

The Fast and Spurious: Developer Productivity with GenAI

Anonymous Author(s)

Abstract

Generative AI (GenAI) tools are increasingly being adopted in software development as productivity aids, since there is evidence that GenAI tools can improve individual aspects of productivity. However, productivity is multidimensional; accelerating one aspect of work may simply shift effort to another. In this paper, we investigate how GenAI adoption affects different dimensions of developer productivity. We surveyed 415 software practitioners to understand how they perceive productivity changes associated with AI adoption, using the SPACE framework (Satisfaction and well-being, Performance, Activity, Communication and collaboration, and Efficiency and flow). Our results reveal systematic redistribution of effort across SPACE dimensions. While frequent GenAI users reported faster task completion and higher output volume, these gains were offset by increased code review burden, persistent cognitive load from output verification, and unchanged collaboration patterns. We further provide an empirical mapping between the challenges perceived by developers and potential strategies to mitigate them. Overall, our findings suggest that, at the current stage of GenAI adoption, perceived productivity gains may be *spurious*—surface-level acceleration, often accompanied by redistributed effort and hidden costs.

Keywords

Generative AI, developer productivity, SPACE

ACM Reference Format:

Anonymous Author(s). 2026. *The Fast and Spurious: Developer Productivity with GenAI*. In *Companion Proceedings of the 34th ACM Symposium on the Foundations of Software Engineering (FSE '26)*, June 5–9, 2026, Montreal, Canada. ACM, New York, NY, USA, 9 pages. <https://doi.org/XXXXXXX.XXXXXXX>

1 Introduction

Generative AI (GenAI) tools have experienced rapid growth, with software development emerging as an early adopter [21, 44]. According to the 2025 Stack Overflow Developer Survey, nearly 84% of developers use or plan to use GenAI tools [30]. This rapid adoption is driven by a widespread belief that GenAI can boost developer productivity [43], yet there is limited evidence to understand if GenAI improves the overall productivity, or merely creates a spurious sense of speed.

To ensure that any observed productivity gains are real and sustainable, it is necessary to consider how GenAI integrates with software engineering (SE) practices. Without clear

process safeguards, GenAI may shift work effort without reducing it—or worse, introduce new forms of technical debt and developer burnout. For example, Moreschini et al. [20] showed that GenAI can incur prompt engineering debt and explainability debt, leaving teams with code that may “work” but lacks clarity, testability, or adaptability. Similarly, Feng et al. [7] demonstrated that GenAI adoption can heighten burnout by increasing job demands on developers. Thus, achieving productivity gains with GenAI requires alignment with established SE rigor and empirical investigation. In this context, integrating GenAI into software development is not just a tooling decision, but a socio-technical design challenge [9].

The evidence of GenAI’s impact on productivity is mixed. While GenAI tools such as GitHub Copilot and ChatGPT have been reported to automate routine tasks and improve efficiency [25, 46], Tong [35] found that developers completed tasks 19% slower when using AI, and Vaithilingam et al. [39] observed higher task-failure rates and no significant improvement in completion time. Such inconsistencies likely arise because existing studies often scope their study to particular development activities or narrow productivity metrics rather than taking a comprehensive view.

However, productivity in software development is a multi-dimensional socio-technical system [12]. It is not simply measurable by the output volume. It includes long-term work quality, psychological well-being, team collaboration, uninterrupted focus, and satisfaction with work [12, 28, 33]. When organizations optimize one dimension, they may inadvertently create other constraints. Understanding these interdependencies is critical for sustainable GenAI adoption.

To investigate this socio-technical interplay, we adopt the multidimensional productivity framework SPACE [12] as our analytical lens for investigating developer productivity across five dimensions: *Satisfaction and well-being*, *Performance, Activity, Communication and collaboration*, and *Efficiency and flow*. We employ SPACE as a measurement taxonomy and a structuring lens to understand how perceived productivity gains in one dimension may introduce new demands or trade-offs in others. This systems-oriented perspective helps characterize how developers’ effort and attention shift during GenAI adoption, pinpoint where challenges emerge, and inform strategies for designing more resilient software engineering practices for AI-infused workflows. This motivates us to answer the following research questions:

- *RQ1. How does GenAI adoption affect developer productivity across multiple dimensions?*
- *RQ2. What productivity-related gaps, challenges, and strategies do developers perceive in GenAI adoption?*

To address these research questions, we conducted a large-scale survey of **415 professional developers** grounded in



This work is licensed under a Creative Commons Attribution 4.0 International License.

FSE'26, Montreal, Canada

© 2026 Copyright held by the owner/author(s).

ACM ISBN XXX-XXXX/XXXXXXXXX

<https://doi.org/XXXXXXX.XXXXXXX>

the SPACE framework. We analyzed the data using a mix of quantitative and qualitative methods.

As a result, our study situates developers' experience with GenAI within the *SPACE* framework to provide emerging empirical results that raise important discussion questions about what it means to be "fast" in the context of broader software development, and the potential for perceived "spurious" forms of productivity in AI-mediated software development. *"I think in the current state, AI can give you a productivity boost by bringing you 70% there in a few minutes. However, the last 30% will likely take you close to what you would've done before because you also have to thoroughly review AI code."* [P20].

Our contributions are twofold. First, combining quantitative patterns with qualitative explanations, we characterize how developers perceive GenAI's productivity impact across five dimensions. Overall, developers do not report substantial productivity changes, although frequent GenAI users perceive modest improvements in *Efficiency and flow* and *Satisfaction and well-being*, with limited perceived gains in *Performance, Activity, or Communication and collaboration*. Second, we develop an empirical mapping between the productivity-related challenges developers encounter and the strategies proposed to address them, providing actionable guidance for practitioners integrating GenAI into their workflows.

2 Background and Related Work

As GenAI rapidly enters everyday software development, understanding existing productivity measurement frameworks and prior GenAI studies has become a key focus in SE research.

2.1 Frameworks for Measuring Productivity

Developer productivity is a complex, multifaceted concept that has been studied by both academics and practitioners [25]. Traditional metrics such as lines of code (LoC) [19], commit counts [40], and task completion rates [45] capture only narrow aspects of work and can be misleading or easily gamed. For example, the number of commits is often inflated by shallow or trivial changes [23], and LOC measures may reward verbosity rather than efficiency. Speed and volume of activity alone do not present a complete picture of productivity. Other forms of work—such as design, problem solving, and knowledge sharing—are equally important [12], as are software quality, impact, and delivery speed [32].

One such framework is the Developer Experience (DevEx) framework [22], which highlights feedback loops, cognitive load, and flow state as key drivers of developer effectiveness. At the system level, the DORA (DevOps Research and Assessment) metrics [41] operationalize software delivery performance through deployment frequency, change lead time, mean time to recovery, and change failure rate. While these frameworks broaden the perspective beyond code quantity, they primarily address specific contexts of individual experience (DevEx) and delivery throughput (DORA).

Forsgren et al. [12] proposed the *SPACE* framework, a model that conceptualizes productivity as a combination of interpersonal and technical dimensions. The framework emphasizes

that productivity arises from the interplay among human, technical, and organizational factors. For instance, high *activity* (e.g., frequent commits) does not necessarily indicate improved *performance* unless accompanied by quality, sustained satisfaction, collaboration, and flow. In this study, we adopt the *SPACE* framework as an analytical lens to examine how GenAI adoption influences developer productivity across its five dimensions: *Satisfaction and well-being*, *Performance, Activity, Communication and collaboration*, and *Efficiency and flow*.

Satisfaction and well-being captures how fulfilled, motivated, and supported developers feel in their work. *Performance* represents the quality and impact of outcomes (e.g., software reliability, feature completeness, or user satisfaction). *Activity* refers to the volume of work performed (e.g., commits and code reviews). *Communication and collaboration* refers to how developers communicate and work together. *Efficiency and flow* describes how smoothly and continuously work proceeds (e.g., ability to maintain focus and minimize interruptions).

Critically, the *SPACE* framework views productivity as a system of interdependent dimensions rather than isolated metrics. High activity without corresponding performance gains may indicate redistributed rather than reduced effort. Similarly, efficiency gains that increase cognitive load trade one constraint for another. This systems view positions *SPACE* as both a measurement and planning framework for understanding how interventions affect the full productivity landscape.

2.2 Empirical Evidence on Developer Productivity with GenAI

As GenAI tools are increasingly integrated into software development, recent studies have highlighted productivity gains through faster task completion. For example, Dohmke et al. [6] reported that developers using GitHub Copilot completed programming tasks up to 55.8% faster. In a randomized controlled trial with 96 full-time Google engineers, Paradis et al. [24] found that AI assistance reduced time spent on a complex, enterprise-grade coding task by approximately 21%. Additionally, Rodriguez [29] found that GenAI accelerated task completion and streamlined code review processes by suggesting actionable improvements and reducing reviewer effort.

However, recent literature suggests that GenAI's productivity benefits may be overstated. AI-generated code can be inconsistent or incorrect, often requiring developers to re-prompt, validate, and debug the output, which then interrupts their workflow and reduces overall efficiency [2, 18]. Kuhail et al. [14] found that over-reliance on AI assistance may erode developers' coding proficiency and critical thinking, while reducing peer interaction, such as asking fewer questions, sharing fewer solutions, and engaging less in coordination [13, 26, 34]. Moreover, the 2025 Stack Overflow Survey further reported that only 29% of developers trusted AI accuracy and 66% spent more time debugging than expected [30].

These studies offer fragmented views of GenAI's impact on software development, focusing on programming tasks and short-term efficiency gains while overlooking broader productivity gains. This paper presents a comprehensive analysis

Table 1: Survey Items mapping

| Survey Items | |
|--|---|
| Satisfaction and well-being | |
| S1 | I feel my current workload is manageable |
| S2 | I feel mentally and physically exhausted from work (r)* |
| S3 | I am becoming less interested in work (r)* |
| S4 | I feel secure in my job and confident about my future in this company |
| Performance | |
| P1 | The number of lines of code I change per day |
| P2 | The proportion of my test cases that pass |
| P3 | The number of new API methods I learn each day |
| Activity | |
| A1 | The number of commits I make |
| A2 | The amount of time I devote to writing code |
| A3 | The number of test cases I write |
| A4 | The amount of time I spend browsing the web for work-related information (r)* |
| A5 | The number of work items (tasks, bugs) I close |
| A6 | The number of code reviews I contribute to |
| A7 | The amount of time I devote to reviewing code (r)* |
| Communication and collaboration | |
| C1 | The number of meetings I attend (r)* |
| C2 | The amount of time I spend in meetings (r)* |
| C3 | The number of work-related emails I write |
| C4 | The amount of time I spend responding to email (r)* |
| Efficiency and flow | |
| E1 | The amount of time I spend on each work item (r)* |
| E2 | The amount of time I spend browsing the web for personal matters during work (r)* |

*(r) indicates the items reverse-coded for visualization.

of developer productivity in GenAI-mediated development, grounded in the multi-dimensional *SPACE* framework.

3 Research Method

To capture developers' multi-dimensional productivity perceptions with genAI, we used a survey-based approach grounded in the *SPACE* framework [12].

3.1 Survey Design

We designed our survey instrument to comprise the five dimensions of the *SPACE* framework for productivity measurement [12]. Since at the time of the survey design there was no validated questionnaire for *SPACE*, we adapted the questionnaires from Casic and Panselina [4], Meyer et al. [16], Trinkenreich et al. [36], which measured developer satisfaction, activity, performance, collaboration, and efficiency. The adaptations were made to reflect the GenAI context. See the complete questionnaire in the supplementary material [1]. Respondents were offered the option to enter a \$50 raffle for every 50 responses. The protocol was approved by the university's IRB.

Survey Questions: We first asked questions about participants' demographics, including gender, professional background (role, years of experience, seniority level), and organization size. We also collected information about genAI-usage experience (frequency of use, adapted from [27]) to assess their perceived impact on productivity. The rest of the survey focused on each of the dimensions in the *SPACE* framework: **Satisfaction and well-being** measures developers' perception of fulfillment, motivation, and support about their work and team [12]. Four statements assessed (1) workload manageability, (2) physical and mental exhaustion, (3) engagement, and (4) job security [4, 36].

Performance captures perceived outcomes of the development process, such as velocity [12]. We used three items adapted from Meyer et al. [16] to assess the impact of AI on: (1) number of lines of code changed per day, (2) proportion of test cases passed, and (3) number of API methods learned.

Activity presents the volume of work performed [12], measured through seven statements adapted from Meyer et al. [16]: (1) number of commits, (2) time spent writing code, (3) number of test cases, (4) time spent doing work-related browsing, (5) number of closed work items, (6) number of code reviews, and (7) time spent reviewing code.

Communication and collaboration describes how developers interact and coordinate [12]. We adapted four statements from Meyer et al. [16]: (1) number of meetings attended, (2) time spent in meetings, (3) number of work-related emails written, and (4) time spent responding to emails.

Efficiency and flow captures making progress with minimal interruptions [12]. We assessed this by using two adapted statements from Meyer et al. [16]: (1) time spent per work item and (2) time spent on personal browsing during work.

Finally, we added an open-ended question inviting participants to share reflections on how AI adoption has influenced their work. The survey took approximately 5–8 minutes to complete, and we held a raffle for a \$50 gift card, which participants could enter by sharing their email.

Sandbox and Pilot Survey. We sandboxed the survey with five people with software engineering research experience, iteratively refining it until no further concerns arose. We then piloted with two professionals: one from a multinational organization and one from a U.S. national lab; both found the survey clear, easy to follow, and relevant to assess productivity.

Participant Recruitment. Following ethical data collection guidelines, we recruited participants from diverse technical backgrounds, expertise, organization sizes, and domains. We recruited participants from 56 OSS communities, including organizational repositories (e.g., IBM, Oracle, Google), widely-used infrastructure, AI projects, and data science communities (e.g., PyTorch). We invited participants via email with study details and consent information. Responses were anonymized per GDPR and IRB approval. This two-week survey followed established SE research practices [8, 10, 11].

We received 688 responses. After removing invalid entries, we kept 415. Most participants worked in full-stack (36.9%), backend (16.9%), or data/ML roles (15.4%). A majority identified as men (90.6%), over half worked at large or extra-large organizations (57.8%), and most (82.2%) had more than five years of experience. More details in the supplementary material [1].

3.2 Data Analysis

RQ1.Productivity Impacts of GenAI Adoption. To answer RQ1, we analyzed how productivity perceptions vary by GenAI usage frequency. We categorized participants based on their response to "how often do you use AI tools for your software development work?" into non-frequent (Never, Rarely, Sometimes) and frequent users (Often, Always).

We analyzed the Likert-scale response distributions and trends using stacked bar charts and violin plots. The analysis focused on the five dimensions of the *SPACE* framework: Satisfaction and well-being (S1–S4), Performance (P1–P3), Activity (A1–A7), Communication and collaboration (C1–C4), and Efficiency and flow (E1–E2). To improve interpretability, we

reverse-coded selected items (e.g., S2 and S3). Table 1 presents the mapping of the survey items. The complete set of charts is available in the supplementary material [1].

RQ2. Perceived Productivity Gaps, Challenges, and Strategies. To understand developers’ experiences with GenAI adoption and to identify perceived gaps, challenges, and potential strategies, we addressed RQ2 through a qualitative analysis of responses (N = 206) to the open-ended question: *“If you have any final thoughts or experiences on how adopting AI tools has impacted your work, please share them below.”*

We used the five dimensions of SPACE as our codeset. Two authors independently coded 20% of the responses, and compared their outputs using the Jaccard index [15], achieving 90% agreement on inter-rater reliability. Given this level of consensus [15], one of the researchers coded the remaining responses. Based on this analysis, we developed an empirical mapping that maps seven identified gaps/challenges and eight potential strategies onto the SPACE dimensions.

4 Results

Section 4.1 presents the results for RQ1, examining how GenAI adoption impacts different dimensions of developer productivity through the SPACE framework, highlighting where productivity improves and where it remains unchanged. Section 4.2 reports the results for RQ2, identifying the remaining gaps and challenges in using GenAI, and the potential strategies to address those.

Our results show that GenAI adoption creates a constraint redistribution problem. Section 4.1 documents how GenAI-facilitated improvements in Activity and Efficiency dimensions create demands in Satisfaction, Performance, and Communication dimensions. Section 4.2 identifies specific constraint patterns and strategies that teams planning GenAI adoption or GenAI tool developers can use to anticipate these tradeoffs.

4.1 Productivity Across SPACE Dimensions (RQ1)

To answer RQ1, we begin by examining overall trends across the SPACE framework’s dimensions, and then drill down into item-level variations within each dimension.

Figure 1 displays the median values for non-frequent and frequent AI user groups across the SPACE dimensions. Frequent AI users have slightly higher median scores than non-frequent users in all dimensions except *Communication and collaboration*, where medians are the same. Nevertheless, all values remain *within the neutral (or “no-change”)* range, as indicated by the gray-shaded area. These results suggest that GenAI adoption has not yet produced substantial positive or negative shifts in productivity.

These overall patterns indicate that, at an aggregate level, GenAI adoption has not yet led to pronounced positive or negative shifts in developers’ perceived productivity. Instead, the effects appear incremental and uneven, with small median differences that may mask meaningful variation at the item level. Accordingly, we next examine each SPACE dimension in detail to unpack where GenAI adoption is associated with

concrete improvements, where effects are limited, and where challenges persist across specific productivity aspects. Refer to the supplementary item-level stacked bar charts (Figs. 1–5) for each dimension of SPACE.

Satisfaction and well-being. Among frequent AI users, positive ratings increased for workload manageability (S1) and perceptions of job security (S4): 68.6% and 55.4% *Agreed/Strongly Agreed*, respectively. A similar pattern was observed among non-frequent AI users, 55.3% (S1) and 52.9% (S4) reported positive perceptions on these items.

Developers described these gains as GenAI easing repetitive tasks in their work, *“I no longer need to waste my time with repetitive tasks. There is always a way to automate with AI [P369]”*, *“AI tools have had an overall positive impact on my work satisfaction; fewer mundane tasks, more focus time. [P3]”*

However, these improvements are partial; more than half of respondents still reported feeling exhausted (S2: 65.2% vs. 62.8%). Participants noted that efficiency gains can accompany expectations to deliver more and faster, particularly in contexts that overestimate AI’s capabilities. As one participant observed, *“There’s also a growing expectation to deliver more simply because these tools are available, and sometimes non-technical stakeholders view AI as a ‘magic ball’ that can solve everything.”* Around half of the participants became less interested in work (S3: 46% for frequent and 59.6% for non-frequent users). *“I move fast with AI and move mountains of work, but I am losing my passion for the craft [P201]”*.

Satisfaction and well-being (S)

- More developers report manageable workloads.
 - Increased confidence in job security.
 - Reduced disinterest in daily work.
-
- High levels of exhaustion persist despite AI adoption.

Performance. Non-frequent AI users predominantly reported *no change* in work quality or outcomes across performance-related items (P1: 66.7%, P2: 83.1%, P3: 66.7%). In contrast, among frequent users, coding throughput (P1) increased: 72.7% (47.2% *More* + 25.5% *Much more*). However, these gains did not extend uniformly to other performance items. For test case pass rates (P2) and learning velocity (API methods learned per day, P3), most of the frequent AI users—67.4% and 58.6% respectively—reported *no change* or a decline.

Qualitative responses help explain this divergence. While AI-driven acceleration allows developers to deliver more lines of code and generate broader test coverage, increased volume often shifts effort to review and validation, limiting net performance gains. *“When I use code that’s been written with AI, I feel slightly less control over my outcomes, even with through unit tests [P138]”*. Besides, the absence of oversight may constrain genuine performance improvement, highlighting that *speed does not always equal progress*. *“By my estimate, LLM-based tools have so far been more about speed than quality improvement [P5]”*.

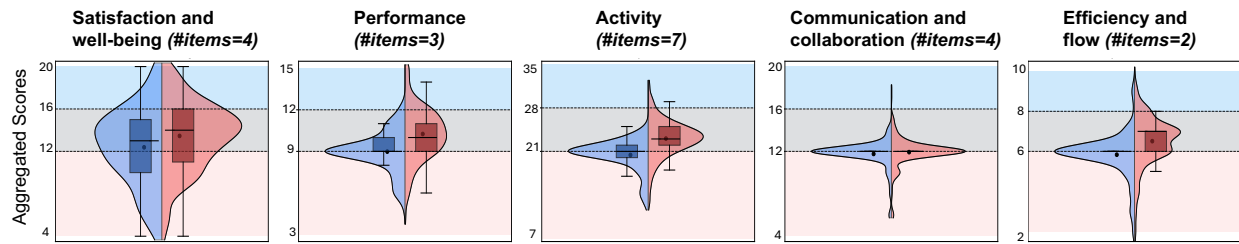


Figure 1: Violin plots of aggregated SPACE scores. Left = non-frequent AI users (blue), right = frequent AI users (red). Boxplots show median (line) and mean (dot). Gray band marks the neutral (“no-change”); blue marks positive and red negative perceptions; example tick labels shown for *Satisfaction & Well-being* (4 items: 12 = all Neutral, 16 = all Agree).

Performance (P)

- Higher coding throughput with frequent GenAI use.
- Greater volume of code changes delivered per day.

- Test success rates show little to no improvement.
- Learning velocity remains largely unchanged.

Activity. In most activity-related items, frequent AI users reported higher levels of activity than non-frequent users, except code review (A6 and A7). Frequent AI users were less likely to report spending more time writing code (A2) or searching for work-related information (A4), with 75.4% and 61.3% of them, respectively, reporting no increase. Moreover, a larger share of frequent users reported producing more commits (A1): 48.3% vs 7.9%; more test cases (A3): 56.5% vs 24.8%; and completing more work items (A5): 55.9% vs 9.1%, compared to non-frequent users. “*Agentic systems streamline small coding tasks and free up time for design thinking. [P1]*” and “*I spend less time on generating the code, and more time evaluating its behavior and design [P104]*.”

However, these activity gains are accompanied by increased review effort. The majority of frequent users (84.3%) reported that GenAI did not reduce the time spent on code reviews (A7), and frequent users were more likely than non-frequent users to report conducting more code reviews (A6: 25.1% vs. 9.8%). As one participant noted, “*AI-generated code unfairly puts more onus on code reviewers to understand how the code works and find bugs or security issues.[P204]*” Developers emphasized that the acceleration often shifts effort toward validation and maintenance. “*Reviewing LLM-generated content such as code and docs wastes time – coworkers are (accidentally but carelessly) sabotaging our work by ‘creating work’.The LLMs save these coworkers time because they are faster at producing content, but other coworkers have to spend disproportionately more time to review and correct all that content [P127]*.” Despite this shift, developers viewed the trade-off favorably sometimes, noting that “*adopting AI tools has greatly improved my workflow, especially by enabling the faster creation of tests and use cases. While I do spend more time on projects now, it’s still far less than it would take to create everything manually, and the overall quality and coverage have improved significantly.[P338]*”

Activity (A)

- Increased output of commits, test cases, and completed work items.
- Reduced time on direct code writing.
- Reduced time searching for work-related information.

- Increased involvement in code review activities.
- No reduction in time spent reviewing code.

Communication and collaboration. Across all communication and collaboration items (C1–C4), most of the developers reported no positive change, regardless of AI usage frequency. Specifically, more than three-quarters of all users reported no positive change (i.e., reporting *No Change*) for meeting frequency (C1: 93.1% vs 90.8%), time spent in meetings (C2: 92.3% vs 90.1%), work-related emails written (C3: 93.6% vs 91.3%), and time spent responding to emails (C4:76.3% vs 82.3%). Among frequent AI users, *No Change* responses dominate across all four items, exceeding 70% in each case, indicating that GenAI adoption has not substantially altered day-to-day communication practices.

However, participants noted GenAI reducing friction in personal communication tasks: “*Superhuman’s AI model lets me search my inbox and answer questions related to email with very little time and effort. This saves at least an hour a week.[P287]*”

Communication and Collaboration (C)

- Team communication and collaboration patterns remain largely unchanged with GenAI use.
- Meetings and email-related activities show little to no reduction.

Efficiency and flow. Frequent AI users were less likely than non-frequent users to report spending more time on each work item (E1: 35.8% vs 82.2%). Developers attributed this gain to AI’s ability to accelerate a wide range of knowledge and content creation tasks: “*By leveraging AI across various tasks such as writing, ideation, presentation creation, coding, information gathering, and research, I have been able to achieve significant results in a very short period of time. [P180]*” and “*AI has improved my efficiency by helping me not spend time checking syntax so I can focus on functionality [P321]*.”

Table 2: The productivity challenges and potential strategies for the negative observations

| SPACE | Observations | Challenges | Potential Strategy |
|-------|---|--|--|
| S | High levels of exhaustion persist despite AI adoption. | Ch1. AI-induced cognitive workload from verifying outputs. Ch2. AI shifts effort to reviewing others' AI outputs rather than reducing total workload. Ch3. Organizational "AI = faster output" expectations increase pressure and stress. | St1. Structured organizational training on using GenAI. St2. Set team/management norms framing GenAI as assistive rather than a driver of higher output expectations. St3. Surface uncertainty via confidence indicators and explanation norms. |
| P | Test success rates show little to no improvement. Learning velocity remains largely unchanged. | Ch4. Verbosity of AI outputs. Ch5. Reliance on AI for tasks before acquiring the necessary foundational knowledge | St4. Integrate GenAI tools with project-specific context. St5. Pair GenAI use with structured fundamental learning resources. |
| A | Increased involvement in code review activities. No reduction in time spent reviewing code. | Ch2. AI shifts effort to reviewing others' AI outputs rather than reducing total workload. Ch4. Verbosity of AI outputs. | St6. Use GenAI for first-pass review, not final approval. St3. Surface uncertainty via confidence indicators and explanation norms. St7. Add quality gates for AI-heavy changes. |
| C | Team communication and collaboration patterns remain largely unchanged with GenAI use. Meetings and email-related activities show little to no reduction. | Ch6. GenAI primarily supports individual tasks rather than shared coordination work. Ch7. Collaboration relies on human alignment rather than automatable tasks. | St2. Set team/management norms that treat GenAI as assistive. St8. Define guidelines for GenAI use in shared communication artifacts. |
| E | Improvements in sustained focus and flow are limited. | Ch1. AI-induced cognitive workload from verifying outputs. | St1. Structured organization level training on using GenAI. |

Frequent AI users also reported spending less time than non-frequent AI users on personal web browsing during work (E2: 19% vs. 7.1%). However, the fact that 76.3% of frequent AI users reported non-positive responses regarding reductions in interruptions indicates that GenAI use alone does not mitigate context switching or external disruptions.

Efficiency and Flow (E)

- Reduced time on individual work items.
- Reduced time on non-work-related web browsing.
- Improvements in sustained focus and flow are limited.

4.2 Productivity Challenges and Potential Strategies (RQ2)

To answer RQ2, we qualitatively analyzed open-ended responses using the SPACE dimensions to identify the gaps and challenges underlying the observations reported in the RQ1 results (highlighted in red in Section 4.1), as well as perceived strategies to mitigate them. Table 2 shows an empirical mapping of these challenges and the corresponding strategies.

Satisfaction and well-being. Persistent high levels of exhaustion despite GenAI adoption emerged as the primary negative observation in the RQ1 results. One contributing factor is the **cognitive load** (Ch1) associated with repeated prompting, verification, debugging, and error correction of AI outputs. "I'm able to close our features faster, but for the cost of significantly higher human brain compute load. [P207]" This also includes **review burden** (Ch2) of others' AI generated code. "...I was feeling resentment because I was sure I was spending more time reviewing and re-reviewing her doc than she had spent working on it.[P117]" Moreover, the perception that GenAI accelerates output generation can raise organizational

expectations, resulting in **increased pressure and stress on developers** (Ch3). "AI tools so far have been similar to other types of tools in practice... but with much more management hype / misplaced expectations and general feelings of dread [P160]"

Surfacing uncertainty through confidence indicators and explanation norms (St3) can help developers calibrate trust in GenAI outputs better, and reduce review burden. For example, when submitting a pull request, developers could be required to indicate which parts were AI-generated, their confidence in the correctness of those changes, and any verification steps taken. As one participant noted, "Junior developers don't know how to use it and they blindly apply it without thinking it through before submitting. [P234]" Additionally, because organizational norms strongly shape how GenAI affects well-being, teams and management should foster an environment that treats **GenAI as assistive rather than as a mechanism to push for increased output** (St2). "Having training and clear policy for use has really helped our team to adopt the tools in a positive manner.[P268]" Subsequently, **organizational-level training** (St1) and clear usage policies can support more positive adoption. "My company has made broad statements encouraging us to use AI to help us be more productive, but has never offered any training on how to do it.[P351]"

Performance. Due to the **verbosity of AI-generated outputs** (Ch4), particularly in test generation, test success rates may show little to no improvement. "AI generates many test cases, including verbose ones, which makes code-reviewing more time-consuming [P23]." Accordingly, **reliance on AI for tasks before acquiring sufficient foundational knowledge** (Ch5) contributes to unchanged learning velocity. Developers were concerned that relying on GenAI too early can bypass key learning processes, especially for junior developers: "I fear that juniors will take much longer to get to an experience level that is comparable to what devs with 10+ years of experience. [P192]"

One potential strategy to mitigate challenges related to test quality is to **integrate GenAI tools with project-specific context** (St4), including providing access to design decisions, architectural design, and broader project context, so that AI-generated tests better reflect system intent rather than generic assumptions based on small chunk of code. *“We only use AI as if it were a ‘genius graduate’ junior coder that does not understand business needs and workflow. [P14]”*

Developers highlighted the importance of **pairing GenAI use with structured fundamental learning resources** (St5) rather than relying on AI as a substitute for fundamental skill development in situations where the task demands knowledge beyond their existing expertise. *“What it concerns me is that when I ask it to write code in languages I have a lot of expertise in, it’s all bad and I hate it, so that makes me a lot less confident in the code it generated for languages I know a lot less well. [P57]”* *“I’d prefer my teammates embrace investing in themselves and learning fundamentals instead of relying on AI to produce sub-par code. [P415]”*

Activity. With GenAI adoption, developers reported increased involvement in code reviews without a corresponding reduction in workload, reflecting a shift in effort rather than a net savings. In particular, **AI shifts effort toward reviewing others’ AI-generated outputs** (Ch2), as higher output volumes increase supervision needs. *“There’s an increase in the amount of effort and time I’m reviewing work of other folks cranking out more stuff with the help of AI tools. [P359]”* This burden is further compounded by the **verbosity of AI outputs** (Ch4), which often results in low-quality content. *“...Reviewing that low-quality content and giving feedback on it consumes a lot of time [P127].”* *“Sometimes you spend more time reviewing than if you had written it yourself. [P207]”*

Using AI as a **first-pass reviewer rather than as a source of final, review-ready content** (St6) may help reduce review burden. *“Reviewing code written by AI is painful, when the author did not take care to clean it up. [P355]”*. Additionally, as discussed earlier, **surfacing uncertainty via confidence indicators and explanation norms** (St3) could prevent additional supervision work from being pushed downstream. *“AI tools have shifted me into a role where I spend more time performing detailed review of code [P104].”* Together, these observations highlight the importance of introducing **quality gates for AI-heavy changes** (St7) to mitigate review overhead. *“LLM approach needs strict guidance and ideal examples to mimic. If your code is sloppy or undocumented, your results will probably be poor. It’s just a fuzzy mirror. [P87]”*

Communication and collaboration. Our RQ1 findings indicate that Communication and collaboration showed little to no change with the adoption of GenAI. We do not frame outcome of this as a challenge; instead, it highlights an explanatory gap regarding where GenAI’s impact is limited. A potential explanation is that GenAI tools currently emphasize **support for individual work activities, offering less direct support for shared coordination and collective practices** (Ch6). Developers’ experiences suggest that while GenAI can assist with individual productivity, it does not substantially alter

how teams align or coordinate work. This aligns with prior empirical findings showing that AI tools tend to optimize individual task execution but have limited impact on collaborative practices such as meetings and coordination routines [5].

Moreover, **collaboration often depends on human alignment** (Ch7) including shared understanding, and social negotiation, which are difficult to automate through GenAI alone [42]. *“Creativity gets a collaborator, not a replacement. [P269]”*

Therefore, developers and prior work pointed toward organizational strategies that clarify GenAI’s role in collaborative settings. First, setting team- and management-level norms that frame **GenAI as assistive** (St2) rather than as a replacement may help prevent over-reliance on AI at the individual level while preserving human-centered collaboration.

In addition, **defining guidelines for GenAI use in shared communication artifacts** (St8) emerged as a relevant strategy, aimed at making individual and ad-hoc GenAI use in communication work more explicit and collectively aligned [38]. *“Our organization does use it more regularly for non-code tasks such as meeting notes/summaries, incident summaries, documentation, etc. [P75]”* While prior studies suggest that GenAI may reduce time spent in less relevant meetings and lower effort devoted to non-critical email communication [17, 31], our findings indicate that such collaboration-related practices remained largely unchanged. One possible explanation is that GenAI use in shared communication artifacts currently lacks shared norms or guidance.

Efficiency and flow. Developers faced challenges related to **AI-induced cognitive workload from repeatedly verifying and correcting outputs** (Ch1), which may undermine their ability to maintain concentration and result in limited improvements in sustained focus and flow. *“I move fast with AI and move mountains of work, but I am losing my passion for the craft and the ability to quickly focus. [P201]”*

To mitigate this challenge, developers emphasized the importance of **structured, organization-level training on how to use GenAI effectively** (St1). Such training (e.g., prompt engineering) can help developers better scope problems, recognize when verification is necessary, and avoid inefficient prompting and rework. *“If you have a very well-defined problem and some strong test cases, AI can really get there. [P20]”* *“It massively improved certain refactors and updates, but required adapting approaches to tasks to ensure best usage of AI tools. [P57]”*

5 Limitations

Limitations of Perceptual Productivity Assessment. This study captures developers’ perceptions of GenAI’s impact on their productivity using self-reported survey data, which may not reflect objective productivity outcomes. However, we grounded our survey design in well-established productivity frameworks and prior validated measures used in related work [4, 16, 36]. Future studies are needed to complement perceptual findings with additional empirical analyses (e.g., repository mining or longitudinal behavioral measures).

Limitations of Adoption Frequency as a Proxy. To address RQ1, we compared developers with different frequencies of

GenAI use to investigate how adoption relates to productivity perception. While usage frequency provides a practical indicator of GenAI engagement, it may not fully capture other important dimensions of adoption, such as the types of tasks supported, the depth of integration into workflows, or reliance on specific GenAI features. To provide a more comprehensive perspective, we designed our analysis across multiple SPACE dimensions, and we complemented closed-ended items with an open-ended question to contextualize participants' experiences. Future work should consider richer operationalizations of GenAI adoption (e.g., task-level or longitudinal behavioral measures) to further characterize its productivity impact.

Limitations of Sample Representativeness. We acknowledge that no single sample can fully represent the global software workforce. However, our dataset includes 415 software practitioners from 56 organizations, which is comparable in scale and diversity to prior empirical studies of software engineers [11, 37]. Our participants span a broad range of company sizes, roles, and experience levels, offering diverse perspectives and providing a reasonable basis for understanding developers' productivity perception in the context of GenAI adoption.

6 Concluding Remarks

GenAI tools are reshaping how developers engage with their work. Prior studies have demonstrated that these tools can accelerate task completion [25]. Yet our findings present a more complicated picture, that GenAI adoption has not yet produced the broad productivity gains that many organizations expect, but rather has led to "spurious productivity," a surface-level acceleration that obscures stagnant or redistributed effort.

6.1 The Redistribution of Effort

The concept of spurious productivity becomes evident when examining where developer effort shifts. Rather than eliminating work, developers described how AI tools redistributed it from code generation to downstream activities:

Debugging of AI outputs. Developers described trading manual coding time for cycles of prompting, verifying, and fixing AI-generated code. *"...any speed gained by code being produced faster is quickly lost when having to heavily scrutinize the generated code, and I have no interest in outsourcing the code I write to a AI tool only to become a professional code reviewer.[P161]"*

Increased code review burden. Our data revealed that the increased volume of AI-assisted output created a corresponding increase in review workload. Frequent AI users were more likely than non-frequent users to report conducting a greater number of code reviews, and most (84.3%) reported that GenAI did not reduce the time these reviews take. Participants explained that AI-generated code shifts cognitive burden to reviewers. *"...I spend more time reviewing code anyway because a couple teammates have adopted AI tools. I have to review their code much more closely now because the output of these tools is frankly not very good, and the people using them are not paying close enough attention to the output.[P161]"*

Cognitive load from constant output verification. Beyond measurable review time, developers also described a less visible

cost— the mental effort required to continuously evaluate AI suggestions. This cognitive load may explain why exhaustion levels remained high despite reported efficiency gains in task completion. *"In general, I welcome AI adoption, but the state of AI tools still produces very hit and miss outcomes and induces a ton of frustration.[P66]"*

6.2 Implications for Measuring Productivity

Our findings have important implications for evaluating GenAI adoption. While frequent AI users showed perceived increases in activity metrics such as more commits, test cases, and work items closed, performance metrics told a different story, with little improvement in test pass rates or learning velocity. Organizations that focus solely on activity-level measures may conclude that GenAI is effective, whereas those that assess performance outcomes may reach very different conclusions.

The disconnect recalls Fredrick Brooks' observation in the engineering management classic, *The Mythical Man-Month* [3], that there can be no single metric for programmer productivity, and that attempts to find one typically measure volume rather than performance. Organizations should use a holistic framework to evaluate productivity gains from AI.

6.3 SPACE as a Tool To Guide Practitioners

The key insight of this study is that effort saved in one SPACE dimension often resurfaces in another. When AI generates code faster (Activity), verification demands increase (Efficiency and flow). When AI produces more output (Performance), the greater review burden increases cognitive load (Satisfaction and well-being). Practitioners and tool designers who focus on accelerating a single dimension without addressing the others will find that total effort remains roughly constant; it simply moves to wherever constraints are weakest.

Our work suggests that SPACE could provide a planning framework for this constraint problem. Consider a team adopting AI-assisted code generation. Rather than deploying the tool and measuring commits, the team could work backward through the SPACE dimensions to identify where effort will likely shift, then address those areas proactively. For example, before using AI to generate features, teams could use AI to create comprehensive test suites and design documentation to thoroughly constrain and validate the output. This investment in Performance artifacts might reduce the cognitive load required to verify AI-generated code later.

In **conclusion**, developers report only modest productivity gains from GenAI across *SPACE*, with most dimensions showing little change and some improvements reflecting trade-offs or *spurious* productivity. We also provide an empirical mapping of developers' perceived challenges and potential strategies, offering actionable guidance for AI-infused workflows. Future work will complement these insights with repository mining and objective productivity signals to deepen the understanding of sustainable productivity under GenAI adoption.

References

- [1] Anonym. 2025. Developer Productivity with GenAI – Appendix. doi:10.5281/zenodo.18437751

- [2] Joel Becker, Nate Rush, Elizabeth Barnes, and David Rein. 2025. Measuring the impact of early-2025 AI on experienced open-source developer productivity. *arXiv preprint arXiv:2507.09089* (2025).
- [3] Frederick P Brooks Jr. 1995. *The mythical man-month: essays on software engineering*. Pearson Education.
- [4] Ana Casic and Eri Panselina. 2025. Quiet cracking: The hidden crisis silently reshaping work. <https://www.talentlms.com/research/quiet-cracking-workplace-survey>
- [5] Eleanor W Dillon, Sonia Jaffe, Nicole Immorlica, and Christopher T Stanton. 2025. *Shifting work patterns with generative ai*. Technical Report. National Bureau of Economic Research.
- [6] Thomas Dohmke, Marco Iansiti, and Greg Richards. 2023. Sea change in software development: Economic and productivity analysis of the ai-powered developer lifecycle. *arXiv preprint arXiv:2306.15033* (2023).
- [7] Zixuan Feng, Sadia Afroz, and Anita Sarma. 2025. From Gains to Strains: Modeling Developer Burnout with GenAI Adoption. *arXiv preprint arXiv:2510.07435* (2025).
- [8] Zixuan Feng, Amreeta Chatterjee, Anita Sarma, and Iftekhar Ahmed. 2022. A case study of implicit mentoring, its prevalence, and impact in Apache. In *ESEC/FSE*. 797–809.
- [9] Zixuan Feng, Reed Milewicz, Emerson Murphy-Hill, Tyler Menezes, Alexander Serebrenik, Igor Steinmacher, and Anita Sarma. 2025. Charting Uncertain Waters: A Socio-Technical Framework for Navigating GenAI's Impact on Open Source Communities. *arXiv preprint arXiv:2508.04921* (2025).
- [10] Zixuan Feng, Igor Steinmacher, Marco Gerosa, Tyler Menezes, Alexander Serebrenik, Reed Milewicz, and Anita Sarma. 2025. The multifaceted nature of mentoring in oss: strategies, qualities, and ideal outcomes. In *CHASE*. IEEE, 203–214.
- [11] Zixuan Feng, Thomas Zimmermann, Lorenzo Pisani, Christopher Gooley, Jeremiah Wander, and Anita Sarma. 2025. When Domains Collide: An Activity Theory Exploration of Cross-Disciplinary Collaboration. *arXiv preprint arXiv:2506.20063* (2025).
- [12] Nicole Forsgren, Margaret-Anne Storey, Chandra Maddila, Thomas Zimmermann, Brian Houck, and Jenna Butler. 2021. The SPACE of Developer Productivity: There's more to it than you think. *Queue* 19, 1 (2021), 20–48.
- [13] Ranim Khojah, Mazen Mohamad, Philipp Leitner, and Francisco de Oliveira Neto. 2024. Beyond code generation: An observational study of chatgpt usage in software engineering practice. *FSE* 1 (2024), 1819–1840.
- [14] Mohammad Amin Kuhail, Sujith Samuel Mathew, Ashraf Khalil, Jose Berengueres, and Syed Jawad Hussain Shah. 2024. "Will I be replaced?" Assessing ChatGPT's effect on software development and programmer perceptions of AI tools. *Science of Computer Programming* 235 (2024), 103111.
- [15] J Richard Landis and Gary G Koch. 1977. The measurement of observer agreement for categorical data. *biometrics* (1977), 159–174.
- [16] André N Meyer, Thomas Fritz, Gail C Murphy, and Thomas Zimmermann. 2014. Software developers' perceptions of productivity. In *FSE*. 19–29.
- [17] Microsoft Corporation. 2024. AI at Work Is Here. Now Comes the Hard Part. <https://www.microsoft.com/en-us/worklab/work-trend-index/ai-at-work-is-here-now-comes-the-hard-part>. Accessed: 2026-01-19.
- [18] Courtney Miller, Rudrajit Choudhuri, Mara Ulloa, Sankeerti Haniyur, Robert DeLine, Margaret-Anne Storey, Emerson Murphy-Hill, Christian Bird, and Jenna L Butler. 2025. "Maybe We Need Some More Examples." Individual and Team Drivers of Developer GenAI Tool Use. *arXiv preprint arXiv:2507.21280* (2025).
- [19] Audris Mockus, Roy Fielding, and James Herbsleb. 2002. Two case studies of open source software development: Apache and Mozilla. *TOSEM* 11, 3 (2002).
- [20] Sergio Moreschini, Elvira-Maria Arvanitou, Elisavet-Persefoni Kanidou, Nikolaos Nikolaidis, Ruoyu Su, Apostolos Ampatzoglou, Alexander Chatzigeorgiou, and Valentina Lenarduzzi. 2026. The Evolution of Technical Debt from DevOps to Generative AI: A multivocal literature review. *Journal of Systems and Software* 231 (2026), 112599.
- [21] Anh Nguyen-Duc, Beatriz Cabrero-Daniel, Adam Przybylek, et al. 2025. Generative AI for software engineering: A research agenda. *Softw. Pract. Exp.* (2025).
- [22] Abi Noda, Margaret-Anne Storey, Nicole Forsgren, and Michaela Greiler. 2023. DevEx: What Actually Drives Productivity: The developer-centric approach to measuring and improving productivity. *Queue* 21, 2 (2023), 35–53.
- [23] Edson Oliveira, Eduardo Fernandes, Igor Steinmacher, Marco Cristo, Tayana Conte, and Alessandro Garcia. 2020. Code and commit metrics of developer productivity: a study on team leaders perceptions. *EMSE* 25, 4 (2020), 2519–2549.
- [24] Elise Paradis, Kate Grey, Quinn Madison, et al. 2025. How much does AI impact development speed? An enterprise RCT. In *Proc. ICSE-SEIP*. 618–629.
- [25] Sida Peng, Eirini Kalliamvakou, Peter Cihon, and Mert Demirer. 2023. The Impact of AI on Developer Productivity: Evidence from GitHub Copilot. *arXiv preprint arXiv:2302.06590*.
- [26] Ketai Qiu, Niccolò Puccinelli, Matteo Ciniselli, and Luca Di Grazia. 2025. From today's code to tomorrow's symphony: The AI transformation of developer's routine by 2030. *TOSEM* 34, 5 (2025), 1–17.
- [27] Ya Gao & GitHub Customer Research. 2024. Research: Quantifying GitHub Copilot's impact in the enterprise with Accenture. <https://github.blog/news-insights/research/research-quantifying-github-copilots-impact-in-the-enterprise-with-accenture/> Accessed: 2026-02-02.
- [28] Daniel Rodriguez, MA Sicilia, E Garcia, and Rachel Harrison. 2012. Empirical findings on team size and productivity in software development. *Journal of Systems and Software* 85, 3 (2012), 562–570.
- [29] Mario Rodriguez. 2023. Research: Quantifying GitHub Copilot's impact on code quality. <https://github.blog/news-insights/research/research-quantifying-github-copilots-impact-on-code-quality/>. Accessed: 2026-02-02.
- [30] Alan Shimmel. 2025. Stack Overflow Survey Shows AI Adoption for Devs. *DevOps.com* (Aug. 2025). <https://devops.com/stack-overflow-survey-shows-ai-adoption-for-devs/> Accessed: 2025-10-05.
- [31] S.J. Steinhart. 2024. Survey: AI Tools Can Boost Productivity by Reducing Time Spent in Meetings. <https://www.nysscpa.org/article-content/survey-ai-tools-can-boost-productivity-by-reducing-time-spent-in-meetings-041024#sthash.M96q7cw8.dpbs>. Accessed: 2026-01-19.
- [32] Margaret-Anne Storey, Brian Houck, and Thomas Zimmermann. 2022. How developers and managers define and trade productivity for quality. In *CHASE*. 26–35.
- [33] Margaret-Anne Storey, Thomas Zimmermann, Christian Bird, Jacek Czerwinka, Brendan Murphy, and Eirini Kalliamvakou. 2019. Towards a theory of software developer job satisfaction and perceived productivity. *TSE* 47, 10 (2019).
- [34] Franziska Tobisch and Florian Matthes. 2025. Knowledge Sharing and Coordination in Large-Scale Agile Software Development—A Systematic Literature Review and an Interview Study. In *International Conference on Agile Software Development*. Springer, 81–99.
- [35] Anna Tong. 2025. AI slows down some experienced software developers, study finds. <https://www.reuters.com/business/ai-slows-down-some-experienced-software-developers-study-finds-2025-07-10/>. Accessed: 2025-02-02.
- [36] Bianca Trinkenreich, Fabio Santos, and Klaas-Jan Stol. 2024. Predicting attrition among software professionals: Antecedents and consequences of burnout and engagement. *TOSEM* 33, 8 (2024), 1–45.
- [37] Bianca Trinkenreich, Klaas-Jan Stol, Anita Sarma, Daniel M German, Marco A Gerosa, and Igor Steinmacher. 2023. Do i belong? modeling sense of virtual community among linux kernel contributors. In *ICSE*. IEEE, 319–331.
- [38] University of North Carolina at Chapel Hill. [n. d.]. Staff Generative AI Usage Guidance. <https://ai.unc.edu/staff-generative-ai-usage-guidance/> Accessed: 2026-01-19.
- [39] Priyan Vaithilingam, Tianyi Zhang, and Elena L Glassman. 2022. Expectation vs. experience: Evaluating the usability of code generation tools powered by large language models. In *CHI EA*. 1–7.
- [40] Bogdan Vasilescu, Yue Yu, Huaimin Wang, Premkumar Devanbu, and Vladimir Filkov. 2015. Quality and productivity outcomes relating to continuous integration in GitHub. In *ESEC/FSE*. 805–816.
- [41] Brennan Wilkes, Alessandra Maciel Paz Milani, and Margaret-Anne Storey. 2023. A framework for automating the measurement of devops research and assessment (dora) metrics. In *ICSM*. IEEE, 62–72.
- [42] Anita Williams Woolley. 2025. Generative AI and collaboration: opportunities for cultivating collective intelligence. *Journal of Organization Design* (2025), 1–6.
- [43] Liang Yu. 2025. Paradigm shift on Coding Productivity Using GenAI. *arXiv preprint arXiv:2504.18404* (2025).
- [44] Ilya Zakharov, Ekaterina Koshchenko, and Agnia Sergeyuk. 2025. AI in Software Engineering: Perceived Roles and Their Impact on Adoption. In *FSE*. 1305–1309.
- [45] Minghui Zhou and Audris Mockus. 2010. Developer fluency: Achieving true mastery in software projects. In *FSE*. 137–146.
- [46] Albert Ziegler, Eirini Kalliamvakou, X Alice Li, Andrew Rice, Devon Rifkin, Shawn Simister, Ganesh Sittampalam, and Edward Aftandilian. 2022. Productivity assessment of neural code completion. In *MAPS*. 21–29.