Behind Agentic Pull Requests: An Empirical Study on Developer Interventions in AI Agent-Authored Pull Requests

Syrine Khelifi
ÉTS Montreal, University of Quebec
Montréal, Canada
syrine.khelifi.1@ens.etsmtl.ca

Ali Ouni
ÉTS Montreal, University of Quebec
Montréal, Canada
Ali.Ouni@etsmtl.ca

Maha Khemaja
ISSAT Sousse, University of Sousse
Sousse, Tunisia
maha.khemaja@issatso.u-sousse.tn

Abstract

AI coding agents are increasingly being adopted to autonomously author pull requests (PRs). While these agents can perform a wide range of tasks, little is known about how much human intervention is required to collaborate with them and integrate AI Agents contributions in practice. In this paper, we conduct an empirical study on human intervention in agent-authored PRs (APRs) as a measure of human effort and oversight. Using the AIDev dataset, we first compare how often humans intervene in APRs vs human-authored PRs (HPRs), as well as the resulting outcomes of this intervention. We then conduct a qualitative thematic analysis of human interventions in APRs and derive a taxonomy including 4 high-level categories and 42 intervention actions. Our results show that, human interventions occur less frequently in APRs than in HPRs (52.17% vs. 83.59%), but when it occurs in APRs, it requires higher review effort, including larger code churn and longer durations. Our taxonomy results show that most human effort is spent on guidance-level interventions with 58.02%, focusing on restricting the agent's actions and enforcing project-conventions, followed by decision-level interventions at 21.16%, direct code changes at 17.05% and operational-level intervention at 3.69%. Which indicates that, collaboration with coding agents, is shifting developer work from implementation to supervision, guidance and quality control.

Keywords

Agentic AI, Human Intervention, Pull Request

ACM Reference Format:
Syrine Khelifi, Ali Ouni, and Maha Khemaja. 2026. Behind Agentic Pull Requests: An Empirical Study on Developer Interventions in AI Agent-Authored Pull Requests. In *23rd International Conference on Mining Software Repositories (MSR '26), April 13–14, 2026, Rio de Janeiro, Brazil.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3793302.3793586

1 Introduction

The rapid adoption of AI coding agents is transforming the landscape of software engineering [9, 12, 15]. These agents are capable of autonomously author PRs and contribute to open-source software projects [10] by fixing bugs [13], writing tests [13] and refactoring code [11]. Tools such as Claude Code, Cursor, Devin, GitHub Copilot, and OpenAI Codex are now acting more as AI teammates working alongside developers rather than being AI assistants [13]. They have the ability to generate and author PRs and contribute directly to software projects [20]. Although AI-powered tools have shown substantial capabilities, previous studies have emphasized that human oversight remains essential to guaranty high-quality software [8, 16]. This indicates that AI coding agents, do not fully eliminate human involvement. Despite their growing use, an open key question that remains is how much human effort is required to work with APRs. Several works have studied the review effort in AI-assisted PRs [6, 19]. Although AI coding agents can autonomously propose code changes, human oversight is still needed [13]. This human intervention, such as editing code, adding commits, or correcting agent-authored changes, represents a measurable form of human effort that reflects how developers collaborate with AI agents in practice. Quantifying such effort, helps in understanding the actual cost and level of human oversight these AI coding agents require. Without it, claims about productivity gains from AI coding agents [17], are hard to interpret. In this paper, we study human intervention in APRs as a way to quantify developers' oversight, to provide a foundation for understanding the human effort required to review, correct and integrate agent-authored contributions. Our study is based on the AIDev dataset [14]. This dataset contains 932,791 APRs, created by five AI coding agents: Claude Code, Cursor, Devin, GitHub Copilot, and OpenAI Codex, across a set of 116,211 open-source projects (OSS). Our study addresses two research questions. Hence, we investigate how often humans intervene in APRs as compared to HPRs and we perform a thematic analysis [4], to identify the specific types of interventions that humans perform on APRs and propose a taxonomy for human intervention types. Specifically, we address the following research questions:

- **RQ1.** ***How often do humans intervene in Agentic-PRs vs Human-PRs?***
- **RQ2.** ***What types of interventions do human developers perform on Agentic-PRs?***

Our findings reveal that, while humans tend to intervene less frequently in APRs (52.17%) compared to HPRs (83.59%), interventions in APRs often require more review effort, involves longer review cycles (42.9 hours vs. 22.2 hours), and more iterative oversight, even when those APRs are rejected. Moreover, our taxonomy reveals that, the majority of this effort, is mostly directed at interacting with coding Agents by guiding them (58.02%), and constraining their actions, rather than directly modify code (17.05%). This indicates a shift in developers' work from implementation to supervision, coordination and quality control.

Open Science. We provide our comprehensive replication package for future replications and extensions [1].



This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.
MSR '26, Rio de Janeiro, Brazil
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2474-9/26/04
https://doi.org/10.1145/3793302.3793586

2 Study design

Our study consists in two main steps: (1) Data collection and (2) Data analysis, as shown in Figure 1.

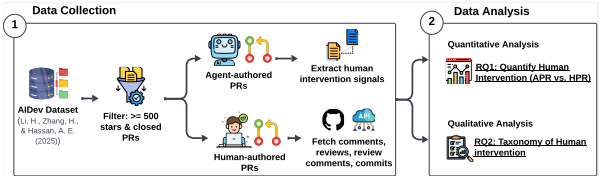


Figure 1: Study Design Overview.

2.1 Data collection

Our study is based on the AIDev dataset [14]. This dataset contains 932,791 APRs, including their commits, commit details, related issues, comments, reviews and review comments, as well as a total of 6,618 HPRs. To enable a fair comparison between APRs and HPRs, we limit our analysis to repositories with at least 500 stars and containing both APRs and HPRs. We filter them to only keep closed PRs. An analysis of HPRs, shows that this filter captures the majority of the dataset: 92.61% of HPRs are both closed and belong to repositories with more than 500 stars. Only 7.08% of HPRs are open PRs from repositories with more than 500 stars, and 0.3% are closed PRs from smaller repositories. We obtain 11,048 APRs and 6,103 HPRs. For each repository, we randomly sample an equal number of APRs and HPRs, where the number of sampled PRs equals the smaller set of the two PR sets (*e.g*, if a repository contains 10 APRs and 4 HPRs, we sample 4 PRs from each group). This results in two balanced datasets of 5,869 APRs and 5,869 HPRs, spanning the same set of repositories and ensuring that all comparisons are made within identical projects. For commit-level analysis, we take into consideration authorship attribution practices across AI coding agents. Prior work [13] has shown that while Devin, GitHub Copilot, and Cursor explicitly indicate their authorship in commit metadata, OpenAI Codex attributes its commits to humans. This makes detecting human intervention in those commits infeasible. We therefore exclude OpenAI Codex PRs from commit-level human intervention analysis, while still considering their comments, reviews, and review comments. For HPRs, we retrieve the corresponding comments, reviews, review comments, and commits using the GitHub API, allowing us to compute identical human intervention metrics for both APRs and HPRs.

2.2 Data analysis

Our data analysis consists of quantitative and qualitative analysis.

2.2.1 Quantitative data analysis. To answer RQ1, we compare APRs with HPRs in terms of frequency of human intervention. We define human intervention as any human-authored activity associated with a PR. This includes: comments, reviews, review comments and commits where the author is a human. Any PR that has at least one artifact (comment, review, review comment or commit) that was authored by a human, is considered to have human intervention. Using this definition, we compute the proportion of PRs that receive human intervention in each group, further distinguishing between accepted and rejected PRs. To characterize the effort associated with human intervention, we analyze a set of PR review and size metrics that have been widely used to capture review intensity and coordination effort in prior studies of PR-based workflows [2, 6, 7]. These metrics include: number of messages, number of inline comments, code churn, number of commits, review duration, number of modified files and PR description length.

2.2.2 Qualitative data analysis. To answer RQ2, we identify and analyze the types of human interventions performed on APRs. For this, we perform a thematic analysis [4], a commonly used technique in software engineering research [2, 3, 6, 18]. To perform this analysis, we focus on the APRs that received at least one human intervention. We have 3,062 APRs with human intervention. From this subset, we select a random representative sample for manual analysis, achieving a 95% confidence level with a 5% margin of error, resulting in a sample size of 342 APRs with human intervention. For each APR, we extract all the associated comments, reviews, review comments and commits authored by humans and we filter out empty and non-English artifacts. Our process for creating the taxonomy of intervention types, includes the following steps:

Step 1: Initial inspection of human-authored artifacts: In this step, we started by manually inspecting all relevant PR artifacts associated with APRs, including commits, comments, reviews, and review comments authored by humans in order to familiarize ourselves with the nature of human interventions and to identify recurring patterns in how developers intervene in APRs.

Step 2: Open coding of human interventions: For this step, we used an open coding process to the inspected artifacts. Human interventions were assigned descriptive labels to describe the primary intent of the intervention, such as modifying functionality, making a decision about PR outcome, etc.

Step 3: Iterative refinement and merging of labels: Next, labels were iteratively refined and similar labels were merged. This is to ensure consistency and reduced redundancy among types.

Step 4: Abstraction into higher-level categories: Finally, the refined labels were grouped into higher-level intervention categories that capture main types of human intervention in APRs. These higher-level categories form the proposed taxonomy of human intervention types, representing common patterns of developer effort and oversight when working with AI coding Agents.

3 Study results

3.1 RQ1: How often do humans intervene in Agentic-PRs vs Human PRs?

Humans intervene more often in HPRs than in APRs, accounting for 83,59% of HPRs compared to 52,17% of APRs. Table 1, shows a clear difference in intervention frequency between APRs and HPRs. Overall, HPRs are almost twice as likely as APRs to receive human intervention. One explanation is that maintainers may be more inclined to engage with human-authored contributions, as they might be easier to negotiate, clarify, or iteratively refine compared to APRs.

Human intervention often leads to higher acceptance rates for both APRs (36.65%) and HPRs (70.83%), but it might be more effective for HPRs than APRs. Table 1 shows that, across

both APRs and HPRs, PRs involving human intervention are more frequently accepted than rejected. Comparing accepted and rejected PRs within each PR type, we observe that intervention is much more strongly associated with acceptance in HPRs than in APRs. This suggests that while intervention is beneficial for both PR types, it is more effective when it comes to HPRs. A possible explanation is that human-human interaction enables faster mutual understanding than human-agent interaction.

Table 1: Human intervention in APRs vs HPRs.

PR Type	Accepted		Rejected	
	With Int. (%)	Without Int. (%)	With Int. (%)	Without Int. (%)
APR	2,151 (36.65 %)	1,277 (21.76%)	911 (15.52%)	1,530 (26.07%)
HPR	4,157 (70.83%)	702 (11.96%)	749 (12.76%)	245 (4.45%)

APRs with human intervention tend to involve more review effort than HPRs with human intervention. While human intervention occurs more frequently in HPRs than in APRs as shown in Table 1, APRs with human intervention, tend to involve more extensive review processes. As shown in Table 2, APRs with human intervention are larger, with longer duration than HPRs with intervention and involve more interactions. We observe that APRs with human intervention, showed higher median churn (196. vs. 66), longer median durations (42.9 vs 22.2), and longer PR descriptions (median 977 vs. 482) than HPRs with human intervention. This suggests that HPRs tend to resolve more quickly with less intensive review process than APRs.

Table 2: Comparison of code review metrics in APRs vs HPRs with and without intervention (Int.)

Metric	APR				HPR			
	With Int.		Without Int.		With Int.		Without Int.	
	Median	Mean	Median	Mean	Median	Mean	Median	Mean
NumberMessages	**2.50**	**3.65**	1.50	1.47	**1.50**	**2.22**	0.50	0.51
NumberInlineComments	**0.00**	**2.58**	0.00	0.18	**0.00**	**1.34**	0.00	0.03
NumberCommits	**3.50**	**5.71**	1.50	2.54	**2.00**	**4.54**	1.00	2.35
Churn	**196.00**	**3870.66**	76.00	1022.25	**66.00**	**828.20**	34.50	482.33
NumberOfModifiedFiles	**5.00**	**38.59**	2.50	12.20	**3.00**	**7.41**	2.00	5.18
Description Length	**977.00**	**1399.33**	512.50	925.51	**482.50**	**1850.16**	530.50	2855.75
Duration (Hours)	**42.90**	**139.41**	6.46	70.85	**22.20**	**160.11**	7.37	90.55

Human intervention in APRs often requires high effort even when PRs are rejected. The results in Table 3 show that, human intervention in APRs often involves high review effort, even when PRs are rejected. Rejected APRs with intervention have a median churn of 203, a median review duration of 71.06 hours, and a median of 3 commits. These values are higher than those observed for rejected HPRs, which have a median churn of 79, a median duration of 39.03 hours, and 2 commits. This shows that intervention in APRs can involve multiple revisions without necessarily leading to acceptance. Compared to rejected HPRs with intervention, HPRs tend to remain smaller and are closed more quickly.

3.2 RQ2: What types of interventions do humans perform on Agentic-PRs?

Our manual qualitative analysis revealed 4 high-level intervention categories, that include a set of 12 subcategories, and 30 intervention

Table 3: Comparison of code review metrics and PR outcomes of APRs vs HPRs with human intervention.

Metric	APR				HPR			
	Accepted		Rejected		Accepted		Rejected	
	Median	Mean	Median	Mean	Median	Mean	Median	Mean
NumberMessages	2.00	3.46	**3.00**	**3.83**	1.00	1.80	**2.00**	**2.64**
NumberInlineComments	0.00	3.03	**0.00**	**2.13**	0.00	1.68	**0.00**	**1.00**
NumberCommits	4.00	5.71	**3.00**	**5.70**	2.00	4.14	**2.00**	**4.94**
Churn	189.00	4224.97	**203.00**	**3516.34**	53.00	960.67	**79.00**	**695.73**
NumberOfModifiedFiles	5.00	43.06	**5.00**	**34.12**	3.00	6.62	**3.00**	**8.20**
Description Length	974.00	1351.10	**980.00**	**1447.56**	428.00	1243.16	**537.00**	**2457.15**
Duration (Hours)	14.73	83.89	**71.06**	**194.92**	5.37	65.19	**39.03**	**255.03**

sub-types. Figure 2, presents an overview of the resulting taxonomy, which consists of four high-level categories: (1) Guidance-level intervention, (2) Decision-level intervention, (3) Code change-level intervention, and (4) Operational-level intervention.

Guidance-level intervention: This is where developers do not directly change the code but provide feedback, constraints, or instructions to guide the agent's behavior. Guidance-level intervention is the most prevalent type of human intervention in APRs and accounts for 58,02% of all labelled interventions. This indicates that human contributors, engage more frequently in providing guidance, feedback, and clarification rather than directly making code changes themselves. This also suggests that the the agent's contributions, require more feedback and refinement rather than direct code changes or direct acceptance or rejection from human contributors. Within this category, Code quality & scope control is the most prevalent one with 35,45%, indicating that humans primarily intervene to constrain, refine, and align agent-generated changes with project standards. These interventions mainly involve asking the agent to remove unnecessary or redundant changes (34.78%), and enforcing project-specific conventions (29.56%), showing that agents frequently introduce unnecessary changes or fail to follow project conventions. Oversight & Feedback represents 31,55% of Guidance-level interventions and is mostly constituted by general discussion (57%). This suggests that developers frequently engage in iterative conversations to understand, evaluate, or reason about what the agent has done, rather than giving direct fixes. Task Management accounts for 22,40%, dominated by delegating tasks to the agents (73.39%), which indicates frequent human intervention in reassigning or restructuring tasks to guide agents toward desired outcomes, while Behavioral guidance represents a smaller portion at 10,58%, mainly involving requests to correct incorrect or hallucinated behavior of the agents (59.22%), indicating that explicit behavioral correction is less common than ongoing refinement and oversight.

Decision-level intervention: This is where developers decide the outcome of the PR whether to request changes, accept or reject the PR. This type of intervention represents the 2nd most common category, accounting for 21,16% of all labelled interventions. This suggests that humans tend to engage more often in guiding the agent's behavior and make decisions rather than directly editing the agent's code. This category is dominated by PR approvals, which represents 89.85% of cases. In contrast, PR rejections are lower at 10.14%, suggesting that while agents occasionally fail to meet project expectations, outright dismissal of agent-generated work is uncommon. This probably suggests that APRs are generally

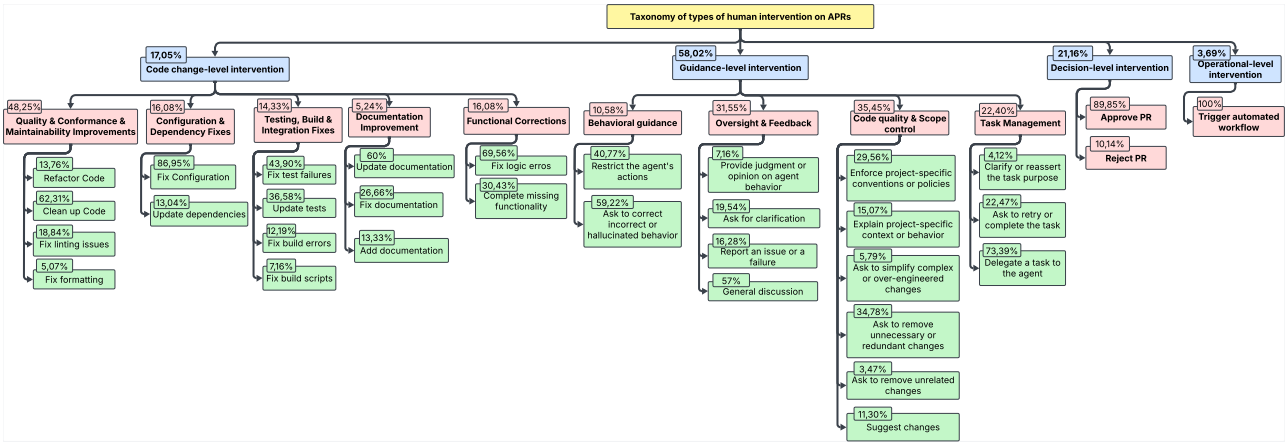


Figure 2: Taxonomy of human interventions performed on AI Agent Pull Requests (APRs)

brought to an acceptable state through earlier human interventions, reducing the need for final-stage rejection.

Code change-level intervention: This intervention accounts for 17,05% of all labelled interventions. This is where the developers directly edit the agent code to correct, refine, or complete the proposed changes. This indicates that, although agent contributions are useful, human developers still frequently intervene at the implementation level to provide code fixes and ensure completeness of the agent's solutions. Within this category, we observe that Quality & Conformance, & Maintainability Improvements constitutes 48.25% of all code change-level interventions. These interventions are mainly driven by code cleanup (62.31%), followed by lint fixing (18.84%) and code refactoring (13.76%). This indicates the agent code, often requires human oversight to improve readability and structure. Both functional corrections and Configuration & Dependency Fixes account for 16.08% of code change interventions. Functional corrections mainly focus on fixing logic errors (69.56%). Configuration & Dependency Fixes are dominated by Configuration Fixes (86.95%). Testing, Build & Integration Fixes, represents 14.34% of code change-level interventions and are mostly dominated by fixing test failures (43,90%) and updating tests (36,58%), which suggests that agents frequently introduce changes that break tests or that require test adjustments. Finally, Documentation Improvement, represents the lowest type of intervention, with 5.24%, and most documentation improvements consists in updating and fixing the documentation.

Operational-level intervention: This is where developers, use commands to trigger automated workflows (*e.g,* CI jobs, formatting, etc.). This type of intervention is the least one present out of all the interventions, accounting only for 3,69% of all interventions.

4 Conclusion and Takeaways

In this paper, we performed a quantitative and qualitative analysis of human interventions on agentic-PRs, that reveals various takeaways and practical implications.

Implication #1: APRs require more human review effort even when the intervention is less frequent than in HPRs. Our results show that human intervention occurs less frequently in APRs compared to HPRs. However, when intervention does occur in APRs, it often requires more review effort. Compared to HPRs that received human intervention, APRs tend to be larger, remain open for longer durations and involve more revisions. For practitioners, this highlights the importance of taking into account review costs of integrating AI coding agents into development workflows. This calls for the need to have mechanisms that can automatically triage APRs likely to require more human intervention to help focus reviewers attention and reduce unnecessary review efforts.

Implication #2: Human effort in APRs is mainly spent on guidance and oversight, not implementing or correcting code. Our taxonomy reveals that the majority of human interventions in APRs take the form of guidance-level actions rather than direct code changes. Developers most frequently intervene by enforcing project conventions and clarifying intent. For practitioners, this suggests that working with AI coding agents shifts developer effort away from implementation toward review, coordination, and oversight activities. Teams integrating these agents should therefore focus on review practices, clear project conventions, and communication mechanisms that support iterative guidance, rather than assuming that agents will reduce the need for human intervention.

Implication #3: Human intervention in APRs frequently addresses unnecessary, over-engineered, or misaligned agent outputs. The results of our taxonomy show that interventions also involve removing unnecessary changes, restricting the agents actions, simplifying over-engineered solutions, and correcting hallucinated agent behavior. This indicates that agents often generate changes that go beyond the intended task or rely on unsupported assumptions, requiring repeated human intervention to restrict and realign their actions. For practitioners, this suggests that improving how agent behavior is specified upfront may help reduce such intervention. One potential mechanism is the use of agent README files, which are project-level configuration files that guide their behavior, and which have been examined in prior work [5]. Clarifying task boundaries, and defining stricter permission related constraints in these files may help limit misaligned or complex agent outputs and reduce the need for repeated guidance level intervention during APR review.

References

[1] 2025. Replication Package for "Behind Agentic Pull Requests: An Empirical Study on Developer Interventions in AI Agent-Authored Pull Requests". https://figshare.com/s/5475e2d11c95edcba3dc. Anonymous link provided for review.

[2] Narjes Bessghaier, Ali Ouni, Mohammed Sayagh, Moataz Chouchen, and Mohamed Wiem Mkaouer. 2025. Towards understanding code review practices for infrastructure-as-code: An empirical study on OpenStack projects. *Empirical Software Engineering* 30, 3 (2025), 106.

[3] Narjes Bessghaier, Mohammed Sayagh, Ali Ouni, and Mohamed Wiem Mkaouer. 2023. What constitutes the deployment and runtime configuration system? An empirical study on openstack projects. *ACM Transactions on Software Engineering and Methodology* 33, 1 (2023), 1–37.

[4] Virginia Braun and Victoria Clarke. 2012. Thematic analysis. American Psychological Association. (2012).

[5] Worawalan Chatlatanagulchai, Hao Li, Yutaro Kashiwa, Brittany Reid, Kundjanasith Thonglek, Pattara Leelaprute, Arnon Rungsawang, Bundit Manaskasemsak, Bram Adams, Ahmed E Hassan, et al. 2025. Agent READMEs: An Empirical Study of Context Files for Agentic Coding. *arXiv preprint arXiv:2511.12884* (2025).

[6] Moataz Chouchen, Narjes Bessghaier, Mahi Begoug, Ali Ouni, Eman Alomar, and Mohamed Wiem Mkaouer. 2024. How do software developers use chatgpt? an exploratory study on github pull requests. In *Proceedings of the 21st International Conference on Mining Software Repositories*. 212–216.

[7] Flávia Coelho, Nikolaos Tsantalis, Tiago Massoni, and Everton LG Alves. 2021. An empirical study on refactoring-inducing pull requests. In *Proceedings of the 15th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*. 1–12.

[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] Ahmed E Hassan, Hao Li, Dayi Lin, Bram Adams, Tse-Hsun Chen, Yutaro Kashiwa, and Dong Qiu. 2025. Agentic Software Engineering: Foundational Pillars and a Research Roadmap. *arXiv preprint arXiv:2509.06216* (2025).

[10] Ahmed E Hassan, Gustavo A Oliva, Dayi Lin, Boyuan Chen, Zhen Ming, et al. 2024. Towards AI-native software engineering (SE 3.0): A vision and a challenge roadmap. *arXiv preprint arXiv:2410.06107* (2024).

[11] Kosei Horikawa, Hao Li, Yutaro Kashiwa, Bram Adams, Hajimu Iida, and Ahmed E Hassan. 2025. Agentic Refactoring: An Empirical Study of AI Coding Agents. *arXiv preprint arXiv:2511.04824* (2025).

[12] Stefano Lambiase, Gemma Catolino, Fabio Palomba, and Filomena Ferrucci. 2024. Motivations, Challenges, Best Practices, and Benefits for Bots and Conversational Agents in Software Engineering: A Multivocal Literature Review. *Comput. Surveys* 57, 4 (2024), 1–37.

[13] Hao Li, Haoxiang Zhang, and Ahmed E Hassan. 2025. The rise of ai teammates in software engineering (se) 3.0: How autonomous coding agents are reshaping software engineering. *arXiv preprint arXiv:2507.15003* (2025).

[14] Hao Li, Haoxiang Zhang, and Ahmed E. Hassan. 2025. The Rise of AI Teammates in Software Engineering (SE) 3.0: How Autonomous Coding Agents Are Reshaping Software Engineering. *arXiv preprint arXiv:2507.15003* (2025).

[15] Junwei Liu, Kaixin Wang, Yixuan Chen, Xin Peng, Zhenpeng Chen, Lingming Zhang, and Yiling Lou. 2024. Large language model-based agents for software engineering: A survey. *arXiv preprint arXiv:2409.02977* (2024).

[16] Moritz Mock, Jorge Melegati, and Barbara Russo. 2024. Generative AI for test driven development: preliminary results. In *International Conference on Agile Software Development*. Springer, 24–32.

[17] Sida Peng, Eirini Kalliamvakou, Peter Cihon, and Mert Demirer. 2023. The impact of ai on developer productivity: Evidence from github copilot. *arXiv preprint arXiv:2302.06590* (2023).

[18] Danilo Silva, Nikolaos Tsantalis, and Marco Tulio Valente. 2016. Why we refactor? confessions of github contributors. In *Proceedings of the 2016 24th acm sigsoft international symposium on foundations of software engineering*. 858–870.

[19] Miku Watanabe, Yutaro Kashiwa, Bin Lin, Toshiki Hirao, Ken'Ichi Yamaguchi, and Hajimu Iida. 2024. On the use of chatgpt for code review: Do developers like reviews by chatgpt?. In *Proceedings of the 28th International Conference on Evaluation and Assessment in Software Engineering*. 375–380.

[20] Miku Watanabe, Hao Li, Yutaro Kashiwa, Brittany Reid, Hajimu Iida, and Ahmed E Hassan. 2025. On the use of agentic coding: An empirical study of pull requests on github. *arXiv preprint arXiv:2509.14745* (2025).