working itself but the extent to which the environment enables attention appropriate to the task.

A second distinction concerns hours worked. A developer who completes additional tasks by working longer evenings has increased output but not necessarily efficiency. Research on hybrid working explicitly identifies the distinction between additional work hours and changes in how working time is allocated [14]. The removal of commuting may improve available personal time, increase work time, or produce both effects. Analyses should therefore report workload and hours alongside output and quality, rather than interpreting extended availability as evidence of higher productivity.

Repository activity provides behavioural evidence but remains an imperfect proxy. Commit counts and timing are affected by branching strategies, squashing, release cycles, team conventions and the type of work undertaken. A commit is neither a standard unit of value nor a complete record of engineering contribution. Reviews, incident response, mentoring and architectural decisions may have greater downstream value than a high volume of code changes. A defensible evaluation combines activity indicators with quality, delivery and collaboration measures and interprets them at appropriate levels of aggregation [6].

3.3. Coordination costs and task interdependence

The productivity effect of work location is likely to differ between activities that can be completed independently and those requiring frequent feedback or resolution of dependencies. A global cross-sectional survey of 297 respondents on pandemic working-from-home experiences, published in 2024, identified mixed effects on software activities and challenges in coordination, communication and project planning [15]. Although these results reflect 2021 conditions, they illustrate why perceived individual gains can coexist with delays at project level.

The implications are especially relevant for large or tightly coupled codebases. A developer may progress rapidly on a task but still wait for a design decision, an environment fix or a reviewer. Such waiting may not appear in an individual's activity metrics, yet it lengthens delivery lead time. Hybrid

policies can reduce these bottlenecks only if the relevant collaborators have compatible schedules and access to shared information. Attendance on arbitrary days is therefore a weak substitute for intentionally coordinated access to expertise.

Answer to RQ1. The evidence supports the possibility of stable or improved individual performance under well-designed remote or hybrid arrangements, but does not establish universal productivity gains. Team-level performance depends on coordination costs, task interdependence, work hours and the quality of feedback. Evidence is stronger for a specific two-day hybrid policy than for unrestricted fully remote arrangements.

4. Communication, collaboration and learning (RQ2)

4.1. Information exchange versus joint problem-solving

Digital tools reduce the need for physical presence when exchanging routine information. Written updates, issue trackers, design documents and asynchronous code review can make decisions searchable, improve transparency and accommodate different working schedules. However, transmitting information is not equivalent to establishing shared understanding or coordinating action. Ambiguous requirements, complex architectural decisions and unresolved disagreements often require rapid iteration and richer communication.

A large study of Microsoft information workers during the pandemic found that firm-wide remote work was associated with more siloed collaboration networks and a shift towards asynchronous communication; the share of collaboration time devoted to cross-group connections fell by approximately one-quarter relative to the pre-pandemic level [16]. The outcome describes a change in network structure, not a 25% decline in productivity or collaboration quality. It is also evidence from a historical firm-wide remote transition across information workers, rather than a direct estimate of modern hybrid software-team performance.

Research on remote-first and hybrid software teams identifies coordination as a process that must be actively designed [17]. Studies of fully remote co-creation also show that teams can jointly produce shared digital artefacts through deliberately organised activities and tools [18]. This suggests that geographical separation is not an absolute barrier to collaboration. Rather, effective distributed work requires mechanisms for exposing context, resolving ambiguity, distributing knowledge and maintaining common ownership.

4.2. Meeting design and the role of co-presence

Agile practices offer recurring opportunities for coordination, but the existence of a meeting is not evidence that coordination is effective. A systematic review of meetings involving remote software teams identified benefits alongside technical and social challenges [19]. Excessive synchronous communication can reduce time available for concentrated development; insufficient synchronous contact can leave decisions unresolved or poorly understood. The appropriate balance depends on the purpose of interaction.

Christensen et al.'s 2025 study of recurring agile meetings distinguishes information sharing from activities involving active discussion, brainstorming and relationship building [20]. The findings support choosing meeting formats according to intent rather than prescribing a uniform mode. Routine updates may be written or virtual; complex design workshops and retrospectives may benefit from richer interaction, which can include co-location where feasible. A hybrid meeting must also account for unequal access to side conversations, whiteboards and non-verbal cues. Making decisions visible to all participants is more important than merely providing a video link.

4.3. Mentorship, onboarding and knowledge continuity

Software engineering is cumulative knowledge work. Understanding a system often requires access to tacit architectural knowledge, domain assumptions and local conventions not fully represented in code or documentation. Pair programming and collaborative review provide opportunities to transmit that knowledge. Research on pair programming in hybrid settings shows that its implementation depends on the circumstances of collaboration, rather than simply on whether suitable tools are available [21].

More direct evidence of the trade-off between individual output and learning comes from Emanuel et al., whose 2026 study examined software engineers at a large US firm between 2019 and 2024. Exploiting changes in proximity around office closures and return mandates, the authors estimated that sitting near teammates increased coding feedback by 18.3% and improved code quality; the benefits and mentoring costs differed across workers [22]. This is quasi-experimental evidence from one firm, not a randomised comparison of all possible hybrid schedules. Nevertheless, it supports the concern that short-term individual production may omit the longer-term value of giving and receiving feedback.

These findings have implications for junior developers and employees joining unfamiliar projects. A remote-first team can respond through explicit mentorship assignments, scheduled pairing, accessible architecture documentation and predictable reviewer availability. The effectiveness of such interventions relative to spontaneous in-person learning remains an empirical question. The appropriate policy should be evaluated through time to meaningful contribution, feedback quality, learning progress and retention, not solely through the number of completed tasks.

> **Answer to RQ2.** Remote and hybrid work can sustain communication and purposeful collaboration, but information access alone does not guarantee knowledge transfer or shared understanding. Structured documentation, meeting design and feedback practices are particularly important where work depends on tacit knowledge, cross-team relationships and mentoring.

5. Developer wellbeing and work–life balance (RQ3)

5.1. Autonomy, boundaries and social connection

Flexible work can improve perceived work–life balance through reduced commuting and greater control over the timing and environment of work. Research on hybrid software professionals identifies wellbeing as a balance between individual circumstances, preferences and work demands [23, 11]. These advantages are not uniform. Home working may expose developers to isolation, inadequate workspaces and competing responsibilities, while office attendance may introduce commuting burdens and reduce scheduling autonomy. Hybrid work is therefore better understood as a configurable set of resources and demands than as an inherently beneficial intermediate arrangement.

Longitudinal evidence also highlights the need to distinguish wellbeing from immediate productivity. Russo et al. followed developers across six measurement points between April 2020 and April 2022. Wellbeing and social contact quality improved as restrictions relaxed, while productivity remained broadly stable [24]. The study does not isolate a hybrid-policy effect, but demonstrates that these outcomes can move differently. The absence of a productivity decline is consequently insufficient evidence that a work arrangement is sustainable or that wellbeing needs are being met.

The general literature on software-engineer wellbeing is multidimensional. Godliauskas and Šmite reviewed 44 quantitative survey studies and developed a theory relating predictors and outcomes of wellbeing [25]. Their framework provides conceptual support for considering professional and social conditions together. It does not establish that remote or hybrid work independently causes higher

wellbeing, nor that a change in wellbeing automatically produces a measurable gain in software quality.

5.2. Psychological safety and sustainable quality

Psychological safety enables developers to ask for assistance, surface defects, challenge assumptions and acknowledge uncertainty. Studies of agile software teams identify interpersonal and leadership conditions that help establish psychological safety, and describe how it supports behaviours directed towards software quality [26, 27]. Research on remote software development further examines the ways in which distributed interactions can change the expression of these behaviours [28].

The mechanism is important but should not be overstated. Evidence linking psychological safety to quality-oriented behaviours does not by itself quantify a causal path from hybrid schedules through wellbeing to defect rates. A plausible hypothesis is that an environment in which developers can seek help and discuss mistakes reduces the likelihood that hidden problems persist. Testing this pathway would require measurements of team climate, interactions and subsequent quality outcomes over time.

Similarly, research on post-pandemic software-team resilience suggests that continuity depends on more than immediate task completion [29]. A team with narrow ownership of critical components may appear productive while remaining exposed to staff departures, absences or incidents. Knowledge distribution, inclusive communication and sustainable workloads are therefore relevant to resilience as well as developer experience.

5.3. Heterogeneous effects and equity

The consequences of work arrangements depend on career stage, accessibility requirements, household circumstances, commute length, work preferences and task responsibilities. Research involving software professionals from underrepresented groups highlights the need to consider differentiated experiences of hybrid work rather than assuming that one policy serves all employees equally [30]. The proximity study additionally indicates that feedback and mentoring opportunities may be unequally affected by physical separation [22]. These observations warrant evaluation of both access to flexibility and access to professional development.

Junior developers may need more frequent feedback on unfamiliar tasks; senior developers may incur a substantial mentoring load. Neither group should be treated as homogeneous. Remote work may also remove some barriers to participation while introducing others, including exclusion from informal decision-making. A fair policy should therefore preserve reasonable flexibility while ensuring consistent access to information, mentorship and advancement. Outcomes should be disaggregated where sample sizes and privacy constraints permit, avoiding simplistic age or demographic stereotypes.

> **Answer to RQ3.** Flexible arrangements can improve wellbeing and work–life balance, but benefits depend on autonomy, workable boundaries, social support and equitable access to professional development. Existing evidence supports plausible links between wellbeing, psychological safety and sustainable team functioning; it does not establish a universal causal effect on software quality.

6. Management, tooling and organisational outcomes (RQ4)

6.1. Leadership, coordination and fairness

Distributed work shifts the management challenge from observing presence towards establishing shared objectives, transparent dependencies and meaningful feedback. Research on hybrid software practices emphasises the importance of aligning work arrangements with the work itself [31, 11, 5].

The practical implication is that managers should evaluate agreed outcomes and barriers to delivery rather than relying on office attendance, online status or activity counts as substitutes for contribution.

Proximity bias is a plausible risk where managers have more informal contact with co-located employees. The risk is not confined to remote arrangements: it can arise in any team with unequal access to decision-makers. Mitigation requires explicit review criteria, consistent feedback opportunities and transparent documentation of decisions. Whether a given intervention reduces bias should be evaluated with relevant outcomes, including access to assignments and progression, rather than assumed from policy design alone.

A comparative case study reported in a 2024 master's thesis describes different approaches in two companies: fixed on-site days in one and flexible event-based attendance in the other [32]. This grey-literature evidence illustrates organisational variation, rather than demonstrating the superiority of either policy. Team-driven schedule alignment may be useful when coordination is intensive, but identical attendance requirements can impose unnecessary costs on independent work. Managers should establish which activities require shared time, how exceptions are handled, and how remote participants can access the same context. This is a shift from managing location to managing interdependence, without assuming that physical presence is irrelevant.

6.2. Tooling as infrastructure, not a substitute for process

Version control, issue management, continuous integration and communication platforms form the technical infrastructure of distributed development. Their contribution depends on how teams integrate them into everyday work. Co-creation research illustrates the role of shared artefacts in fully remote collaboration [18]. Research on social debt and technical debt in large-scale agile projects further motivates attention to the consequences of neglected relationships and coordination [33].

Fragmentation becomes problematic when decisions, requirements and implementation details are distributed across disconnected channels. A team may possess several capable tools while still lacking a reliable record of why a decision was made. Useful controls include a canonical location for work items, links between issues and code changes, durable records of decisions, clear ownership and norms for urgent versus non-urgent messages. These are recommendations derived from the synthesis; no universal percentage of time lost to application switching is inferred.

Distributed access also introduces security and operational requirements, including reliable identity management, appropriate access control, reproducible development environments and secure dependency management. These risks are not uniquely caused by remote work, and the evidence reviewed does not isolate their relative prevalence across work modes. Their significance lies in the need for consistent engineering infrastructure that permits distributed contribution without sacrificing governance or quality.

6.3. Retention, cost and organisational sustainability

The strongest retention result discussed in this review comes from the randomised hybrid-work study [10]. A reduction in employee departures can preserve institutional knowledge and reduce recruitment disruption, although the magnitude of any cost saving depends on role, labour market and organisational practice. It should not be converted into a universal monetary estimate without appropriate cost data. Similarly, reduced office expenditure is not automatic where organisations retain existing leases, maintain hybrid facilities or fund home-working infrastructure.

A sustainable policy must consider both workforce outcomes and engineering outcomes. Improved retention can coexist with slower onboarding; better individual focus can coexist with longer coordination delays. Conversely, co-location can support learning while imposing commuting and

flexibility costs. The decision is not whether one arrangement is inherently more economical, but whether its combined costs and benefits are acceptable in the organisational context.

> **Answer to RQ4.** Effective remote and hybrid work requires deliberate leadership, accessible information, coherent tooling and equitable performance evaluation. Evidence supports retention benefits in a particular hybrid intervention, while broader claims about financial savings, culture and long-term performance remain context-dependent.

7. Integrated discussion

7.1. A conditional socio-technical framework

The four research questions converge on a central proposition: the effects of work arrangements depend on how work location interacts with the social and technical organisation of software development. The model proposed here distinguishes policy inputs, mediating processes, contextual moderators and outcomes. It is a conceptual synthesis, not a validated causal model. Table 2 presents operational implications and potential measures.

First, policy inputs include remote-work frequency, schedule autonomy, alignment of office days and availability of communication and development infrastructure. Second, candidate mediating processes include uninterrupted focus, coordination latency, feedback access, social connection and work–life boundary management. Third, contextual moderators include task interdependence, team maturity, geographical dispersion, career stage and individual circumstances. Finally, outcomes span individual efficiency and wellbeing, team delivery and quality, and organisational retention and resilience. A change in one outcome need not imply improvement in the others.

This framework leads to four propositions for future empirical testing. P1: protected focus time is more likely to improve individual efficiency when tasks are relatively independent and interruptions are controlled. P2: the relationship between geographical separation and team delivery is conditioned by coordination practices and dependency structure. P3: the effect of proximity on long-term capability depends partly on access to feedback and mentorship, particularly for developers acquiring unfamiliar knowledge. P4: flexibility is more likely to support sustainable retention when accompanied by manageable workload and equitable career opportunities. These propositions are intended to clarify mechanisms and generate testable hypotheses, not to claim independently established causal relationships.

7.2. Evidence strength and cross-level trade-offs

Table 1. Selected studies and the scope of inference.

Study and design	Evidence / outcome	Interpretive boundary
Bloom et al. (2024) [10] Randomised trial; 1,612 employees	Two home days per week; attrition 4.8% versus 7.2%; no detected performance loss.	One Chinese technology firm; mixed roles; specified hybrid policy.
Emanuel et al. (2026) [22] Quasi-experimental study	Proximity associated with more coding feedback and higher code quality.	One US firm; identifying assumptions; not a universal hybrid-policy estimate.
Šmite et al. (2022) [8] Survey synthesis	7,686 data points; stable mean perceived productivity, heterogeneous experiences.	Pandemic conditions; perception rather than measured delivery.
Nguyen-Duc et al. (2024) [15] Cross-sectional survey	297 responses; mixed effects and coordination/planning difficulties.	Data collected in 2021; non-experimental and self-reported.
Russo et al. (2024) [24] Longitudinal study	Wellbeing and social contact improved while productivity remained stable.	2020–2022; concurrent changes in pandemic restrictions.
Christensen et al. (2027; online 2026) [5] Systematic review	42 studies, 48 cases; individual advantages and team-level challenges.	Secondary synthesis; possible overlap with cited primary studies.

The studies in Table 1 demonstrate why the results cannot be expressed as a single effect of remote work. The interventions and populations differ, as do the outcome measures and observation periods. The randomised evidence addresses a specific hybrid schedule, the proximity study addresses feedback and quality in a particular organisational environment, and much software-specific evidence concerns experiences and practices. Synthesising these designs is useful for explaining plausible mechanisms, but pooling them or treating all studies as equally causal would be misleading.

The most important cross-level trade-off is between immediate individual efficiency and collective capability. A senior developer working uninterrupted may complete more implementation tasks but spend less time helping colleagues; the benefit may be offset by reduced learning or slower resolution of dependencies. A well-designed remote-first team might compensate by allocating explicit mentoring time, whereas a poorly designed office-first team might experience similar knowledge barriers. Accordingly, neither individual activity nor office occupancy alone provides a defensible criterion for engineering effectiveness.

7.3. Implications for practice

Table 2. Proposed operationalisation of the conceptual framework.

Mechanism / risk	Candidate intervention	Illustrative evaluation measures
Focus and workload	Protected focus blocks; explicit limits on after-hours expectations.	Task cycle time, work hours, interruption burden, perceived flow.
Coordination latency	Shared decision records; overlap windows; meetings matched to intent.	Time blocked, review waiting time, lead time, cross-team dependency delays.
Learning and feedback	Named mentors; planned pairing; explicit review and onboarding practices.	Time to meaningful contribution, feedback access and quality, learning outcomes.
Wellbeing and inclusion	Flexible arrangements; accessible tools; equal access to decisions and feedback.	Work–life balance, psychological safety, voluntary turnover, progression access.
Quality and resilience	Shared ownership, reproducible environments, integrated issue and code workflows.	Escaped defects, rework, recovery time, ownership concentration.

These measures should be interpreted together and predominantly at team level, with privacy safeguards and attention to task complexity. They are not proposed as individual surveillance metrics or validated thresholds. Organisations can use a baseline and a defined evaluation period to assess policy changes, accounting for product releases, staffing changes and seasonal factors. Where feasible, staggered or experimental policy changes provide stronger evidence than before-and-after comparisons without a control group.

8. Limitations and threats to validity

Search and selection validity. The original documentation does not support independent reconstruction of the complete search and screening workflow. Targeted updating and removal of weaker sources improve the traceability of particular claims, but do not guarantee that all relevant studies or contrary results have been retrieved. English-language restrictions and database coverage may introduce language and publication bias. Grey literature is particularly susceptible to selection and reporting bias. No independent second reviewer or formal inter-rater agreement is documented.

Construct validity. Work mode, productivity, collaboration and wellbeing are defined inconsistently across studies. Self-reported productivity, repository activity, manager ratings and delivery outcomes measure different constructs. Similarly, a fixed hybrid schedule is not equivalent to employee-selected work location or fully remote work. The synthesis does not assume that these measures or interventions are interchangeable.

Internal validity. Much of the software-specific literature uses observational or qualitative designs; selection into work modes, prior performance, management quality, task allocation and home circumstances may confound observed relationships. Historical pandemic studies also capture restrictions and disruption beyond work location. The strongest causal studies have bounded interventions and settings, and even quasi-experimental estimates depend on assumptions that may not hold elsewhere.

External validity. Several influential studies concern large technology organisations with substantial digital infrastructure. Their findings may not transfer to smaller firms, geographically

dispersed contractors, regulated environments or less resourced teams. The selected literature does not provide a representative global sample, and there is insufficient evidence to specify optimal schedules for distinct demographic or occupational groups. The 2026 literature is also incomplete because the search update ends on 24 November.

Synthesis validity. Some primary studies may appear in the cited secondary reviews, so convergent reporting is not necessarily independent corroboration. The proposed framework and practice recommendations are interpretive; the absence of a preregistered extraction protocol and study-level quality ratings precludes formal confidence grading. A complete systematic or multivocal review would require a reproducible search, transparent screening records, explicit quality assessment and a publicly accessible extraction dataset.

9. Future research

9.1. Longitudinal and causal evaluation

Future research should distinguish policy assignment from actual attendance and record schedule alignment, task mix, project complexity and collaboration dependencies. Longitudinal studies could track delivery, defects, code-review feedback, workload, wellbeing and staff retention across multiple quarters. Randomised, cluster-randomised or staggered policy designs, where feasible and ethical, could estimate effects more credibly while accounting for spillovers between collaborators. The primary outcomes and analysis plan should be established before evaluation, including practical equivalence margins where the research question concerns the absence of harmful effects.

9.2. Career development, equity and contextual diversity

Research should examine career progression, mentorship access and skill acquisition alongside immediate output. Comparisons should include early-career and experienced developers, small and medium-sized companies, public-sector teams and organisations outside major technology hubs. Studies should distinguish voluntary from mandatory arrangements and investigate how commuting, accessibility, home environments and local labour norms influence preferences and outcomes. Qualitative work can help explain mechanisms that aggregate performance metrics may obscure; quantitative work can then test their prevalence and magnitude.

9.3. Generative AI and distributed engineering

Generative AI introduces an additional dimension of work design. AI assistants may reduce some coordination needs by helping developers locate information, understand unfamiliar code or draft documentation; they may also increase review demand, obscure knowledge gaps or create new quality risks. A 2025 preprint reporting a randomised study of 16 experienced open-source developers found longer task completion times when AI tools were available in that particular setting [34]. This result should not be generalised to later tools or other developer populations, and it does not evaluate hybrid work directly. It illustrates why perceived productivity gains require independent measurement.

A useful research design would compare AI-assisted and non-assisted workflows across deliberately defined remote and hybrid settings. Outcomes should include time to complete tasks, code quality, review effort, knowledge retention and collaboration patterns. Tool versions, experience and task complexity should be reported. The interaction between AI use and work mode should be estimated explicitly, rather than attributing changes observed during AI adoption to location policy alone.

9.4. Reproducible evidence synthesis

A future systematic MLR should extend the original search with a preregistered protocol, multiple bibliographic databases, backward and forward citation searching and a separate grey-literature

strategy. It should report complete queries, search dates, record counts, reasons for full-text exclusion and design-appropriate quality criteria. Extraction should capture both publication date and data-collection period, work-mode definitions, sample characteristics, outcomes, effect estimates and limitations. This would enable a transparent evaluation of whether apparently contradictory findings reflect differences in methods, policies or contexts.

10. Conclusion

Remote and hybrid work are viable operating models for software engineering, but their effectiveness cannot be inferred from physical location alone. The evidence reviewed indicates that hybrid flexibility can maintain measured performance and improve retention in a specific experimental setting, while software-specific studies identify opportunities for focused work and persistent challenges in distributed coordination. Collaboration, knowledge sharing and psychological safety require attention beyond the availability of digital tools. Proximity can contribute to feedback and skill development, particularly where engineering work is interdependent; flexibility can contribute to autonomy and work–life balance.

The central implication is that organisations should optimise sustainable engineering capability rather than short-term individual activity or attendance. Policy design should account for task interdependence, mentoring needs, accessible collaboration and workload, with outcomes evaluated at individual, team and organisational levels. The proposed framework makes these trade-offs explicit and identifies mechanisms for further empirical investigation. Given the limitations of the documented search and the heterogeneity of available evidence, no universal ranking of remote, hybrid and office-based work is warranted. The priority for future research is to establish which configurations work, for whom, under which conditions, and at what long-term cost or benefit.

References

[1] Ford, D., Storey, M.-A., Zimmermann, T., Bird, C., Jaffe, S., Maddila, C., Butler, J.L., Houck, B., Nagappan, N.: A Tale of Two Cities: Software Developers Working from Home during the COVID-19 Pandemic. ACM Trans. Softw. Eng. Methodol. 31, 27:1-27:37 (2022). https://doi.org/10.1145/3487567.

[2] Smite, D., Moe, N.B., Hildrum, J., Gonzalez-Huerta, J., Mendez, D.: Work-from-home is here to stay: Call for flexibility in post-pandemic work policies. Journal of Systems and Software. 195, 111552 (2023). https://doi.org/10.1016/j.jss.2022.111552.

[3] Šmite, D., Moe, N.B., Klotins, E., Gonzalez-Huerta, J.: From forced Working-From-Home to voluntary working-from-anywhere: Two revolutions in telework. Journal of Systems and Software. 195, 111509 (2023). https://doi.org/10.1016/j.jss.2022.111509.

[4] Khanna, D., Christensen, E.L., Gosu, S., Wang, X., Paasivaara, M.: Hybrid Work meets Agile Software Development: A Systematic Mapping Study. In: Proceedings of the 2024 IEEE/ACM 17th International Conference on Cooperative and Human Aspects of Software Engineering. pp. 57–67. Association for Computing Machinery, New York, NY, USA (2024). https://doi.org/10.1145/3641822.3641863.

[5] Christensen, E.L., Khanna, D., Broomandi, F., Paasivaara, M., Deshpande, A.: Revisiting hybrid work in agile software engineering: A systematic literature review. Journal of Systems and Software 245, 113160 (2027; published online 3 November 2026). https://doi.org/10.1016/j.jss.2026.113160.

[6] Forsgren, N., Storey, M.-A., Maddila, C., Zimmermann, T., Houck, B., Butler, J.: The SPACE of developer productivity: There's more to it than you think. Queue 19(1) (2021). https://doi.org/10.1145/3454122.3454124.

[7] Garousi, V., Felderer, M., Mäntylä, M.V.: Guidelines for including grey literature and conducting multivocal literature reviews in software engineering. Information and Software Technology. 106, 101–121 (2019). https://doi.org/10.1016/j.infsof.2018.09.006.

[8] Smite, D., Tkalich, A., Moe, N.B., Papatheocharous, E., Klotins, E., Buvik, M.P.: Changes in perceived productivity of software engineers during COVID-19 pandemic: The voice of evidence. Journal of Systems and Software. 186, 111197 (2022). https://doi.org/10.1016/j.jss.2021.111197.

[9] Rot, A., Sobinska, M., Busch, P.: Programming Teams in Remote Working Environments: an Analysis of Performance and Productivity. In: 2023 13th International Conference on Advanced Computer Information Technologies (ACIT). pp. 376–381 (2023). https://doi.org/10.1109/ACIT58437.2023.10275614.

[10] Bloom, N., Han, R., Liang, J.: Hybrid working from home improves retention without damaging performance. Nature. 630, 920–925 (2024). https://doi.org/10.1038/s41586-024-07500-2.

[11] De Souza Santos, R., Grillo, W.D.N., Cabral, D., De Castro, C., Albuquerque, N., França, C.: Post-Pandemic Hybrid Work in Software Companies: Findings from an Industrial Case Study. In: Proceedings of the 2024 IEEE/ACM 17th International Conference on Cooperative and Human Aspects of Software Engineering. pp. 68–78. Association for Computing Machinery, New York, NY, USA (2024). https://doi.org/10.1145/3641822.3641878.

[12] Ritonummi, S., Siitonen, V., Salo, M., Pirkkalainen, H., Sivunen, A.: Flow Experience in Software Engineering. In: Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering. pp. 618–630. Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3611643.3616263.

[13] Ritonummi, S., Kotlarsky, J., Siitonen, V., Salo, M., Pirkkalainen, H.: The Impact of Interruptions on Different Types of Flow in Collaborative Software Development Work. ICIS 2024 Proceedings. (2024). https://aisel.aisnet.org/icis2024/isdesign/isdesign/2/

[14] Smite, D., Tkalich, A., Moe, N.B., Chatzipetrou, P., Klotins, E., Helland, P.K.: Dual Effects of Hybrid Working on Performance: More Work Hours or More Work Time. In: Marchesi, L., Goldman, A., Lunesu, M.I., Przybyłek, A., Aguiar, A., Morgan, L., Wang, X., and Pinna, A. (eds.) Agile Processes in Software Engineering and Extreme Programming – Workshops. pp. 63–70. Springer Nature Switzerland, Cham (2025). https://doi.org/10.1007/978-3-031-72781-8_7.

[15] Nguyen-Duc, A., Khanna, D., Le, G.H., et al.: Work-from-home impacts on software project: A global study on software development practices and stakeholder perceptions. Software: Practice and Experience 54(5), 896–926 (2024). https://doi.org/10.1002/spe.3306.

[16] Yang, L., Holtz, D., Jaffe, S., Suri, S., Sinha, S., Weston, J., Joyce, C., Shah, N.P., Sherman, K., Hecht, B., Teevan, J.: The effects of remote work on collaboration among information workers. Nature Human Behaviour 6, 43–54 (2022). https://doi.org/10.1038/s41562-021-01196-4.

[17] De Souza Santos, R.E., Ralph, P.: A grounded theory of coordination in remote-first and hybrid software teams. In: Proceedings of the 44th International Conference on Software Engineering. pp. 25–35. ACM, Pittsburgh Pennsylvania (2022). https://doi.org/10.1145/3510003.3510105.

[18] Jackson, V., Prikladnicki, R., Van Der Hoek, A.: Co-Creation in Fully Remote Software Teams. In: Proceedings of the IEEE/ACM 46th International Conference on Software Engineering, pp. 1–12 (2024). https://doi.org/10.1145/3597503.3623297.

[19] De Andrade, A.S.L., Jackson, V., Prikladnicki, R., Van Der Hoek, A.: On meetings involving remote software teams: A systematic literature review. Information and Software Technology. 175, 107541 (2024). https://doi.org/10.1016/j.infsof.2024.107541.

[20] Christensen, E.L., Paasivaara, M., Salman, I.: Hybrid Work in Agile Software Development: Recurring Meetings. In: 2025 IEEE/ACM 18th International Conference on Cooperative and Human Aspects of Software Engineering (CHASE). pp. 120–130. IEEE, Ottawa, ON, Canada (2025). https://doi.org/10.1109/CHASE66643.2025.00023.

[21] Tkalich, A., Moe, N.B., Andersen, N.H., Stray, V., Barbala, A.M.: Pair Programming Practiced in Hybrid Work. In: 2023 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM). pp. 1–7. IEEE, New Orleans, LA, USA (2023). https://doi.org/10.1109/ESEM56168.2023.10304797.

[22] Emanuel, N., Harrington, E., Pallais, A.: The Power of Proximity to Coworkers. The Quarterly Journal of Economics 141(3), 1825–1870 (2026). https://doi.org/10.1093/qje/qjag027.

[23] Santos, R. de S., Magalhaes, C., Franca, C.: Hybrid Work Well-Being: Software Professionals Finding Equilibrium. IEEE Softw. 41, 78–85 (2024). https://doi.org/10.1109/MS.2024.3388551.

[24] Russo, D., Hanel, P.H.P., Van Berkel, N.: Understanding Developers Well-Being and Productivity: A 2-year Longitudinal Analysis during the COVID-19 Pandemic. ACM Transactions on Software Engineering and Methodology 33(3), 1–44 (2024). https://doi.org/10.1145/3638244.

[25] Godliauskas, P., Šmite, D.: The well-being of software engineers: A systematic literature review and a theory. Empirical Software Engineering 30, 35 (2025). https://doi.org/10.1007/s10664-024-10543-8.

[26] Alami, A., Zahedi, M., Krancher, O.: Antecedents of psychological safety in agile software development teams. Information and Software Technology. 162, 107267 (2023). https://doi.org/10.1016/j.infsof.2023.107267.

[27] Alami, A., Zahedi, M., Krancher, O.: The role of psychological safety in promoting software quality in agile teams. Empir Software Eng. 29, 119 (2024). https://doi.org/10.1007/s10664-024-10512-1.

[28] Tkalich, A., Šmite, D., Andersen, N.H., Moe, N.B.: What Happens to Psychological Safety When Going Remote? IEEE Software. 41, 113–122 (2024). https://doi.org/10.1109/MS.2022.3225579.

[29] Souza Santos, R. de, Adisaputri, G., Ralph, P.: Post-pandemic Resilience of Hybrid Software Teams. In: 2023 IEEE/ACM 16th International Conference on Cooperative and Human Aspects of Software Engineering (CHASE). pp. 1–12 (2023). https://doi.org/10.1109/CHASE58964.2023.00009.

[30] Santos, R. de S., Magalhaes, C., Santos, R., Correia-Neto, J.: Exploring Hybrid Work Realities: A Case Study with Software Professionals from Underrepresented Groups. In: Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering. pp. 27–37. Association for Computing Machinery, New York, NY, USA (2024). https://doi.org/10.1145/3663529.3663825.

[31] Kemell, K.-K., Saarikallio, M.: Hybrid Work Practices and Strategies in Software Engineering-Emerging Software Developer Experiences. IEEE Access. 11, 112861–112876 (2023). https://doi.org/10.1109/ACCESS.2023.3322934.

[32] Broomandi, F.: Adapting agile software development in hybrid work environments through a comparative multiple case study. Master's thesis, LUT University (2024). https://lutpub.lut.fi/handle/10024/168819.

[33] Saeeda, H., Ovais Ahmad, M., Gustavsson, T.: Navigating social debt and its link with technical debt in large-scale agile software development projects. Software Qual J. 32, 1581–1613 (2024). https://doi.org/10.1007/s11219-024-09688-y.

[34] Becker, J., Rush, N., Barnes, E., Rein, D.: Measuring the Impact of Early-2025 AI on Experienced Open-Source Developer Productivity. arXiv preprint arXiv:2507.09089 (2025). https://doi.org/10.48550/arXiv.2507.09089.