Software Developers' Perceptions of Productivity: An Industry-focused Study

Murilo Coelho
Atlântico Institute
State University of Ceará
Fortaleza, Ceará, Brazil
paulo_coelho@atlantico.com.br

Isabelle Reinbold
Atlântico Institute
Fortaleza, Ceará, Brazil
isabelle_reinbold@atlantico.com.br

Lizie Sancho
Atlântico Institute
Fortaleza, Ceará, Brazil
lizie_sancho@atlantico.com.br

Matheus Paixao
State University of Ceará
Fortaleza, Ceará, Brazil
matheus.paixao@uece.br

Allysson Allex Araújo
Federal University of Cariri
Juazeiro do Norte, Ceará, Brazil
allysson.araujo@ufca.edu.br

Sávio Freire
Federal Institute of Ceará
Morada Nova, Ceará, Brazil
savio.freire@ifce.edu.br

ABSTRACT

Context: Developer productivity is a critical factor of software project success and, by extension, a key driver of organizational performance and software quality in software engineering. In the era of Generative Artificial Intelligence (GenAI), understanding the developers' perception of productivity becomes paramount for a software organization seeking to adopt GenAI tools. **Objective**: This paper aims to examine developers' perception of productivity, focusing on its measurement and the expected impact of adopting GenAI. **Method**: We conducted two semi-structured focus groups with software developers from a large science and technology institute. We also administered a confirmatory questionnaire to support the verification of our findings. The collected data were analyzed using content analysis. **Results**: We identified five key productivity factors: quality, expectations, deadlines, progress, and efficiency, including distinct measurement systems for coding, code review, and documentation activities. Participants consistently emphasized the potential of GenAI to streamline and enhance a range of routine development tasks, albeit with certain limitations regarding their use. **Conclusion**: Our findings indicate that productivity constitutes a complex multidimensional construct influenced by interrelated factors. Moreover, GenAI emerge as a transformative technology whose impacts on productivity require deeper investigation in future studies, particularly regarding balancing efficiency gains with the preservation of technical abilities.

KEYWORDS

Productivity, Industry, Large Language Models, Qualitative Study.

1 Introduction

Software development productivity represents a relevant factor in the success of modern organizations, directly influencing software quality, projects outcomes, team effectiveness, and organizational competitiveness [17, 31, 37, 38]. As digital transformation accelerates across industries, the ability to develop high-quality software efficiently has become a strategic need for organizations seeking to gain market relevance and drive innovation [4, 23].

However, productivity in software engineering (SE) remains a complex and multifaceted construct, which is challenging to define and to measure consistently in different contexts [11, 12, 37]. Recently, the rise of Generative Artificial Intelligence (GenAI), particularly Large Language Models (LLMs), has introduced a potentially disruptive force into software development workflows [5, 16]. These tools assist with coding, debugging, documentation, and other complex tasks, reshaping how developers allocate effort and perceive productivity [9, 28].

In Brazil's Research and Technology Organizations (RTOs), understanding and optimizing developer productivity takes on heightened importance, as they differ from typical software companies in mission, constraints, and evaluation criteria [1]. While many commercial firms focus primarily on speed, scalability, and market delivery, Brazil's RTOs operate under a hybrid mission. They must balance the goal of advancing scientific knowledge through applied research (producing outputs like technical reports and proofs of concept) with the demand for delivering practical software solutions for their partners [2]. This dual mandate introduces challenging and often conflicting productivity expectations. Moreover, developers in these settings frequently face institutional constraints, such as limited infrastructure and bureaucratic delays, which directly shape their perceptions of productivity and influence a more cautious stance toward the adoption of new technologies like GenAI.

While several empirical studies have explored productivity in SE [6, 14, 18, 34], much of the literature still emphasizes broad trends across heterogeneous developer populations, often relying on large-scale surveys. However, a growing body of work is beginning to adopt qualitative and case study approaches to capture more nuanced findings, including studies with focused samples [3, 8]. Despite this progress, the subjective and context-specific interpretations of productivity (particularly in relation to GenAI support within dual research-technology (R&T) environments) remain underexplored. In contexts such as Brazil's RTOs, where scientific inquiry intersects with industrial collaboration, productivity assumes distinct dimensions that merit deeper investigation. In particular, addressing this issue requires a closer examination of developers' perceptions and lived experiences [29, 36].

Conducted in the Atlântico Institute (a large Brazilian RTO currently evaluating the integration of GenAI tools), this study aims to investigate how software developers perceive productivity in their day-to-day work and how they anticipate the impact of GenAI on their workflows. Atlântico currently employs over five hundred professionals distributed nationwide, predominantly operating under a remote work model. The novelty of this exploratory and empirical study lies in its focus on an underexplored setting (Brazilian RTO), the use of focus groups to uncover lived experiences, and the mapping of results to the SPACE Framework [13], a multidimensional model for understanding software productivity. By anchoring our analysis in the SPACE, we may enhance the theoretical clarity and practical interpretability of our findings.

This study involved two focus groups with 16 developers, followed by a confirmatory questionnaire to validate our findings. Hence, the industry focused nature of this research provides findings for organizations similarly positioned at the intersection of R&T. In summary, our qualitative content analysis revealed that developers' productivity perceptions revolve around five elements: `progress/evolution`, `expectations`, `time/deadlines`, `quality`, and `efficiency`. These perceptions differ across development activities (such as coding, reviewing, and documenting) each associated with distinct indicators. With respect to GenAI, developers noted expected benefits (e.g., `reduced time`, `support for writing code and documentation`) but also voiced concerns (e.g., `hallucinations`, `code duplication`, `reduced critical thinking`, and `debugging overhead`).

This study offers contributions to both academic research and industry practice by examining software productivity through the rarely explored lens of a Brazilian RTO. First, it provides empirical evidences into how developers within this RTO environment perceive and articulate productivity. Second, it maps the identified productivity perceptions, incorporating challenges and expectations related to GenAI, onto the dimensions of the SPACE framework. Third, it proposes a conceptual map to help organizations evaluate the trade-offs of GenAI adoption, supporting teams in balancing efficiency gains with sustained software quality.

The remainder of this paper is organized as follows. Section 2 presents the background on productivity and related work. Section 3 describes the research method. Section 4 reports the results. Section 5 discusses our findings, including their mapping to the SPACE Framework and the conceptual map. Section 6 addresses the trustworthiness of this study. Finally, Section 7 concludes the paper and outlines directions for future work.

2 Background

Despite productivity being a widely explored field, defining it within the software development industry remains a challenge because it relates to "knowledge work," since its outputs are mainly information [10]. As such, how can a metric encompass mental and physical processes? How can a piece of knowledge or collaboration be measured?

Jaspan and Sadowski [19] described productivity as a broad concept, in which a single metric cannot capture. By employing a single productivity measure, one is likely to exclude important factors. An example would be using a developer's velocity as single productivity proxy as it hides fundamental aspects of a developers' work, such as mentoring activities and code quality. In general, productivity is defined as the ratio between the outputs and the inputs. In this way, it is not what one gains after subtracting the costs (profit), it means how much effort and investment one needs to create an expected output [32].

When considering the software industry, Wagner and Deissenboeck [37] explained that productivity inputs are related to the effort to develop and maintain a software, as outputs are associated with the value it brings for the users, which can vary based on the market and its competitors. In spite of all the difficulties, many researchers have been studying frameworks, metrics, techniques, and productivity drivers for several years. This effort is naturally justified by several benefits, such as the ones pointed out by Jaspan and Sadowski [19]: (i) to indicate global trends in an organization's performance, (ii) to demonstrate the effectiveness of tools and/or practices, (iii) to be used on interventions to compare productivity results, and (iv) to point out areas that need improvement.

To discuss developers' perception of productivity from Atlântico Institute, we elected to employ the holistic and multidimensional SPACE framework [13], recognized for its popularity and practical method to choose the most suitable metrics for different productivity aspects. SPACE is an acronym for the following dimensions that influence productivity: **S**atisfaction and well-being; **P**erformance; **A**ctivity; **C**ommunication and collaboration; and **E**fficiency and flow. For each dimension, Forsgren et al. [13] presented a list of metrics that can be used to evaluate individuals and teams. To understand productivity, SPACE recommends the selection of multiple metrics from at least two dimensions, ideally three, because their tension can enable a more insightful analysis. Next, one should add at least one perceptual metric to capture how developers view their work. In Section 5, we discuss the insights and conclusions of this study through the lens of the SPACE framework.

Exploring existing research on software development productivity perceptions is important to position our study under the current state of knowledge. For example, Meyer et al. [26] investigated developers' perceptions of productive and unproductive work through two studies. An initial online survey of 379 developers revealed that a productive day is generally characterized by task progression or completion with minimal interruptions. The study also explored potential productivity metrics. Building on the survey findings, an observational study of 11 developers across three companies further examined the definition of a task, the outcomes of productive work in various situations, and the nature of uninterrupted work.

A different research stated that productivity perception varies greatly among individuals and is influenced by numerous factors [25]. This study has identified common trends related to working habits and computer interactions. The study indicated developers often fall into 'morning', 'low-at-lunch', or 'afternoon' productivity patterns, suggesting a personal rhythm to their work. Furthermore, a high volume of mouse and keyboard activity tends to correlate with a positive perception of productivity, whereas time spent on emails, scheduled meetings, and non-work-related websites is generally associated with a feeling of unproductivity.

When it comes to the perceived productivity with AI assistance, Glushkova [15] surveyed 150 practitioners in the software industry to understand the influence of ChatGPT on their activities. The participants concluded that it can support junior developers and amplify senior abilities, increase code velocity and quality, and improve processes like requirements gathering and project planning.

Beyond these findings, two additional qualitative studies reinforce the cited benefits. Banh et al. [3] further emphasized the importance of advanced reasoning capabilities. Their research, based on interviews with 17 engineers from European software development companies, detailed several challenges in adoption: limited integration, intellectual property concerns, underestimated overhead, and issues of reliability and dependency.

Coutinho et al. [8] examined the integration of GenAI tools in a large software organization with over 1,200 employees. Using a mixed-methods approach that included a survey of 13 developers and a four-week observation of internal communication channels, the study identified both the perceived benefits and the practical challenges of AI adoption in professional settings. Developers generally viewed generative AI tools as beneficial for improving individual productivity. However, they identified barriers, including concerns about the credibility and accuracy of AI-generated outputs, the difficulty of crafting effective prompts, and constraints related to data confidentiality in high-security projects.

Despite advancements in frameworks and empirical studies, software development productivity remains a highly complex construct. In this regard, the rise of GenAI tools adds new layers of opportunity and challenge. In contexts like Brazilian RTOs, where research and application intersect, understanding productivity demands a more context-aware approach. This study seeks to advance this evolving understanding, offering empirical findings into how developers perceive productivity and anticipate the impact of GenAI in a relevant yet underexplored setting.

3 Research Method

This section outlines the research questions that guided this work, as well as the data collection and analysis methods used.

3.1 Research Questions

Our goal is to examine developers' perception of productivity, focusing on its measurement and the expected impact of adopting GenAI. We split it into the following research questions (RQs):

- **RQ1: What is the perceived productivity of software developers?** This question investigates the main factors that lead developers to feel productive when performing their activities.
- **RQ2: How do software developers measure productivity within the software development lifecycle?** This question investigates how developers assess productivity in coding, code review, and documentation activities. We chose these activities because they are typically performed together in a developer's daily tasks.
- **RQ3: How is GenAI expected to affect the productivity of software developers?** Given the growing use of GenAI in software development tasks, this question seeks to identify the expected impact of GenAI on developer productivity.

3.2 Data Collection

In an exploratory research scope, we conducted online focus groups to capture qualitative data through group interactions [33]. This interactive approach allowed participants to build on each other's responses, generating richer insights than individual interviews, especially for exploring nuanced concepts like productivity, where meaning is socially constructed through dialogue [22, 24].

The focus groups were conducted with software developers from the Atlântico Institute. It specializes in applied research, technological development, and innovation. While headquartered in Fortaleza, capital of the state of Ceará, it maintains international operations in the information technology sector, with core expertise in four strategic domains: applied artificial intelligence, virtual and immersive reality systems, intelligent automation systems, and distributed architectures and connectivity solutions. Currently employing over five hundred professionals distributed nationwide, the organization predominantly operates under a remote work model. The participants in our study were affiliated with several of these organizational units, which are structured according to the clients they serve, thus covering a diverse range of project domains. Due to confidentiality agreements, we cannot disclose specific unit or project names. We purposefully selected this institute because of its interest in incorporating GenAI tools into its workflows.

To conduct the focus group sessions, we defined a semi-structured interview script consisting of three primary questions and ten follow-up questions, as shown in Table 1. Each question was carefully mapped to our RQs to ensure comprehensive coverage of the research domain. For instance, the question "*From your perspective, how do you define productivity in the performance of a software developer?*" directly addressed RQ1, while follow-up questions such as "*What makes you feel productive on a daily basis?*" provided a deeper exploration of personal productivity experiences. Questions related to specific tasks such as coding, code review, and documentation (Q2.1–2.4) were designed to address RQ2's focus on measuring productivity across different development activities. The final section of questions (Q3) explored the potential impacts of GenAI, addressing RQ3 through inquiries about usage patterns, perceived benefits, challenges, and effects on collaboration.

To select participants, we employed purposive sampling, a common qualitative research approach focused on achieving diversity of perspectives rather than statistical representativeness. To ensure this diversity, we asked project managers at the Atlântico Institute to nominate developers from different software projects and organizational units, with varying experience levels, and to avoid hierarchical dependencies that could limit open discussion. Availability and interest in the study were also considered. This recruitment strategy aligns with the recommended practices for focus group studies in SE, where the emphasis is on depth and diversity of perspectives, rather than sample size. We also explained the research goal, focus group format, and ethical considerations, emphasizing that participation was voluntary and aimed at contributing to the organization's understanding of productivity perceptions. All participants provided informed consent through a form, acknowledging their right to withdraw at any time without consequence.

In total, we had 16 participants, whom we divided into two focus groups (seven and nine members) to ensure better interaction. Both groups represented a diverse cross-section of developers from different teams within Atlântico Institute. Despite scheduling challenges, participants were selected with diversity in mind.

Each focus group was scheduled for 120 minutes and facilitated by three researchers: a primary moderator who guided the discussion, and two secondary moderators who took notes and managed recordings. The sessions were conducted in an online meeting room

Table 1: Script of the focus group questions.

RQ	Question	ID	Follow up questions
RQ1	From your perspective, how do you define productivity in the work of a software developer?	1.1	What makes you feel productive on a daily basis? Can you give an example?
		1.2	What do you think affects or impacts your productivity?
		1.3	How do you measure or perceive your own productivity? Are there any indicators you consider?
RQ2	From your perspective, how do you define productivity on your day-to-day tasks?	2.1	In what circumstances is writing code productive? Is there a metric or indicator that reflects this perception?
		2.2	In what circumstances is code review productive? Is there a metric or indicator that reflects this perception?
		2.3	In what circumstances is writing documentation productive? Are there any metrics or indicators that reflect this perception?
		2.4	What other tasks do you perform on a daily basis? How productive do you feel in these tasks? Are there any metrics or indicators?
RQ3	What do you think about the impact of GenAI on your productivity?	3.1	What do you think are the greatest benefits of GenAI for a developer's productivity?
		3.2	Do you consider any negative impacts of using GenAI tools for software development?
		3.3	If you could sum up the impact of AI on your productivity in one sentence or one word, what would it be?

via Google Meet in November 2024, creating an environment where participants felt comfortable sharing their perspectives.

In line with best practices, the focus groups followed a structured yet flexible approach [24]. The sessions were designed to foster dialogue among participants, not just between the moderator and individuals, which is the hallmark of the focus group method. We began with introductions and a brief explanation of the research goal, followed by obtaining verbal confirmation of informed consent. An introductory round allowed participants to become comfortable speaking in a group setting. The main discussion then progressed through our semi-structured protocol, with the moderator using open-ended questions to encourage participants to respond to and build upon each other's ideas. This interactive dynamic is what differentiates a focus group from a collective interview and is aligned with accepted practices, even when conducted remotely due to the Atlântico distributed work model. The moderator employed probing techniques when necessary to elicit deeper reflections, particularly when discussing abstract concepts like productivity perception. Our moderation strategy prioritized open dialogue and collective reflection, not forced consensus; divergent views were allowed to surface and were preserved as valuable data. The other moderators documented these group dynamics, capturing moments of both natural convergence and contrasting viewpoints. Following each session, all moderators conducted a debriefing to document initial impressions and emerging themes. This reflection helped capture fresh findings before they were diluted over time and provided a preliminary direction. All sessions were recorded with participants' consent and subsequently prepared for detailed analysis.

3.3 Data Analysis

All focus group recordings were transcribed verbatim using Google Colab with Whisper, a general-purpose speech recognition model developed by OpenAI. The accuracy of the transcriptions was verified through a double-checking process conducted by two of the authors. For the data analysis, we employed content analysis [21] following an open coding process inspired by Strauss et al. [35]. The process was designed to be inductive, ensuring no predefined codes imposed on the data. First, two researchers independently read the transcripts and identified "recording units", i.e, short text segments relevant to the research questions. These units were stored and reviewed in a shared spreadsheet. Subsequently, the researchers held iterative discussion meetings to compare, consolidate, and refine the codes derived from these units. Divergences in interpretation were resolved through these consensus meetings. This systematic and inductive approach allowed the categories and themes presented in our findings to emerge directly from the participants' discourse.

Next, the same researchers manually and independently coded the recording units to identify codes related to productivity perceptions (RQ1), measurement approaches (RQ2), and the impacts of GenAI (RQ3). Divergences in coding were addressed through a consensus meeting between the researchers. For example, in the recorded unit: "*When you manage to spend effort on that task that you have as an objective, let's say there, daily, those goals of yours, and you manage to walk, you manage to get out of there according to what you expected, or better than the effort*," one researcher coded it as `expectation`, while the other annotated `expectation` and `progress`. After the consensus meeting, they decided to maintain `expectation` and `progress`, as the recorded unit addresses both the expectation of finishing an activity and the progress towards the goal. Another example, the recording unit "*A productive day is when I make progress on something and hit or exceed the exact level I was aiming for*" was coded as `progress/evolution`, as it describes how the developer perceives a productive day.

During the coding process, the researchers observed that software developers referred to both productive and unproductive factors when describing their perceptions of productivity in their work. Similarly, they mentioned both expected benefits and potential drawbacks associated with the use of GenAI to support their activities. These different and complementary perspectives contribute to a more comprehensive understanding of productivity from the developers' point of view.

3.3.1 Reporting. Our findings are reported in alignment with the three research questions, presenting both the commonalities and variations in developers' perspectives. For each research question,

Table 2: Focus Groups Characterization.

ID	Seniority	*Experience*[a]	*Frequency*[b]	*Use?*[c]
P1-FG1	Senior or more	More 10	Very often	Yes
P2-FG1	Mid-level	Between 1-3	Very often	Yes
P3-FG1	Senior or more	More 10	Very often	No
P4-FG1	Senior or more	More 10	Eventually	Yes
P5-FG1	Senior or more	Between 1-3	Eventually	Yes
P6-FG1	Junior	Between 1-3	Often	Yes
P7-FG1	Senior or more	Between 1-3	Often	Yes
P1-FG2	Senior or more	Between 4-6	Often	No
P2-FG2	Mid-level	Between 4-6	Often	Yes
P3-FG2	Mid-level	Between 4-6	Rarely	No
P4-FG2	Mid-level	Between 4-6	Rarely	No
P5-FG2	Senior or more	Between 1-3	Rarely	Yes
P6-FG2	Mid-level	Between 1-3	Eventually	No
P7-FG2	Senior or more	Between 4-6	Eventually	No
P8-FG2	Senior or more	Between 7-10	Eventually	No
P9-FG2	Mid-level	Between 4-6	Often	Yes

Legend:
[a] Years of experience as a software developer in the Atlântico Institute
[b] Weekly usage frequency of GenAI systems
[c] Uses GenAI in their daily work in the Atlântico Institute

we describe the codes that emerged from our analysis, supported by representative quotes from participants. We maintain participant anonymity through the use of pseudonyms (e.g., P1-FG1 for Participant 1 from Focus Group 1), ensuring ethical research practices while providing the necessary context for quote interpretation.

3.3.2 Results Cross-Checking. To ensure the credibility and confirmability of our findings, we developed a confirmatory questionnaire aimed at validating the results, as presented in our public repository [7]. The questionnaire consisted of six demographic questions and sixteen items specifically addressing the study's main findings. For example, the question "Initially, the concept of productivity was discussed. The following factors were identified as having a positive impact on productivity: *(... results ...)*. Do you agree that these factors were discussed during the focus group you attended?" aimed to verify whether the participant agreed with the factors identified in the focus groups. Additionally, we provided an opportunity for participants to point out any inconsistencies in our findings, as illustrated by the question: "If not, what factors were not discussed?" The questionnaire was distributed to all 16 software developers who participated in the focus group sessions and distributed approximately six months after. This interval was not planned from the outset but emerged naturally from the time required for data transcription, coding, and analysis, and also coincided with a period of increased GenAI adoption within the Atlântico Institute. It is important to note that we did not collect new data through the questionnaire, but rather limited its use to validating the findings.

4 Results

Table 2 summarizes the key demographic characteristics of the participants, ensuring their anonymity.

The seniority levels were classified based on participants' self-reported roles and years of experience. Labels such as "Junior," "Mid-level," and "Senior or more" reflect the internal job titles and the nature of responsibilities within the Atlântico Institute. Most of the developers hold "senior-level" positions or above, followed by "mid-level" and "junior" roles. Regarding experience in the Atlântico Institute, the majority of participants reported having between 1 and 3 years of experience, followed by those with 4 to 6 years, more than 10 years, and 7 to 10 years. Most participants reported using GenAI tools with a frequency of "eventually" (1 to 2 times a week), followed by "often" (3 to 5 times a week), and with the same count, "rarely" (less than once a week) and "very often" (more than once a day). The gender distribution revealed a predominance of male participants (80%), with 20% identifying as female. One participant preferred not to disclose their gender. The average age of developers at the Atlântico Institute was found to be 33 years. Regarding the use of GenAI in work activities, most developers in FG1 reported having used it, whereas in FG2, there was a balance, with nearly equal numbers of developers who use and do not use the tools. The following subsections present the results for each RQ.

4.1 RQ1: What is the Perceived Productivity of Software Developers?

To address this research question, we identified key factors that make developers feel productive or unproductive. We begin by presenting the factors that contribute to productivity. We identified the following factors that contribute to software developers' productivity: `progress/evolution` (23 mentions), `expectations` (21), `time/deadline` (15), `quality` (9), and `efficiency` (6).

In the context of this study, `progress/evolution` refers to the ability to maintain continuity in software development despite inherent process challenges, as we can see in: "*A productive day is when I make progress on something and hit or exceed the exact level I was aiming for*" (P3-FG1). `Expectations`, in turn, are understood as the capacity to meet projected needs (whether from stakeholders or oneself), as illustrated in: "*Whether I'm being productive really depends on what the team expects, what the client wants, and how we handle our sprints*" (P6-FG2). `Time/deadline` corresponds to the predetermined time frame for completing planned activities within the development cycle, as evidenced in: "*We've got this really hands-on way to measure productivity - we track what demands get done within their timeboxes*" (P1-FG2). `Quality` refers to the correct implementation of functionalities in compliance with both functional and non-functional requirements, as demonstrated in: "*Happy I delivered something, but I'd still feel unproductive knowing I could've delivered better quality work*" (P1-FG1). Finally, `efficiency` is defined as the adoption of practices that optimize processes and minimize waste during development, as shown in: "*I see productivity as delivering your tasks in a smart way and then optimizing your time*" (P5-FG2).

Finding #1: Participants identified `progress/evolution`, `expectations`, `time/deadline`, `quality`, and `efficiency` as key factors contributing to productivity, with `progress/evolution` emerging as the most frequently cited.

We also identified the following key factors that make developers feel unproductive: `performing repetitive tasks`, `context switching`, `non-coding activities`, and `unclear requirements`.

More specifically, `performing repetitive tasks` refers to writing code with identical structures without incorporating new elements, as evidenced in: "*There are very repetitive processes we end up doing, not exactly copying and pasting code. [. . .] The process is always the same. You always write the same things in the same places. Not exactly the same things because property names change, table names vary, but having to repeat that is very tedious. It makes me feel unproductive. I'm working hard for little results*"(P4-FG2). In addition, `context switching` refers to constantly alternating between multiple tasks, impairing concentration, as exemplified in: "*I'm greatly affected by context switching. I generally perform better when I take one activity, focus on it until completion, and only then move to another task. I don't know if it affects everyone the same way, but it really impacts me*"(P5-FG1).

`Non-coding activities` is related to periods when developers are engaged in tasks that do not involve actual coding, as illustrated in: "*Things that block our work . . . On those days I end up feeling unproductive because I have tasks to do, but various project factors prevent me. So I'll take on another activity or do something else to feel better, to feel productive, so to speak*" (P6-FG1). It is important to note that software developers often perform non-coding activities, and considering these activities as unproductive is an individual choice. Finally, `unclear requirements` is a lack of demand specificity, requiring additional interpretation effort from developers, as show in: "*A lack of definition in tasks, or lack of clarity about things, these are what prevent you from working more smoothly*" (P6-FG1).

Finding #2: Participants identified `performing repetitive tasks`, `context switching`, `unclear requirements`, and `non-coding activities` as the key factors that make developers feel unproductive.

Considering the factors that make developers feel productive or unproductive, it becomes evident that the concept of productivity constitutes a multidimensional construct, shaped by systematic interactions among five key elements: code quality, adherence to deadlines, fulfillment of multifaceted expectations (personal, team, and client), perceived progress, and efficiency. Conversely, repetitive tasks, context switching, unclear requirements and noncoding workload emerge as detractors of perceived productivity. This understanding is exemplified in: "*Sometimes we had this feeling of productivity, but at the same time we didn't. We were meeting, solving, understanding—everything was new: the technology was new, the requirements were being discovered as the project progressed. So sometimes we didn't have any deliverables, but we made advances in our understanding of the system. On the other hand, the client had demands—they wanted a deliverable at the end of the day. So we had this mixture of 'I was productive' and at the same time 'I wasn't productive,' because while we had gained understanding, the client still expected something tangible. So I think there's this duality*" (P7-FG2).

Finding #3: Aggregating the key factors, we summarize RTO developers' perceptions of productivity as:

"`The perception of productivity is experienced by developers when some progress, whether quantitative or qualitative, is made in writing code that meets quality criteria and requirements, as well as personal, team, and client expectations within the stipulated deadline. This feeling can be negatively impacted by performing repetitive tasks, context switching, unclear requirements or performing non-code activities.`"

4.2 RQ2: How do Software Developers measure Productivity within the Software Development Lifecycle?

We identified distinct productivity indicators organized by core activities: coding, code review, and documentation.

Regarding **coding activities**, software developers defined indicators related to multiple software quality attributes simultaneously, such as `reusability` and `readability`, as we can see, respectively, in: "*I feel productive when I successfully reuse particular code, when the work avoids becoming repetitive. I know that component will be valuable at different stages, creating tangible impact. This enables me to develop generalized solutions that can be effectively repurposed*" (P1-FG1) and "*I make sure the final code is super clear: I use meaningful names for functions, keep each function short and to the point. The code ends up being smooth to read and easy to grasp. That's what matters most: whether someone else can pick it up and understand what was done without struggling*" (P5-FG1).

They also identified `code optimization` and `functionality` as relevant indicators for assessing their coding productivity, as described in: "*There are other cases that'll need this optimization right here, because it'll directly impact the code. Responsibility is this system's core characteristic - it's one of its foundations*" (P4-FG1) and "*Is actual working implementation. When I write the code and verify that my proposed solution - that specific algorithm - is truly functioning as intended*" (P1-FG2), respectively.

Lastly, `meeting established software requirements` was cited as another productivity indicator, as evidenced in: "*Every task has its requirements, right? When you write code that perfectly meets what the task needs, you can call that productivity. The key is keeping tasks as small as possible - that way you can better track and measure exactly how much time you spent on each*" (P5-FG2).

Finding #4: Participants considered source code attributes (such as `code reusability`, `readability`, `optimization`, and `functionality`) as key indicators when assessing their coding productivity. Notably, they also look beyond source code characteristics by viewing `meeting established software requirements` as an additional indicator of productivity.

The findings on **code review** productivity revealed diverse perspectives. The developers considered `learning in the process` as an indicator, as exemplified in: "*When I manage to learn something new or see a different way to do something*" (P6-FG1). The other indicator is `quality of the analyzed code`, as evidenced in: "*When working on projects, we maintain specific quality criteria for pull requests and implementations. Thus, productivity is achieved when these quality standards are met within a timeframe deemed reasonable for the specific task. However, what constitutes a reasonable timeframe inherently depends on the unique characteristics of each project and pull request*" (P4-FG1). These indicators reveal that code reviews ranging from knowledge-sharing processes (`learning in the process`) to code quality standards as productivity indicators.

The developers also cited other indicators as `achieved demand expectations` and `speed in the process`, as evidenced in: "*Productivity to me is when you have a problem to solve, and you manage to reach that exact solution you had set as your goal from the start*" (P2-FG1) and "*I consider a productive review to be one that returns the code for changes when adjustments are needed quickly, because in this case there will need to be adjustments, the code will need to go back for another review, and this cycle should repeat until the code gets merged. And if no changes are required, that this feedback is given quickly so the code can be merged*" (P7-FG1), respectively.

During the focus group sessions, some participants reported a lack of perceived connection between code review and productivity, as exemplified in: "*when identifying potential defects or improvements, I feel I'm contributing, but reviewing code doesn't generate a sense of being productive*" (P8-FG2). Moreover, although they acknowledged its role in ensuring software quality, many participants viewed the activity as inherently unproductive, as evidenced in: "*I'd feel more productive without PR reviews, as they're time-consuming tasks potentially automatable*" (P1-FG2).

Finding #5: Although often perceived as unproductive, participants reported that code reviews can enhance their sense of productivity when the activity expands their knowledge, leads to high-quality code, and is easy and quick to analyze while meeting the expected requirements.

Finally, in **documentation** activities, software developers measure their productivity using the indicators `writing quality` and `clear information`, as illustrated in: "*I've always struggled with starting documents - never know how to begin. Now I use tools that give me starting points, even when writing team cards for test scenarios or acceptance criteria I'd never think of alone. This actually made my team more productive, because now my cards are better described and structured. I feel highly productive having found ways - and tools - that help me write better*" (P2-FG1) and "*When I manage to include everything that matters in the documentation - all the necessary elements. And when I can explain it in a way that's crystal clear for whoever's going to use that documentation*" (P7-FG1), respectively.

During the focus group sessions, the indicators `writing quality` and `clear information` were consistently agreed upon by all developers, as evidenced in: "*the feeling of productivity comes when I manage to include all relevant details, eliminating ambiguities and ensuring anyone can follow the described steps without encountering surprises or needing additional interpretation*" (P5-FG1).

Finding #6: In documentation activities, participants used `writing quality` and `clear information` as indicators to assess their productivity. This perspective highlights how comprehensive, well-structured documentation serves as a mechanism to prevent rework, for example.

4.3 RQ3: How is GenAI expected to affect the Productivity of Software Developers?

We identified both benefits and drawbacks stemming from software developers' expectations regarding GenAI assistance in their activities. Regarding the benefits developers expected `reduce the time spent on technical activities` and `support code writing`, as GenAI tools can generate or test code. These benefits are respectively explained in "*For anything that can save time and streamline processes. Obviously we're still responsible for checking and analyzing if the output fits our needs. Absolutely everything, whether it's generating code or testing a reference*" (P2-FG2) and "*So, time-wise, for those big chunks of code you mostly copy-paste, I think it helps a ton*" (P4-FG1). However, developers also acknowledged the need to review and validate the generated output.

Additionally, they anticipated `learning assistance` and `help initiate tasks`, since GenAI tools may act like a senior developer by providing guidance on specific tasks or suggesting approaches to problem-solving. The following quotes exemplified these benefits: "*ChatGPT turns out to be really productive for basic questions - the kind I'd normally ask more senior colleagues. This way I don't have to constantly bother everyone with simple conceptual doubts*" (P6-FG1) and "*The biggest impact is that initial push to get started. (...) AI tools really help here, especially by giving you that 'hey, I can do this quickly' confidence then you operate at a whole different pace*" (P7-FG1).

Finally, GenAI was expected to support the `use of effective tools and techniques` and to `support documentation writing`, as these tools can offer working examples for tools and generate or review technical content. These benefits are evidenced in "*When the tool shows me working examples and all, proving the approach actually works, I feel way more confident to implement and push forward my ideas and code*" (P6-FG1).

Finding #7: Participants expected GenAI tools to help them `reduce time spent on technical activities`, `support code writing`, provide `learning assistance`, `help initiate tasks`, assist in the `use of effective tools and techniques`, and `support documentation writing`.

On the drawbacks, software developers expected that GenAI tools can generate `dead/duplicated code`, often producing more code than necessary, as evidenced in: "*From my recent experience, what's somewhat concerning about coding with these tools is they seem to measure quality by quantity - they often produce more code than needed. Like some teammates mentioned, you end up reading* *through lots of stuff that shouldn't even be there*" (P7-FG1). The tools can lead to `AI hallucinations`, since generated text is not always precise, as we can see in: "*I've tried using it, but I think the technology still wanders too much when generating text - it ends up not being that useful for me*" (P4-FG2).

Other drawbacks mentioned include the `need to analyze and check for bugs`, as the code generated by GenAI tools requires debugging, as described in: "*With AI, you take half the time to develop, but spend four times longer debugging the created solution. On one hand, you develop and deliver faster; on the other, you spend much more time debugging, more bugs end up appearing, etc*" (P7-FG1); and the `decreased critical sense of novice developers`, as novice developers may become dependent on GenAI tools, as detailed in: "*You gotta use it wisely, or you'll get hooked - especially newbies. The danger is losing your independent thinking skills and becoming tool-dependent*" (P6-FG2).

Finding #8: Participants expected that GenAI tools could produce dead or duplicated code and generate AI hallucinations, as well as diminish the critical thinking of novice developers. They also noted that using these tools increases the need to carefully analyze and verify the generated code for potential bugs.

5 Discussion

This section compares our findings with related studies and maps them onto the dimensions of the SPACE framework. It then introduces a conceptual framework that synthesizes our results and discusses their practical implications for both researchers and industry practitioners.

5.1 Comparison to Related Work

Our findings underscore the cognitive-intensive nature of productivity in software development, distinguishing it from other fields. Unlike domains such as manufacturing or administration, where productivity centers on quantifiable, time-bound throughput [20, 27], developers' perceptions are shaped by intangible and qualitative factors. Productive factors such as progress/evolution, quality, and efficiency (Finding #1) reflect this view, while unproductive ones, including context switching and unclear requirements (Finding #2), emphasize the need for sustained focus. This supports the notion that software productivity depends less on task volume and more on the ability to solve complex problems effectively [26, 34].

In SE field, Meyer et al. [26] investigated on developers' productivity perceptions, suggesting that perceived productivity is strongly associated with task progression and uninterrupted focus. Interestingly, while participants reported that task switching impairs productivity, observations revealed frequent switching, suggesting that interruptions are not uniformly perceived as detrimental. Our findings partially align with this view: participants also emphasized the importance of progress and uninterrupted focus, yet their narratives revealed that context (such as task type and urgency) shapes whether interruptions are viewed as disruptive. In addition, our study emphasizes factors shaped not just by individual preferences but also by organizational context (e.g., quality expectations, delivery deadlines, and team dynamics) particularly relevant in RTO environments where academic and technical pressures coexist.

When it comes to the influence of GenAI, Glushkova [15] reported that developers perceive tools like ChatGPT as supportive of productivity, particularly for junior developers and routine coding tasks. Our participants echoed these benefits (highlighting gains in efficiency and support for code and documentation generation) but also brought forward additional concerns less emphasized in prior work, such as the potential for reduced critical thinking among novice developers and challenges in code review dynamics when AI-generated code is involved. These concerns align with the barriers reported by Banh et al. [3], including limited integration, reliability issues, and underestimated overhead. While Banh et al. [3] focus on engineers from European software companies, our study contributes a perspective from a RTO operating under research and development mandates in Brazil, where adoption of GenAI also raises questions about scientific rigor, explainability, and long-term workforce development. Coutinho et al. [8] examined GenAI adoption in a large organization through surveys and observation, discussing productivity benefits alongside challenges like prompt crafting and data confidentiality. Our findings complement this view, but go further by surfacing how developers interpret GenAI's influence in different types of engineering activities (e.g., coding, documentation, reviewing), and by revealing how these perceptions vary even within a single organization depending on the activity's nature, required autonomy, and expected quality standards.

In summary, our findings confirm and extend prior work, particularly by: (i) grounding the analysis in an RTO setting; (ii) offering a fine-grained perspective on how productivity perceptions vary by activity (e.g., coding vs. reviewing); and (iii) contextualizing the benefits and trade-offs of GenAI based on developer expectations.

5.2 Discussing our Findings through the Lens of SPACE Framework

Our findings suggest that the RTO could benefit from adopting the SPACE framework, which promotes a balanced approach to measuring software developer productivity. This recommendation emerges from the recurring perception among participants that traditional productivity metrics alone (such as task completion rates or code volume) do not sufficiently capture critical aspects like code quality, collaboration, or well-being. To enrich our interpretation, we mapped the insights derived from the focus group sessions to the five SPACE dimensions (Satisfaction and well-being, Performance, Activity, Communication and Collaboration, and Efficiency and Flow) as illustrated in Figure 1.

The initial dimension of the SPACE framework, encompassing **satisfaction and well-being**, is considered by RTO developers to be fostered through the development of reusable code and the reduction of manual efforts like email correspondence and documentation. In this context, GenAI adoption could provide assistance in documentation authoring and the review of technical materials.

Forsgren et al. [13] explained that **performance** should be related not only by its outputs, but also by its quality and impact on customer satisfaction. In this research, having enough time to deliver high-quality code was considered important. For this study's

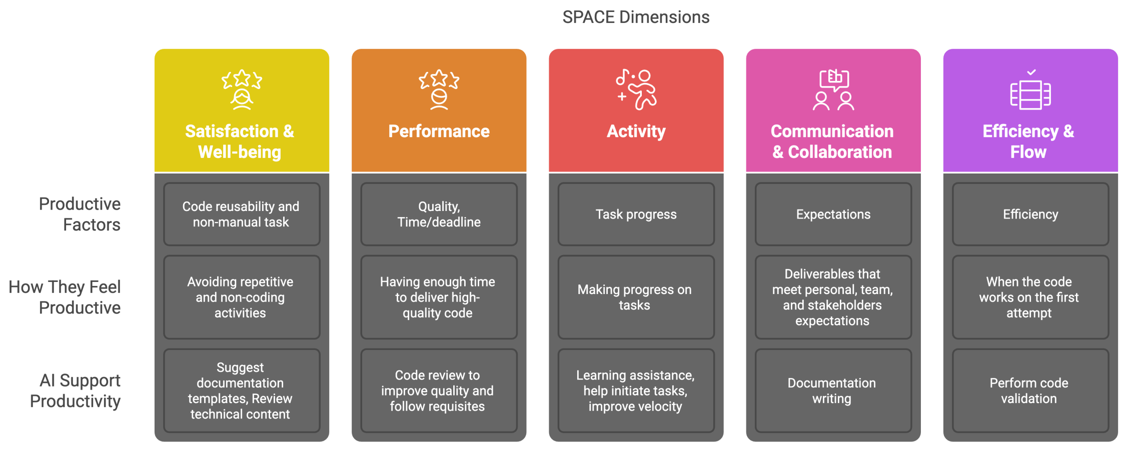
SPACE Dimensions
Satisfaction & Well-being
Performance
Activity
Communication & Collaboration
Efficiency & Flow
Productive Factors
How They Feel Productive
AI Support Productivity
Code reusability and non-manual task
Quality, Time/deadline
Task progress
Expectations
Efficiency
Avoiding repetitive and non-coding activities
Having enough time to deliver high-quality code
Making progress on tasks
Deliverables that meet personal, team, and stakeholders expectations
When the code works on the first attempt
Suggest documentation templates, Review technical content
Code review to improve quality and follow requisites
Learning assistance, help initiate tasks, improve velocity
Documentation writing
Perform code validation

Figure 1: SPACE Framework mapped to productivity perceptions and GenAI support strategies in software development.

participants, GenAI tools would support their work by reviewing code and validating outputs based on the stakeholders' demands.

The **activity** dimension represents the quantity of actions to complete a task or outputs delivered. The participants feel productive when they complete or make progress on a task. Some of them explained that they use or would use GenAI to initiate tasks in order to bring clarity on where to begin. For instance, a junior developer mentioned taking advantage of GenAI to learn and to avoid consulting seniors.

Communication and collaboration dimensions involve team coordination and information flow among members, enhancing discoverability, performance, and reducing burnout risks by exposing bottlenecks. When it comes to the RTO investigated, the developers stated that delivering what is expected by themselves, the team, and the stakeholders is an indication of being productive. Consequently, they would make good use of GenAI tools to write clear documentation.

Finally, the ability to progress without distractions or interruptions (**efficiency and flow**), as defined by the SPACE framework, supports continuous delivery and high performance. Our findings showed that developers perceive this aspect of productivity when the code works on the first attempt, suggesting that GenAI may enhance effectiveness in this regard.

To sum up, the expectation to use GenAI tools (**RQ3**) reveals important relations with previously identified productivity factors (**RQ1**). Specifically, the `Reduced time spent on technical activities` benefit directly relates to both the `Time/Deadline factor` (**RQ1**) and the `Speed in the process` metric for code review (**RQ2**). This example of our results demonstrates that GenAI adoption may positively impacts productivity perception among the participants.

5.3 Implications for Practitioners and Researchers

Figure 2 presents a conceptual map that summarizes our findings. It highlights the factors that software developers involved in this study identified as contributing to feelings of productivity or unproductivity during focus group sessions. The map also includes the expected benefits and drawbacks associated with the use of GenAI tools to support software development activities. Lastly, it illustrates the indicators used to assess productivity in coding, code review, and documentation tasks.

The map offers contributions to both industry and academia. For industry practitioners, it provides empirical evidence on the multidimensional factors influencing developer productivity within the complex environment of a Brazilian RTO. It can serve as an evidence-based foundation for designing organizational strategies that balance GenAI-driven efficiency gains with the preservation of technical quality standards.

From an academic perspective, the map serves as a starting point for systematically investigating productivity dynamics in RTOs, an area that remains understudied in the literature. It also opens avenues for future research aimed at deepening the understanding of productivity and unproductivity factors, and how these relate to emerging technologies and evolving engineering practices.

6 Trustworthiness of the Study

To assess the validity threats of this study, we adopted the qualitative framework proposed by SIGSOFT [30]. Regarding **credibility**, we acknowledge that focus groups are susceptible to researcher bias. To mitigate this, three researchers acted as mediators in each session, and a confirmatory questionnaire was later administered to validate preliminary interpretations. As the questionnaire was sent six months after the focus groups, we recognized the risk of recall bias and therefore embedded synthesized findings for participants to confirm or challenge. Only two discrepancies emerged: one participant did not recall the discussion on documentation-related productivity indicators (upon reviewing the focus group session in which they participated, we found that this topic had been only minimally discussed), and another clarified that `non-coding activities` may or may not be perceived as unproductive, depending on the developer. These limited divergences suggest that the time gap did not substantially hinder participants' ability to reflect critically on the findings. Another methodological consideration is our focus on **coding**, **code review**, and **documentation** activities. This decision was both methodologically and empirically

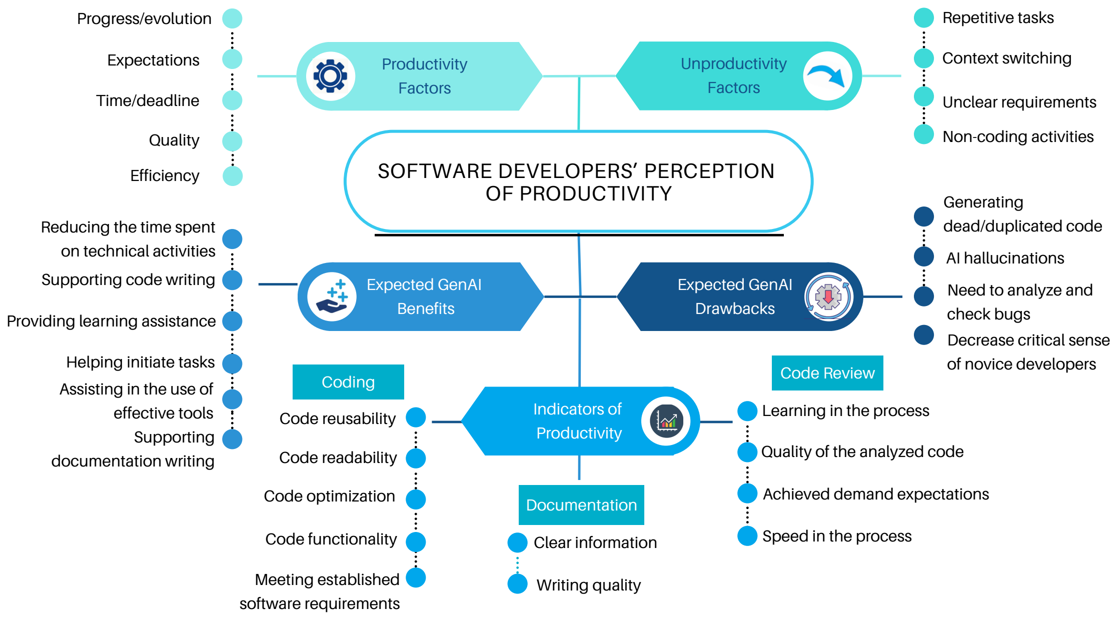

Figure 2: Software developers' perception of productivity conceptual map

grounded, as these were the most frequently cited by participants when describing their productivity experiences during the focus groups. We acknowledge that this scope may exclude other relevant aspects of software work (e.g., requirements elicitation, testing, team meetings) that could reveal additional productivity factors. We view this as a necessary trade-off to ensure analytical depth in the developer-centric activities most salient to our participants.

For **multivocality**, moderators actively promoted participatory equity by encouraging input from less vocal members. Group diversity was planned to include participants with different seniority levels and organizational tenure. In discussions about GenAI use, FG2 had a balanced mix of users and non-users, while most FG1 participants used the tools. The benefits and drawbacks reported in RQ3 aligned with expectations, as moderators asked participants to reflect from a user experience perspective when applicable.

Concerning **reflexivity**, we adopted an independent analysis conducted by two researchers, with arbitration by a third in cases of interpretive disagreement. To ensure methodological **rigor**, we fully documented the process, from session transcripts to coding procedures. The codebook and analytical protocols are openly available [7] for transparency and replication. On **transferability**, we compared our findings with related studies and mapped them to the SPACE framework [13], which helped contextualize the results and highlight both convergences and context-specific limitations.

Lastly, another consideration is the sample size ($N = 16$). Although related studies have surveyed larger populations, they primarily relied on questionnaires—methods suited for breadth but limited in contextual depth. In contrast, our focus group approach within a large industrial setting prioritized interpretive depth over statistical generalizability. Given the interactive nature of focus groups, smaller samples are both expected and methodologically appropriate. While our sample aligns with qualitative research standards, we acknowledge the limitations of working with a small participant group.

7 Conclusion

This study examines developers' perceptions of productivity, focusing on its measurement and the expected impact of adopting GenAI within a large RTO (namely Atlântico Institute) which is a domain barely explored in SE research. Based on focus groups with developers, our findings reinforce that productivity is a multidimensional and context-sensitive construct shaped by interrelated factors across activities such as coding, code review, and documentation. Developers also recognized the dual nature of GenAI, noting both benefits (e.g., task acceleration and quality support) and drawbacks (e.g., over-reliance and reduced critical engagement). To contextualize our results, we mapped them to the five dimensions of the SPACE framework.

As future work, we suggest replicating this study in other RTOs to deepen the understanding of productivity perceptions across different organizational contexts. Expanding participation to include roles such as project managers, product managers, and UI/UX designers could provide a broader view of productivity dynamics. Our methodological design—combining focus groups, thematic analysis, and open data—can serve as a replicable foundation for future studies. Moreover, controlled experiments with LLM-based tools could produce quantitative evidence of their impact on software development workflows, complementing our qualitative findings.

ARTIFACT AVAILABILITY

To ensure transparency and replicability, all data supporting this study are openly accessible via our public repository [7].

ACKNOWLEDGMENTS

This study was partially funded by National Council for Scientific and Technological Development (CNPq - Universal Project 404406/2023-8), the Ceará Foundation for the Support of Scientific and Technological Development (Funcap), the State University of Ceará (UECE - PPGCC) and the Atlântico Institute.

Large Language Models (LLMs) were selectively used to assist with language refinement tasks, such as spelling, grammar correction, and translation into English. All conceptual development, analysis, and interpretation remain the sole responsibility of the authors.

REFERENCES

[1] Jailson B de Andrade. 2010. Connecting Science, Technology and Education. *Journal of the Brazilian Chemical Society* 21 (2010), 1594–1594.

[2] Thales Haddad Novaes de Andrade, Lucas Rodrigo da Silva, and Leda Gitahy. 2013. New policies for science and technology and the impacts on public research institutes: a case study in Brazil. *Brazilian Political Science Review* 7 (2013), 37–61.

[3] Leonardo Banh, Florian Holldack, and Gero Strobel. 2025. Copiloting the future: How generative AI transforms Software Engineering. *Information and Software Technology* 183 (2025), 107751.

[4] Sussy Bayona, Jose A Calvo-Manzano, and Tomás San Feliu. 2012. Critical success factors in software process improvement: a systematic review. In *International Conference on Software Process Improvement and Capability Determination.* Springer, 1–12.

[5] Tuomas Bazzan, Benjamin Olojo, Przemysław Majda, Thomas Kelly, Murat Yilmaz, Gerard Marks, and Paul M Clarke. 2024. Analysing the Role of Generative AI in Software Engineering-Results from an MLR. In *European Conference on Software Process Improvement.* Springer, 163–180.

[6] Edna Dias Canedo and Giovanni Almeida Santos. 2019. Factors affecting software development productivity: An empirical study. In *Proceedings of the XXXIII Brazilian Symposium on Software Engineering.* 307–316.

[7] Murilo Coelho, Isabelle Reinbold, Lizie Sancho, Matheus Paixao, Allysson Allex Araújo, and Sávio Freire. 2025. Replication package for the Paper "Software Developers' Perceptions of Productivity: An Industry-focused Study". https://zenodo.org/records/17335369

[8] Mariana Coutinho, Lorena Marques, Anderson Santos, Marcio Dahia, Cesar França, and Ronnie de Souza Santos. 2024. The role of generative ai in software development productivity: A pilot case study. In *Proceedings of the 1st ACM International Conference on AI-Powered Software.* 131–138.

[9] Aline de Campos, Jorge Melegati, Nicolas Nascimento, Rafael Chanin, Afonso Sales, and Igor Wiese. 2024. Some things never change: how far generative AI can really change software engineering practice. *arXiv preprint arXiv:2406.09725* (2024).

[10] Peter F Drucker. 1999. Knowledge-worker productivity: The biggest challenge. *California management review* 41, 2 (1999), 79–94.

[11] Carlos Henrique C Duarte. 2019. The quest for productivity in software engineering: A practitioners systematic literature review. In *2019 IEEE/ACM International Conference on Software and System Processes (ICSSP).* IEEE, 145–154.

[12] Carlos Henrique C Duarte. 2022. Software productivity in practice: A systematic mapping study. *Software* 1, 2 (2022), 164–214.

[13] 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.

[14] Armstrong Foundjem, Ellis Eghan, and Bram Adams. 2021. Onboarding vs. diversity, productivity and quality—empirical study of the openstack ecosystem. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE).* IEEE, 1033–1045.

[15] Daria Glushkova. 2023. *The influence of Artificial intelligence on productivity in Software development.* Ph. D. Dissertation. Politecnico di Torino.

[16] Ahmed E Hassan, Dayi Lin, Gopi Krishnan Rajbahadur, Keheliya Gallaba, Filipe Roseiro Cogo, Boyuan Chen, Haoxiang Zhang, Kishanthan Thangarajah, Gustavo Oliva, Jiahuei Lin, et al. 2024. Rethinking software engineering in the era of foundation models: A curated catalogue of challenges in the development of trustworthy fmware. In *Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering.* 294–305.

[17] Adrián Hernández-López, Ricardo Colomo-Palacios, and Ángel García-Crespo. 2012. Productivity in software engineering: A study of its meanings for practitioners: Understanding the concept under their standpoint. In *7th Iberian Conference on Information Systems and Technologies (CISTI 2012).* IEEE, 1–6.

[18] Sarah Inman, Sarah D'Angelo, and Bogdan Vasilescu. 2024. Developer productivity for humans, part 8: Creativity in software engineering. *IEEE Software* 41, 2 (2024), 11–16.

[19] Ciera Jaspan and Caitlin Sadowski. 2019. No single metric captures productivity. *Rethinking Productivity in Software Engineering* (2019), 13–20.

[20] Robert L Kahn. 1960. Psychologists in administration (A symposium): III. Productivity and job satisfaction. *Personnel psychology* (1960).

[21] Klaus Krippendorff. 2019. Content Analysis: An Introduction to Its Methodology. https://doi.org/10.4135/9781071878781

[22] Richard A Krueger. 2014. *Focus groups: A practical guide for applied research.* Sage publications.

[23] Rui Li, Jing Rao, and Liangyong Wan. 2022. The digital economy, enterprise digital transformation, and enterprise innovation. *Managerial and Decision Economics* 43, 7 (2022), 2875–2886.

[24] Bojana Lobe. 2017. Best practices for synchronous online focus groups. *A new era in focus group research: Challenges, innovation and practice* (2017), 227–250.

[25] André N Meyer, Laura E Barton, Gail C Murphy, Thomas Zimmermann, and Thomas Fritz. 2017. The work life of developers: Activities, switches and perceived productivity. *IEEE Transactions on Software Engineering* 43, 12 (2017), 1178–1193.

[26] André N Meyer, Thomas Fritz, Gail C Murphy, and Thomas Zimmermann. 2014. Software developers' perceptions of productivity. In *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering.* 19–29.

[27] Kanthi MN Muthiah and Samuel H Huang. 2006. A review of literature on manufacturing systems productivity measurement and improvement. *International Journal of Industrial and Systems Engineering* 1, 4 (2006), 461–484.

[28] Anh Nguyen-Duc, Beatriz Cabrero-Daniel, Adam Przybylek, Chetan Arora, Dron Khanna, Tomas Herda, Usman Rafiq, Jorge Melegati, Eduardo Guerra, Kai-Kristian Kemell, et al. 2023. Generative Artificial Intelligence for Software Engineering–A Research Agenda. *arXiv preprint arXiv:2310.18648* (2023).

[29] Oxford Economics and Merck KGaA, Darmstadt, Germany. 2021. The State of Scientific Research Productivity: How To Sustain A Critical Engine of Human Progress. https://www.oxfordeconomics.com/resource/the-state-of-scientific-research-productivity/ Consulting Report.

[30] Paul Ralph, Nauman bin Ali, Sebastian Baltes, Domenico Bianculli, Jessica Diaz, Yvonne Dittrich, Neil Ernst, Michael Felderer, Robert Feldt, Antonio Filieri, Breno Bernard Nicolau de França, Carlo Alberto Furia, Greg Gay, Nicolas Gold, Daniel Graziotin, Pinjia He, Rashina Hoda, Natalia Juristo, Barbara Kitchenham, Valentina Lenarduzzi, Jorge Martínez, Jorge Melegati, Daniel Mendez, Tim Menzies, Jefferson Molleri, Dietmar Pfahl, Romain Robbes, Daniel Russo, Nyyti Saarimäki, Federica Sarro, Davide Taibi, Janet Siegmund, Diomidis Spinellis, Miroslaw Staron, Klaas Stol, Margaret-Anne Storey, Davide Taibi, Damian Tamburri, Marco Torchiano, Christoph Treude, Burak Turhan, Xiaofeng Wang, and Sira Vegas. 2020. Empirical Standards for Software Engineering Research. arXiv:2010.03525 [cs.SE] https://arxiv.org/abs/2010.03525

[31] Caitlin Sadowski and Thomas Zimmermann. 2019. *Rethinking productivity in software engineering.* Springer Nature.

[32] Robin C Sickles and Valentin Zelenyuk. 2019. *Measurement of productivity and efficiency.* Cambridge University Press.

[33] David W Stewart and Prem Shamdasani. 2017. Online focus groups. *Journal of advertising* 46, 1 (2017), 48–60.

[34] Margaret-Anne Storey, Brian Houck, and Thomas Zimmermann. 2022. How developers and managers define and trade productivity for quality. In *Proceedings of the 15th International Conference on Cooperative and Human Aspects of Software Engineering.* 26–35.

[35] Anselm Strauss, Juliet Corbin, et al. 1990. *Basics of qualitative research.* Vol. 15. sage Newbury Park, CA.

[36] Bart van Ark and Dirk Pilat. 2023. *Navigating the Nexus of Science, Technology, Innovation and Productivity: Productivity Puzzles Blog.* https://www.productivity.ac.uk/news/navigating-the-nexus-of-science-technology-innovation-and-productivity-productivity-puzzles-blog/ The Productivity Institute.

[37] Stefan Wagner and Florian Deissenboeck. 2019. Defining productivity in software engineering. *Rethinking productivity in software engineering* (2019), 29–38.

[38] Stefan Wagner and Emerson Murphy-Hill. 2019. Factors that influence productivity: A checklist. *Rethinking productivity in software engineering* (2019), 69–84.