The Impact of AI Tools on Software Development: A Case Study with GitHub Copilot and Other AI Assistants

Sergio Cavalcante[1], Erick Ribeiro[1] and Ana Carolina Oran[2]
[1]*Fundação Paulo Feitoza – FPFtech, Brazil*
[2]*Institute of Computing, Universidade Federal do Amazonas - UFAM, Brazil*

Keywords: ChatGPT, GitHub Copilot, Code Generation, Empirical Study, Software Engineering, Software Development, Artificial Intelligence.

Abstract: **Background** - With the increasing complexity of software projects and the demand for rapid and high-quality deliveries, Generative Artificial Intelligence (GenAI) tools have emerged as powerful allies in software development. **Objective** - This study aims to evaluate the impact of using Code Generation Assistants—such as GitHub Copilot, ChatGPT, and Gemini—in software development environments. **Method** - We conducted a satisfaction survey with 57 volunteers in an R&D organization, including developers, test analysts, and product owners, collecting quantitative and qualitative data on the use of these tools. **Results** - The results indicate that the use of these tools significantly increased productivity, improved code quality, and accelerated professional learning. Additionally, it facilitated the automation of repetitive tasks, allowing focus on more complex challenges. However, challenges such as the need for constant review of generated code and the risk of excessive dependency were identified. **Conclusion** - We conclude that, despite the challenges, GenAI tools have a significant positive impact on software development, and organizational support is crucial to maximize their benefits.

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

1 INTRODUCTION

With the increasing complexity of software projects and the demand for rapid and high-quality deliveries, Generative Artificial Intelligence (GenAI) tools have emerged as powerful allies in the field of software development (Ziegler et al., 2022). Developers and software engineers are always seeking best practices, tools, and ways to improve work processes (Rajlich, 2014). Recently, a notable trend has been the adoption of Code Generation Assistants. Among the code assistants available in the market, GitHub Copilot[1], ChatGPT[2], and Gemini[3] stand out, all based on GenAI, with the potential to transform the way development professionals code software.

Currently, the use of systems based on GenAI is already being explored in other areas outside the corporate environment and in the context of software development. For example, in the field of education through the automated generation of teaching materials (Lim et al., 2023), in the creation of personalized tutorials (Moon et al., 2023), and in assisting code generation for computer science students (Lira et al., 2024). The growing adoption and massive use of this generative technology are due to its ability to generate high-quality content, similar to what a human being would generate. As a result, the potential applications of GenAI are expanding rapidly, and its impact on various industries could be transformative, paving the way for more efficient workflows and innovative solutions across multiple domains.

In the corporate context, studies on the collaborative partnership between software engineers and intelligent code generation assistants and AI tools are still incipient. There is a need for works that explore the possible advantages and disadvantages obtained with the use of this technology, as well as the main changes for the software development industry. In this scenario of rapid changes provided by GenAI, this work aims to answer the following central question: **How are software engineers being impacted by the use of intelligent code generation assistants and AI tools?**

The research presented in this article aims to evaluate the impact of using Code Generation Assistants

[1]https://github.com/features/copilot/
[2]https://chat.openai.com/
[3]https://gemini.google.com/

Cavalcante, S., Ribeiro, E. and Oran, A. C.
The Impact of AI Tools on Software Development: A Case Study with GitHub Copilot and Other AI Assistants.
DOI: 10.5220/0013294700003929
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 27th International Conference on Enterprise Information Systems (ICEIS 2025) - Volume 2*, pages 245-252
ISBN: 978-989-758-749-8; ISSN: 2184-4992
Proceedings Copyright © 2025 by SCITEPRESS – Science and Technology Publications, Lda.

and AI tools in software development environments, through the application of a satisfaction survey with **57 volunteers**. The relevance of this study lies in the growing adoption of these tools, which promise to streamline the coding process, reduce errors, and increase productivity. By investigating developers' perceptions regarding the benefits, limitations, and challenges of using these assistants, we can offer valuable insights into the evolution of these technologies. This work seeks not only to measure the direct impact on professionals' efficiency but also to contribute to a deeper understanding of the acceptance and expectations around these tools in the daily life of software engineering.

This article is organized as follows: Section 2 presents works that explore the use of the main current code generation tools. In Section 3, the methodology applied in this study is presented. Sections 4 and 5 present the results obtained, including insights and observations. Finally, in Section 6, the final considerations and future works are described.

2 RELATED WORKS

This section addresses works that have analyzed the use of automatic code generation tools in the field of software development from different perspectives.

In (Imai, 2022), a pair programming experiment was conducted involving 21 participants, using GitHub Copilot versus a human partner. The study measured productivity based on lines of code produced and code quality by the lines removed. The researchers concluded that Copilot helps generate a higher number of lines of code but, in return, results in a higher number of lines deleted. Although it is a study focused on the amount of code, it is important to note that the number of lines of code is not necessarily indicative of code quality; our work did not follow this line of evaluation.

In (Peng et al., 2023), an experiment was conducted by Microsoft researchers involving 95 programmers. The participants were tasked with creating an HTTP server in the JavaScript language, with the option to seek help online for the challenges faced. The programmers were divided into two groups: one with access to GitHub Copilot (the treatment group) and one without (the control group). The results showed that the treatment group completed their tasks faster, even with less experienced developers. Our experiment also includes developers but is not limited to the perspective of technical people working to solve a problem.

In (Yetiştiren et al., 2023), GitHub Copilot was compared with AWS CodeWhisperer and ChatGPT in various code quality perspectives, such as validity, correctness, security, reliability, and maintainability. They found that the latest versions of these tools influence the ability to generate correct code. Among the findings, ChatGPT was the most successful tool, while Amazon CodeWhisperer was the worst. In the comparison between GitHub Copilot and CodeWhisperer, Copilot performed better. In light of this work, in our study, we also chose to evaluate multiple tools, discarding CodeWhisperer.

3 METHODOLOGY

This study aimed to explore the impact of AI tools on software development within an R&D organization focused on delivering tailored software solutions to clients across various industries. The methodology included defining research objectives, selecting participants based on their use of AI tools, and creating a survey focused on areas such as usage frequency, perceived benefits, and challenges.

3.1 Study Planning

The research was designed to capture both quantitative and qualitative data, aiming to provide a comprehensive view of the impacts perceived by users of these tools. The methodology was divided into three main phases:

1. **Definition of Research Objectives.** The goal was to explore how code generation tools, such as GitHub Copilot, ChatGPT, and Gemini, are being used in the organization and what is the perceived impact by developers, test analysts, product owners (POs), and other roles.
2. **Sample Selection.** The survey was sent to professionals from different areas, including developers (junior, mid-level, and senior), test analysts, and product owners. Participants were selected based on their use of the mentioned tools, either through organizational licenses or personal use. A total of 57 valid responses were collected.
3. **Questionnaire Design.** The questionnaire was developed based on questions raised in the reference study[4] and adapted to the reality of the organization. The ChatGPT tool was used to assist in creating the questions, ensuring they were comprehensive and focused on specific points such as

[4]https://www.tabnine.com/blog/github-copilot-for-business/

productivity, code quality, job satisfaction, learning, and challenges in using the tools. The questionnaire was divided into five main sections:

(a) **Frequency and Type of Use.** Questions to measure the frequency of use and whether professionals were using licenses provided by the company, free versions, or purchased with personal resources.

(b) **Tools Used** Identify which other AI tools were being used besides GitHub Copilot.

(c) **Perceived Benefits.** Questions about the main perceived benefits in using the tools, such as increased productivity, code quality, and learning.

(d) **Overall Impact on Work.** Assessment of how the tools influenced meeting deadlines, focus on problem-solving, and impact on overall satisfaction.

(e) **Challenges and Disadvantages.** Open-ended questions to identify the main challenges and disadvantages perceived in using AI tools.

3.2 Conducting the Study

The study was conducted over three months by three researchers—two from the organization and one from academia—and involved the application of a questionnaire distributed through Microsoft Forms, a tool commonly used by the organization for feedback collection. The questionnaire targeted development teams, QA, and product management, with emphasis on developers and roles such as product owners (POs), who frequently use tools like ChatGPT. The sample consisted of 57 valid responses, with participants reporting their frequency of AI tool usage, licensing types, perceived benefits, and challenges. Quantitative data were collected using structured questions, including Likert scales, while qualitative data were obtained through open-ended questions, later categorized into themes such as ”tool dependence” and ”quality of code suggestions.”

3.3 Data Analysis

The data were analyzed through a combination of quantitative and qualitative analysis. The analysis process followed these steps:

1. **Quantitative Analysis.** For the quantitative data, Microsoft Excel was used to calculate frequencies and distributions of responses. The analysis focused on identifying usage patterns, such as the frequency with which the tools were used and the most common type of license among participants. Graphs were generated to illustrate the distributions of responses, such as the perceived impact on productivity and code quality.
2. **Qualitative Analysis.** Open-ended responses were analyzed using thematic categorization techniques. Responses were grouped into categories such as “productivity benefits”, “increase in code quality”, “technical challenges”, and “perceived disadvantages”. This process allowed us to identify patterns in users’ perceptions and brought important insights for discussing the results.
3. **Data Integration.** Finally, qualitative and quantitative data were integrated to provide a complete view of the impact of AI tools in the work environment. Emphasis was given to cases where qualitative responses corroborated quantitative findings, such as reports on increased productivity and tool dependence.

4 RESULTS

In this section, we present the findings of our study, which aimed to evaluate the impact of using Code Generation Assistants in software development environments, through a satisfaction survey with 57 volunteers. We analyzed both quantitative and qualitative data to offer a comprehensive view of developers’ perceptions regarding the benefits, limitations, and challenges in using these tools.

4.1 Quantitative Analysis

The volunteers reported significant improvements in efficiency and productivity when using AI tools used in the study.

4.1.1 Frequency of Use of the Tools

Among the 57 respondents, the majority reported frequent use of AI tools (such as GitHub Copilot, Copilot 365, ChatGPT, etc.), with emphasis on daily use. The distribution of use was as in Figure 1.

The pie chart shows the frequency of AI tool usage among the 57 respondents, reflecting a high integration of these tools into their workflows. This data highlights the frequent adoption of AI tools, emphasizing their role in improving productivity and streamlining tasks in daily work routines.

4.1.2 Perceived Benefits

When asked about the benefits of using AI, most responses were positive, as shown in Figure 2, with

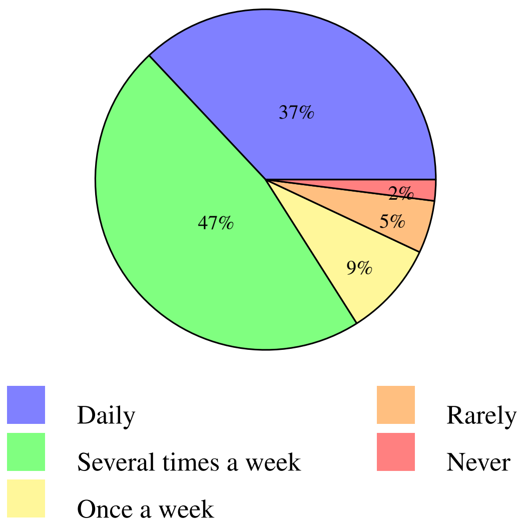

Figure 1: Frequency of Use of the Tools.

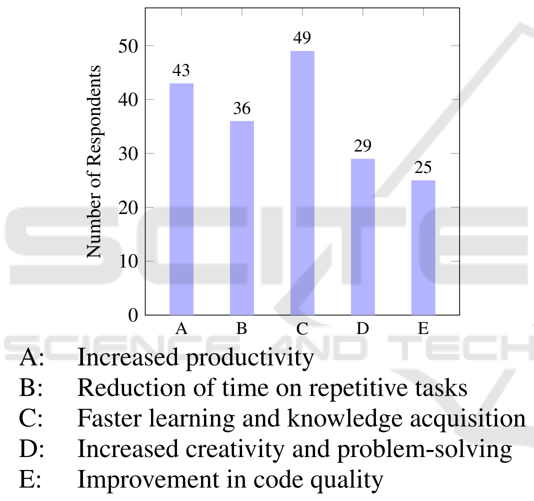

Figure 2: Perceived Benefits Reported by Respondents.

participants citing improved efficiency and decision-making.

The bar chart presents the perceived benefits of using AI tools in software development, highlighting that the most significant impact is on faster learning and knowledge acquisition (C), with 49 respondents reporting this benefit. This represents a large portion of the respondents, indicating that AI tools are strongly facilitating skill development. Increased productivity (A) was the second most cited benefit, with 43 respondents, reflecting the tools' contribution to efficiency in the workplace. Other notable impacts include the reduction of time on repetitive tasks (B) with 36 respondents and increased creativity and problem-solving (D) reported by 29 respondents, which suggests that the tools not only save time but also help developers focus on more complex and innovative tasks. Improvement in code quality (E) was mentioned by 25 respondents, indicating a positive, although slightly lesser, influence on the quality of work. These findings underline that AI tools are highly valued across multiple dimensions, particularly for enhancing learning, productivity, and creativity.

4.1.3 Impact on Overall Work Experience

When asked about the overall impact of AI tools on work.

The pie chart, shown in Figure 3, illustrates the respondents' perception of the overall impact of AI tools on their work experience. This data suggests that a vast majority of the respondents—48 out of 57 (84%) in total—have experienced some level of improvement in their daily work routines due to these tools, reinforcing the idea that AI is contributing meaningfully to enhancing productivity and job satisfaction in software development environments.

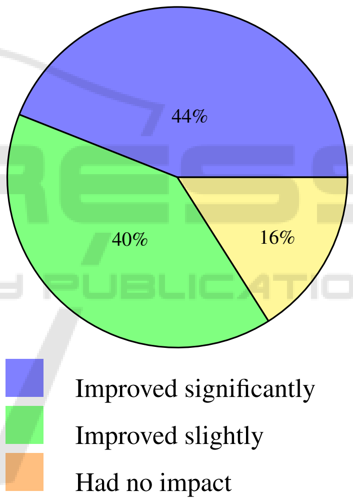

Figure 3: Impact Reported by Respondents.

4.1.4 Ability to Meet Deadlines

In terms of meeting deadlines, the distribution was as in Figure 4.

The pie chart displays respondents' perceptions of how AI tools impacted their ability to meet deadlines. These results suggest that a majority of respondents—43 out of 57 (75%)—recognized some level of improvement in managing deadlines, likely due to the efficiency and time-saving features of AI tools. The significant portion that acknowledged improvement reflects the general sentiment that such tools help professionals avoid delays by automating routine tasks and providing quicker access to solutions, allowing more focus on meeting deadlines.

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

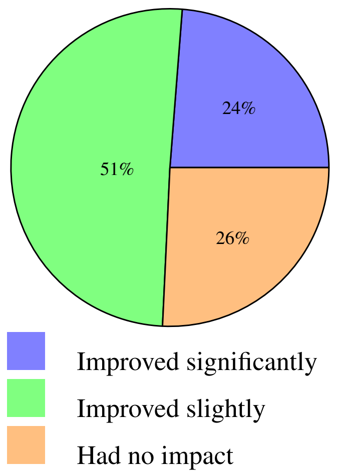

Figure 4: Ability to meet Deadlines.

4.1.5 Focus on Problem Solving

Asked if the tools helped focus on more complex tasks, the distribution was as in Figure 5.f

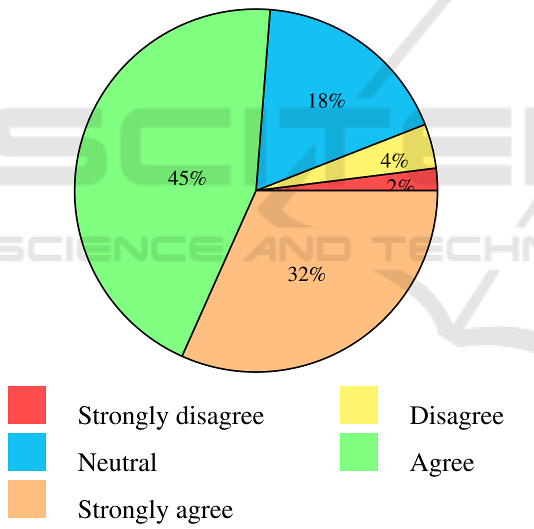

Figure 5: Focus on Problem Solving.

The pie chart illustrates how respondents felt AI tools helped them focus on problem-solving tasks. The overall trend suggests that the majority—43 out of 57 respondents (77%)—felt that the tools helped them focus on the core of their work, reducing time spent on tedious tasks. By allowing engineers to bypass less engaging or repetitive aspects of development, these tools contributed to a more satisfying work experience, as developers could concentrate more on problem-solving and innovation, potentially improving overall job satisfaction.

4.2 Qualitative Analysis

To deepen the understanding of professionals' experiences with AI tools, we conducted a qualitative analysis of open-ended responses. The feedback was categorized into positive aspects and challenges faced, highlighting key themes, which we discuss below. Please note that only a subset of responses will be presented here. To access the complete set of responses, please refer to the following link on GitHub[5].

4.2.1 Positive Points

Code Generation and Code Search. Many respondents cited increased productivity and development speed as one of the main benefits:

> "Specific searches for the use of methods and classes, almost automatic code documentation. Sometimes it suggests more than I would like, but those were exceptions." – **P18 (Data Scientist)**

> "It greatly accelerated my process when writing repetitive codes and solving them more efficiently instead of creating a function; it suggested a native function that I did not know." – **P42 (Developer)**

These comments highlight how AI tools help developers quickly find code snippets and solve simple doubts, saving valuable time.

Code Quality. Improvement in code quality was also highlighted, citing better understanding, succinct, readable, and more efficient code:

> "Improved code quality because I can visualize other possibilities. Improved documentation development speed and also enabled testing new libraries more agilely." – **P31 (Developer)**

> "Improvement in code quality following best practices. Creating, commenting, and compacting ladder programming." – **P32 (Automation Engineer)**

These responses suggest that AI tools not only help code faster but also write better and more efficient code.

Learning. AI was perceived as a vector for learning:

> "It mainly helped in solving problems involving stacks that are generally not used in the lab; currently, I work on a project involving WPF and C#, and I had never had contact with these technologies; the role of AI in increasing my learning process in these stacks has been fundamental for my delivery deadlines and conventional quality of the codes I produce." – **P46 (Developer)**

[5]https://github.com/erickrribeiro/iceis-2025/blob/main/answers.csv

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

These responses indicate that AI tools contribute to faster learning and understanding of new technologies.

Repetitive Tasks. Faster and more focus on the main problem:

> "When working with backend and frontend, you often need to insert repetitive code. An AI tool identifies such recurrence and already inserts the code, increasing productivity." – **P26 (Developer)**.

Infra/DevOps. AI tools assist in creating scripts and complex commands, streamlining infrastructure tasks:

> "Many things like scripts to create something in Terraform, GitLab CI, infra tools in general context, to make a sed command to change text inside an Nginx configuration file; it makes the regular expression, without me having to keep searching how to get the value of a text to make the change via regular expression; it already gives me how to do it, and most of the time it's correct" – **P3 (DevOps)**

QA. AI contributes to efficiency in creating test scripts, speeding up the quality assurance process:

> "Using Copilot in the development of my Scripts greatly improves the speed of typing giant scripts, as the functionality cites an example taking from the context to speed up the process. Thus, greatly helping in rapid development in the software testing part" – **P5 (QA Analyst)**

Data Engineering. AI contributes to productivity in creating Database scripts:

> "In laborious tasks that would be done manually, such as creating a schema in Spark with 300 columns" – **P6 (Data Engineer)**

Documentation/Proposals. Productivity and speed in delivering proposals and documentation:

> "For example, the use of AI tools in the preparation of material made available to the Technical Report team has a relevant impact, as it speeds up the search for fixed concepts and reference material, as well as the possibility of improving the quality of the prepared material" – **P9 (People and Process)**

> "Improved writing project proposals, email drafting, etc." – **P51 (Project Manager)**

These examples above show that the benefits of AI tools extend beyond coding, assisting in documentation and communication tasks.

Insights and Mentoring. Improvement in content generation and ideas:

> "Communication: I ask AI to apply Nonviolent Communication in emails and others; Efficiency: I talk to AI to get insights into problem-solving, conflicts, and lessons learned ceremonies. Team mentoring: review card writing and acceptance criteria; Learning: when I want to apply something new with the team and need mentoring" – **P22 (People and Process)**

> "I don't use AI just for automation but also for creation. In addition to improving and creating texts, making lists, AI can generate ideas that evolve into other ideas. This is extremely useful for solving various problems, from UX issues to tips on how to deal with difficult clients, as AI has the ability to better understand contexts" – **P29 (UX Designer)**

> "It impacted practically all stages of a project's execution. I can focus more on the overall vision of the project and delegate the creation of small modules to AI." – **P25 (Product Owner)**

These reflections above demonstrate how AI tools can serve as a source of inspiration and guidance in various aspects of work.

4.2.2 Negative Points

Need for Review and "Hallucinations". Several respondents expressed concerns about the accuracy of the generated code:

> "There would always be supervision due to common hallucinations in free AI versions" – **P8 (Developer)**

> "As a disadvantage, the approach suggested is not always the most efficient, but it serves as a good guide" – **P41 (Developer)**

These comments above emphasize the importance of reviewing the suggestions generated by AI tools to ensure correctness.

Limited Context Understanding. Some users noted that AI tools sometimes lack a complete understanding of the project's context:

> "Regarding the disadvantages, it still leaves much to be desired in recognizing the complete context of the project's codes as a whole. It still gets lost a lot." – **P25 (Product Owner)**

Generates Dependency. Some respondents mentioned the risk of excessive dependence on the tools:

> "These LLMs are a tool. But people have become dependent on them to the point of not wanting to think anymore and going straight to consult them. If they continue like this, they will stop being people who think and become just firefighters." – **P30 (Developer)**

Thus, the data suggest that AI tools are playing a significant role in transforming software development practices, with positive impacts on the efficiency and satisfaction of the professionals involved.

5 DISCUSSION

The results of this study provide valuable insights into the impact of using AI-based Code Generation Assistants in software development. The initial motivation was to understand how these tools are influencing the work of software professionals, especially in terms of productivity, code quality, and job satisfaction.

5.1 Perceived Benefits

The quantitative data indicate that the adoption of AI tools is significant among the professionals surveyed, with 84% of respondents using them daily or several times a week. This high level of adoption suggests that AI tools are becoming an integral part of developers' workflows.

Most participants reported increased productivity (75%) and reduced time spent on repetitive tasks (63%). These results are consistent with Related Works section, which suggest that AI tools can significantly speed up the coding process and enhance efficiency across various tasks. Additionally, 86% of respondents experienced faster learning and knowledge acquisition, highlighting the educational value of these tools in supporting the continuous development of professionals' skills.

Improvement in code quality was highlighted by 44% of participants. Qualitative responses reinforce this point, with developers like P31 (Developer) mentioning: "Improved code quality because I can visualize other possibilities... and enabled testing new libraries more agilely." This suggests that AI tools not only streamline code production but also encourage more robust and efficient coding practices.

Furthermore, the aspect of learning is significant. Participants reported that AI tools helped them understand new technologies and unfamiliar stacks, as observed by P46 (Developer): "The role of AI in increasing my learning process in these stacks has been fundamental for my delivery deadlines and conventional quality of the codes I produce." This indicates that AI tools can serve as virtual mentors, assisting in professional growth.

The automation of repetitive tasks was another key benefit identified. As mentioned by P27 (Developer): "This is laborious work that takes away useful time and energy for solving more complex and specific problems." By automating these tasks, AI tools allow developers to focus their efforts on more complex challenges, potentially increasing innovation and the quality of the solutions developed.

5.2 Challenges and Limitations of AI Tools

Despite the benefits, participants also pointed out challenges and limitations. The need for constant review of the generated code was highlighted, with concerns about "hallucinations" of the tools, where the suggested code may not be accurate or appropriate. P8 (Developer) warned: "There would always be supervision due to common hallucinations in free AI versions" This emphasizes the need for developers to maintain an active and critical role when using these tools, not blindly trusting the suggestions provided.

Additionally, some participants noted a lack of complete understanding of the project's context by AI tools, as mentioned by P25 (Product Owner). This limitation can lead to code suggestions that do not perfectly align with the project's specific requirements.

A point of concern raised was the risk of excessive dependence on AI tools. P30 (Developer) observed: "People have become dependent on them to the point of not wanting to think anymore and going straight to consult them." This observation raises questions about the long-term impact on the development of professionals' skills. If developers rely too much on AI tools, this can negatively affect their ability to solve problems independently and think critically.

6 LIMITATIONS AND THREATS TO VALIDITY

The primary limitation of this study is its focus on a single organization, which may limit the generalizability of the findings to other contexts or industries. Additionally, as AI tools continue to evolve rapidly, the relevance of the results could diminish over time, necessitating ongoing research. This study provides an observational perspective on the use of AI tools within the organization, offering insights rather than establishing cause-and-effect relationships. Future studies could expand the scope to include multiple organizations and explore performance metrics to complement these findings.

7 CONCLUSIONS

In this study, we investigated the impact of using Artificial Intelligence (AI) tools, such as GitHub Copilot, ChatGPT, and Gemini, in software development within an R&D organization. Through quantitative and qualitative analysis, we sought to understand how these tools influence productivity, code quality, job satisfaction, and other relevant aspects for professionals involved in the software development cycle.

The quantitative results indicated a significant adoption of these tools, with most professionals using them daily or several times a week. The perceived benefits include a substantial increase in productivity, reduced time on repetitive tasks, improved code quality, and acceleration in the learning and knowledge acquisition process. In addition, many participants reported that AI tools helped them meet deadlines more efficiently and focus on more complex tasks, delegating routine tasks to intelligent assistants.

The qualitative analysis complemented these findings, revealing that developers value the ability of AI tools to generate and search code quickly, improve code quality through efficient suggestions, and facilitate learning of new technologies. The positive impact on documentation, proposal preparation, and obtaining insights and mentoring was also highlighted.

However, participants also identified challenges and limitations. The need for constant review of the generated code, the 'hallucinations' of the tools, and the lack of complete understanding of the context were pointed out as points of attention. There were concerns about the possible excessive dependence on AI tools, which could affect developers' ability to think critically and solve problems autonomously. In addition, the importance of organizational support, especially concerning the acquisition of paid licenses, could mitigate some of the observed disadvantages.

While this study evaluated the impact of AI tools, a comparative analysis of Gemini, ChatGPT, and GitHub Copilot remains an opportunity for future research to further explore their unique strengths and applications in each of the software engineering roles.

ACKNOWLEDGEMENTS

We would like to thank Fundação Paulo Feitoza (FPFtech) for the financial support, as well as the opportunity to conduct this research and present it. Additionally, we acknowledge all participants who kindly contributed to this study. We also acknowledge the use of artificial intelligence–generated text in the preparation of this article. Specifically, the sections **Introduction** and **Discussion** were produced with the assistance of the language model ChatGPT.

REFERENCES

Imai, S. (2022). Is github copilot a substitute for human pair-programming? an empirical study. In *Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Companion Proceedings*, ICSE '22, page 319–321, New York, NY, USA. Association for Computing Machinery.

Lim, W. M., Gunasekara, A., Pallant, J. L., Pallant, J. I., and Pechenkina, E. (2023). Generative ai and the future of education: Ragnarök or reformation? a paradoxical perspective from management educators. *The International Journal of Management Education*, 21(2):100790.

Lira, W., Neto, P. S., and Osorio, L. (2024). Uma análise do uso de ferramentas de geração de código por alunos de computação. In *Anais do IV Simpósio Brasileiro de Educação em Computação*, pages 63–71, Porto Alegre, RS, Brasil. SBC.

Moon, J., Yang, R., Cha, S., and Kim, S. B. (2023). chatgpt vs mentor : Programming language learning assistance system for beginners. In *2023 IEEE 8th International Conference On Software Engineering and Computer Systems (ICSECS)*, pages 106–110.

Peng, S., Kalliamvakou, E., Cihon, P., and Demirer, M. (2023). The impact of ai on developer productivity: Evidence from github copilot.

Rajlich, V. (2014). Software evolution and maintenance. In *Future of Software Engineering Proceedings*, FOSE 2014, page 133–144, New York, NY, USA. Association for Computing Machinery.

Yetiştiren, B., Özsoy, I., Ayerdem, M., and Tüzün, E. (2023). Evaluating the code quality of ai-assisted code generation tools: An empirical study on github copilot, amazon codewhisperer, and chatgpt.

Ziegler, A., Kalliamvakou, E., Li, X. A., Rice, A., Rifkin, D., Simister, S., Sittampalam, G., and Aftandilian, E. (2022). Productivity assessment of neural code completion. In *Proceedings of the 6th ACM SIGPLAN International Symposium on Machine Programming*, MAPS 2022, page 21–29, New York, NY, USA. Association for Computing Machinery.