Characterizing the Adoption of Autonomous Coding Agents in Functional Programming Repositories

Kayre Mendes
Federal Institute of Paraná (IFPR)
Telêmaco Borba, Brazil
kayrebianca2019@gmail.com

Jailton Coelho
Federal Institute of Paraná (IFPR)
Telêmaco Borba, Brazil
jailton.coelho@ifpr.edu.br

Abstract

Autonomous coding agents are increasingly participating in collaborative software development, yet little is known about their adoption in functional programming ecosystems. This paper presents an empirical study on the adoption of autonomous coding agents in GitHub repositories developed with functional programming languages. We analyzed 35 repositories written in Elixir, Haskell, F#, Erlang and OCaml, comprising 158 AI-generated Pull Requests. The results show that AI-generated contributions are already present in mature and actively maintained functional programming repositories. Most Pull Requests are small and incremental. OpenAI Codex accounts for 78 of the 93 AI-generated Pull Requests in Elixir repositories, whereas GitHub Copilot is responsible for 38 of the 39 Pull Requests identified in F# repositories. Elixir also presents the highest Pull Request acceptance rate (77.4%), while repository popularity exhibits no meaningful association with AI-generated Pull Request activity ($\rho = -0.029$ for stars and $\rho = -0.038$ for forks).

Keywords

Autonomous Coding Agents, Functional Programming, Software Engineering, Mining Software Repositories, GitHub, Empirical Study

1 Introduction

The rapid evolution of large language models has significantly transformed software development practices. Early advances demonstrated that large language models could effectively generate source code and support programming tasks [2], leading to the widespread adoption of AI-powered programming assistants such as GitHub Copilot [1]. More recently, autonomous coding agents have extended these capabilities by understanding development tasks, modifying source code, executing tests and autonomously submitting Pull Requests within collaborative software development workflows [5, 10]. Consequently, artificial intelligence is evolving from a code completion assistant into an active software engineering teammate capable of autonomously participating in software engineering activities [6, 7].

Several autonomous coding agents have recently emerged, including OpenAI Codex, GitHub Copilot, Claude Code, Devin and Cursor. Although these tools differ in their interaction models and levels of autonomy, they share the common objective of supporting software development through increasingly autonomous execution of programming activities [1, 2]. Their growing adoption has motivated a new line of empirical studies investigating AI-generated contributions in GitHub repositories, including the identification of coding agents, the characterization of AI-generated Pull Requests, and the adoption of agentic coding in collaborative software development [3, 8, 9, 13].

Despite these recent advances, little attention has been devoted to functional programming ecosystems [9]. Functional programming languages differ from imperative and object-oriented languages by emphasizing immutable data, higher-order functions and expressive type systems [11].

To address this gap, this paper presents an empirical study on the adoption of autonomous coding agents in GitHub repositories developed with functional programming languages. Our analysis investigates repository characteristics, Pull Request properties, coding agent distribution, Pull Request acceptance and the relationship between repository popularity and AI-generated Pull Request activity.

The main contributions of this paper are summarized as follows:

- We characterize GitHub repositories written in functional programming languages that adopt autonomous coding agents, considering repository maturity, popularity and development activity.
- We investigate the characteristics of AI-generated Pull Requests, including contribution size, code churn, review process and merge behavior.
- We analyze the distribution of autonomous coding agents across functional programming languages and evaluate how Pull Request acceptance differs among programming languages and coding agents.
- We investigate whether repository popularity is associated with AI-generated Pull Request activity using repository-level correlation analyses.

The remainder of this paper is organized as follows. Section 2 presents the adopted methodology. Section 3 discusses the empirical results. Section 4 presents the threats to validity, while Section 5 concludes the paper.

2 Methodology

This study is based on the AIDev dataset released as part of the MSR 2026 Mining Challenge [7]. The dataset provides GitHub repositories and Pull Requests associated with autonomous coding agents.

The original dataset was enriched with additional metadata collected through the GitHub REST API [4]. Repository-level attributes include repository age, size, stars, forks, watchers, subscribers, open issues and the date of the latest push. Pull Request metadata include commits, changed files, additions, deletions, review comments, merge status and timestamps used to compute review and merge times. The metadata collection process was automated through Python scripts, ensuring a consistent enrichment procedure across all analyzed repositories.

After repository selection, the resulting dataset comprises 35 repositories containing 158 AI-generated Pull Requests. The analyzed repositories correspond to mature and actively maintained software projects, with repository ages ranging from 13.2 to 202.6 months and a median of 78.5 months. Repository popularity is heterogeneous, with a median of 155 stars and 18 forks, whereas repository activity remains recent, with a median of seven days since the last push.

2.1 Repository Selection

Initially, only repositories whose primary programming language corresponded to one of the five functional programming languages considered in this study were selected: Elixir, Erlang, F#, Haskell and OCaml. These languages were selected according to the TIOBE Programming Community Index [12], considering the most popular functional programming languages consistently represented in the ranking.

Subsequently, additional filtering criteria were applied to exclude repositories with limited public adoption. Selected repositories were required to satisfy the following conditions:

- Explicitly identified software license;
- At least 10 GitHub stars;
- At least one GitHub fork.

These criteria are intended to retain repositories with observable community engagement while reducing the inclusion of personal, inactive or experimental projects, a common practice in mining software repository studies.

Finally, the selected repositories were enriched with additional metadata obtained through the GitHub REST API.

2.2 Research Questions

The empirical analysis is guided by the following research questions.

- **RQ1.** What are the characteristics of functional programming repositories adopting autonomous coding agents?
- **RQ2.** What are the characteristics of AI-generated Pull Requests contributed to functional programming repositories?
- **RQ3.** Which autonomous coding agents are adopted across functional programming languages?
- **RQ4.** How are AI-generated Pull Requests accepted in functional programming repositories?
- **RQ5.** Is repository popularity associated with AI-generated Pull Request activity?

RQ1 characterizes the repositories adopting autonomous coding agents by analyzing indicators related to repository maturity, popularity and development activity. RQ2 investigates the characteristics of AI-generated Pull Requests, including metrics associated with contribution size and review process. RQ3 analyzes how autonomous coding agents are distributed across the analyzed functional programming languages. RQ4 evaluates Pull Request acceptance through merge rates according to both programming language and coding agent. Finally, RQ5 investigates whether repository popularity is associated with AI-generated Pull Request activity using Spearman's rank correlation coefficient.

3 Results and Discussion

This section presents the results of the empirical analysis according to the research questions introduced in Section 2.2.

3.1 RQ1: What Are the Characteristics of Functional Programming Repositories Adopting Autonomous Coding Agents?

Figure 1 summarizes the distribution of repository age, popularity and development activity. The analyzed repositories correspond to mature software projects, with repository ages ranging from approximately one year to more than sixteen years. Although repository maturity varies across the sample, the observed distributions indicate that autonomous coding agents are not restricted to recently created repositories.

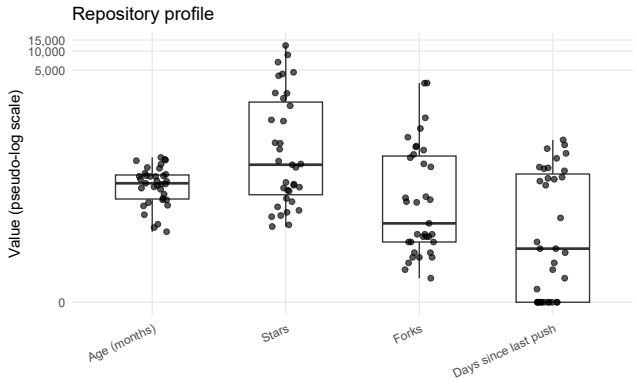

Figure 1: Repository profile of the analyzed functional programming repositories.

Repository popularity exhibits a typical long-tailed distribution. While the median repository contains 155 stars and 18 forks, a relatively small number of projects concentrate substantially higher popularity indicators. This dispersion reflects the presence of a few highly popular repositories acting as outliers, whereas most repositories remain concentrated around considerably lower popularity values. To improve visualization across repositories with heterogeneous popularity levels, the popularity axes employ a pseudo-logarithmic scale. Such behavior is commonly observed in open-source software ecosystems and suggests that AI-assisted development occurs in both moderately popular and highly visible repositories.

The analyzed repositories also remain actively maintained. Repository activity is relatively recent, with a median of only seven days since the last push. This observation indicates that autonomous coding agents are primarily associated with repositories undergoing continuous software evolution rather than inactive or archived projects.

Overall, repository popularity presents substantially greater variability than repository maturity. These observations suggest that autonomous coding agents are already being adopted across repositories with heterogeneous community visibility while remaining concentrated in actively maintained software projects.

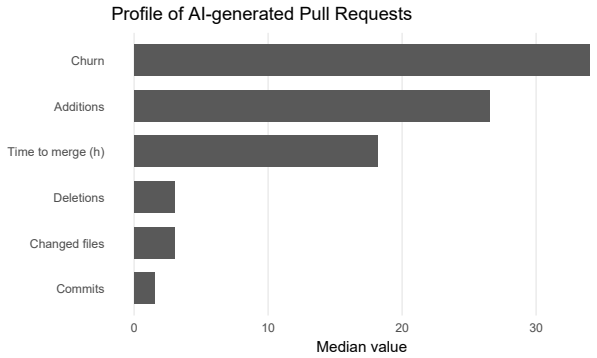

Figure 2: Characteristics of AI-generated Pull Requests.

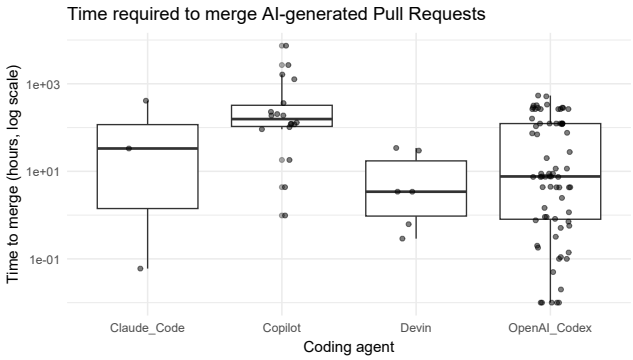

Figure 3: Distribution of merge time according to coding agent.

Summary. Autonomous coding agents are observed in mature and actively maintained repositories with heterogeneous popularity levels.

3.2 RQ2: What Are the Characteristics of AI-Generated Pull Requests Contributed to Functional Programming Repositories?

Figure 2 summarizes the principal characteristics of the analyzed Pull Requests. Overall, AI-generated Pull Requests correspond to relatively small software modifications. Most contributions involve only a few commits, modify a limited number of files and produce relatively low code churn.

This behavior suggests that autonomous coding agents are predominantly employed to support incremental software evolution, although they are not restricted to localized maintenance tasks. Larger Pull Requests affecting hundreds of files or thousands of modified lines are also observed. Although less frequent, these larger Pull Requests indicate that autonomous coding agents occasionally participate in broader development tasks rather than only localized software modifications.

Figure 3 illustrates the distribution of merge times according to the coding agent. Although several Pull Requests are merged within only a few hours, considerably longer review cycles are also observed. Such variability suggests that Pull Request integration depends not only on the coding agent itself but also on repository-specific review practices and project maintenance workflows.

Summary. AI-generated Pull Requests are typically small, incremental software modifications involving limited code churn and few changed files.

3.3 RQ3: Which Autonomous Coding Agents Are Adopted Across Functional Programming Languages?

We investigate how autonomous coding agents are distributed across the analyzed functional programming languages.

A total of five autonomous coding agents were identified in the selected repositories: OpenAI Codex, GitHub Copilot, Claude Code, Devin and Cursor. Figure 4 summarizes the distribution of AI-generated Pull Requests according to programming language and coding agent.

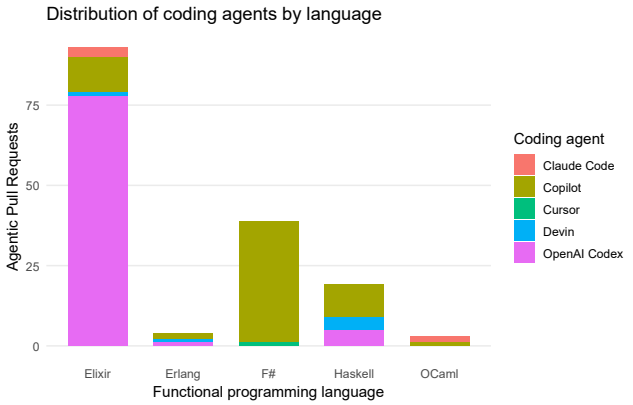

Figure 4: Distribution of AI-generated Pull Requests according to programming language and coding agent.

The results reveal that the adoption of autonomous coding agents is not homogeneous across functional programming ecosystems. OpenAI Codex is the predominant coding agent in Elixir repositories, accounting for 78 of the 93 AI-generated Pull Requests identified for this language. In contrast, GitHub Copilot predominates in F# repositories, where it is responsible for 38 of the 39 observed Pull Requests, and also represents the largest proportion of AI-generated contributions in Haskell repositories.

The remaining coding agents appear considerably less frequently. Devin contributes a small number of Pull Requests distributed across Elixir, Erlang and Haskell repositories, whereas Claude Code is observed only in Elixir and OCaml repositories. Cursor appears only once in the analyzed dataset, corresponding to a single Pull Request submitted to an F# repository.

These observations suggest that the adoption of autonomous coding agents varies substantially across functional programming ecosystems. While some languages exhibit a clear predominance of a single coding agent, others present a more diversified distribution of AI-assisted contributions. Such differences may reflect variations

in ecosystem maturity, developer preferences or the availability of language-specific support provided by each coding agent. Results involving Erlang, OCaml and less frequently observed coding agents should be interpreted cautiously due to the limited number of Pull Requests available for these groups.

Summary. OpenAI Codex predominates in Elixir repositories, whereas GitHub Copilot is the most frequently observed coding agent in F# and Haskell projects.

3.4 RQ4: How Are AI-Generated Pull Requests Accepted in Functional Programming Repositories?

Figure 5 summarizes Pull Request acceptance according to programming language.

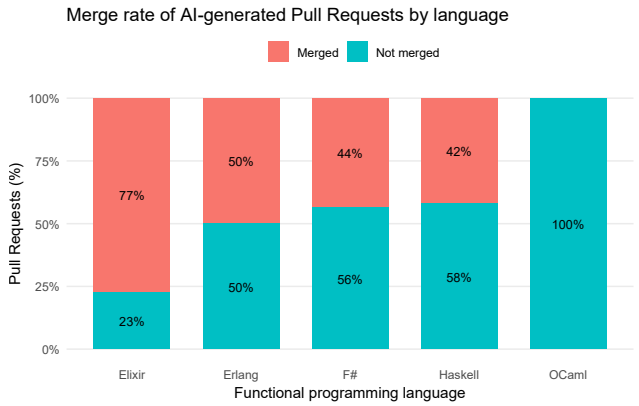

Figure 5: Merge rate of AI-generated Pull Requests by functional programming language.

Elixir repositories present the highest acceptance rate among the analyzed languages, with 72 of the 93 AI-generated Pull Requests successfully merged (77.4%). F# repositories merged 17 of the 39 Pull Requests (43.6%), whereas Haskell repositories merged 8 of the 19 contributions (42.1%). Erlang repositories merged two of the four observed Pull Requests, while no merged Pull Requests were identified for the analyzed OCaml repositories.

Although language-level results reveal different acceptance rates, they do not fully explain the observed behavior. Figure 6 further analyzes merge rates according to both programming language and coding agent.

OpenAI Codex presents the largest number of merged Pull Requests, particularly in Elixir repositories, where 68 of the 78 generated Pull Requests were successfully integrated. In contrast, GitHub Copilot exhibits considerably lower merge rates in several ecosystems, including F# (44.7%) and Haskell (10.0%). Devin presents a 100% observed merge rate; however, this result is based on only six Pull Requests and therefore should not be interpreted as evidence of superior performance. Similarly, Claude Code and Cursor appear only sporadically in the analyzed dataset.

Overall, Pull Request acceptance varies substantially across both programming languages and coding agents. Nevertheless, the relatively small number of Pull Requests associated with some coding

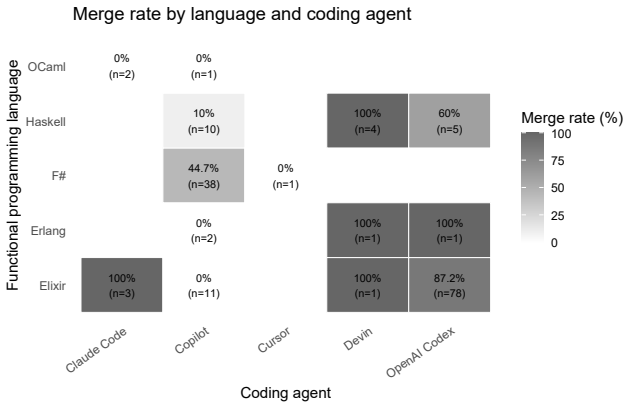

Figure 6: Merge rate according to programming language and coding agent. Values indicate the observed merge rate and the corresponding number of Pull Requests.

agents limits stronger conclusions regarding their comparative performance.

Summary. OpenAI Codex presents the highest number of successfully merged Pull Requests.

3.5 RQ5: Is Repository Popularity Associated with AI-Generated Pull Request Activity?

To evaluate this relationship, Spearman's rank correlation coefficient was computed between repository popularity indicators (stars and forks) and the number of AI-generated Pull Requests associated with each repository. The obtained coefficients indicate no meaningful monotonic association between repository popularity and AI-generated Pull Request activity. The correlation between stars and AI-generated Pull Requests is $\rho = -0.029$, while the correlation between forks and AI-generated Pull Requests is $\rho = -0.038$. Both coefficients are close to zero, suggesting that repository popularity alone does not explain the occurrence of AI-generated Pull Requests.

Figure 7 illustrates the relationship between repository stars and AI-generated Pull Request activity. Despite the presence of repositories with thousands of stars, these projects do not systematically receive larger numbers of AI-generated Pull Requests. Likewise, repositories with relatively modest popularity indicators may also exhibit considerable AI-assisted development activity.

These findings suggest that factors other than repository popularity influence the adoption of autonomous coding agents in functional programming repositories. Repository maintenance practices, contributor workflows and the availability of AI-assisted development tools are potential factors that may explain the observed variability and deserve further investigation.

Summary. No meaningful association was observed between repository popularity and AI-generated Pull Request activity.

4 Threats to Validity

As with any empirical study, this work is subject to threats to validity. Regarding *construct validity*, the analysis relies on the

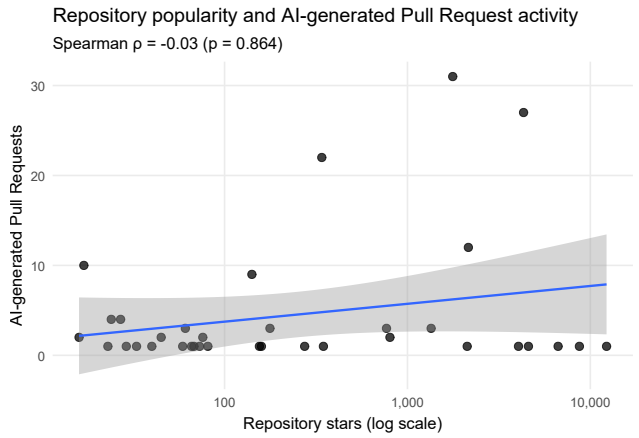

Figure 7: Relationship between repository popularity and AI-generated Pull Request activity. The fitted regression line is included only to facilitate visualization.

AIDev dataset released as part of the MSR 2026 Mining Challenge. Although the dataset was manually curated, the identification of AI-generated Pull Requests may still contain false positives or false negatives. Additionally, only repositories whose primary language corresponds to Elixir, Erlang, F#, Haskell or OCaml were considered.

Regarding *internal validity*, repository metadata were enriched using the GitHub REST API. Since GitHub repositories continuously evolve, the collected metadata represent a snapshot at the time of data collection. Furthermore, the adopted selection criteria (minimum number of stars, forks and an identified software license) intentionally exclude personal, experimental and low-visibility repositories.

Finally, concerning *external validity*, the results are limited to the 35 analyzed repositories and 158 AI-generated Pull Requests. Therefore, the findings should not be generalized to other functional programming languages or software ecosystems without further empirical investigation.

5 Conclusion

We analyzed 35 repositories containing 158 AI-generated Pull Requests. The obtained results indicate that autonomous coding agents are already present in mature and actively maintained functional programming repositories. AI-generated Pull Requests are predominantly incremental, although larger software modifications are also observed. The distribution of coding agents varies substantially across programming languages, with OpenAI Codex predominating in Elixir repositories and GitHub Copilot being the most frequently observed coding agent in F# and Haskell projects. Furthermore, Pull Request acceptance differs across programming languages and coding agents.

Overall, this study provides an initial characterization of autonomous coding agents in functional programming repositories. These findings establish a baseline for future empirical studies investigating AI-assisted software development in functional programming ecosystems.

As future work, we intend to extend the empirical analysis to additional functional programming languages and larger datasets, investigate the characteristics of AI-generated code at the source-code level, and analyze the long-term evolution of repositories adopting autonomous coding agents. Another promising research direction involves comparing functional and non-functional programming ecosystems to better understand whether programming paradigms influence the adoption and effectiveness of autonomous coding agents.

Artifact Availability

The research artifacts produced in this study are publicly available through Zenodo at https://doi.org/10.5281/zenodo.21267191.

Acknowledgements

We would like to thank IFPR.

References

[1] Christian Bird, Denae Ford, Thomas Zimmermann, Nicole Forsgren, Eirini Kalliamvakou, Travis Lowdermilk, and Idan Gazit. 2022. Taking Flight with Copilot: Early Insights and Opportunities of AI-Powered Pair-Programming Tools. *ACM Queue* 20, 6 (2022), 35–57. doi:10.1145/3582083

[2] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating Large Language Models Trained on Code. *CoRR* abs/2107.03374 (2021). doi:10.48550/arXiv.2107.03374

[3] Taher A. Ghaleb. 2026. Fingerprinting AI Coding Agents on GitHub. In *Proceedings of the 23rd International Conference on Mining Software Repositories (MSR '26)*. ACM. doi:10.1145/3793302.3793582

[4] GitHub. 2026. GitHub REST API Documentation. https://docs.github.com/en/rest. Accessed: 2026-07-08.

[5] Georgios Gousios, Martin Pinzger, and Arie van Deursen. 2014. An Exploratory Study of the Pull-Based Software Development Model. In *Proceedings of the 36th International Conference on Software Engineering (ICSE 2014)*. ACM, 345–355. doi:10.1145/2568225.2568260

[6] Ahmed E. Hassan, Gustavo Ansaldi Oliva, Dayi Lin, Boyuan Chen, and Zhen Ming Jiang. 2024. Towards AI-Native Software Engineering (SE) 3.0: A Vision and a Challenge Roadmap. *CoRR* abs/2410.06107 (2024). doi:10.48550/arXiv.2410.06107

[7] 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. *CoRR* abs/2507.15003 (2025). doi:10.48550/arXiv.2507.15003

[8] Daniel Ogenrwot and John Businge. 2026. How AI Coding Agents Modify Code: A Large-Scale Study of GitHub Pull Requests. In *Proceedings of the 23rd International Conference on Mining Software Repositories (MSR '26)*. ACM, 924–928. doi:10.1145/3793302.3793603

[9] Romain Robbes, Théo Matricon, Thomas Degueule, André C. Hora, and Stefano Zacchiroli. 2026. Agentic Much? Adoption of Coding Agents on GitHub. *ACM Transactions on Software Engineering and Methodology* (2026). doi:10.1145/3822180

[10] Margaret-Anne D. Storey, Alexey Zagalsky, Fernando Marques Figueira Filho, Leif Singer, and Daniel M. Germán. 2017. How Social and Communication Channels Shape and Challenge a Participatory Culture in Software Development. *IEEE Transactions on Software Engineering* 43, 2 (2017), 185–204. doi:10.1109/TSE.2016.2584053

[11] Simon Thompson. 2011. *Haskell: The Craft of Functional Programming* (3 ed.). Addison-Wesley.

[12] TIOBE Software. 2026. TIOBE Programming Community Index. https://www.tiobe.com/tiobe-index/. Accessed: 2026-07-08.

[13] Miku Watanabe, Hao Li, Yutaro Kashiwa, Brittany Reid, Hajimu Iida, and Ahmed E. Hassan. 2026. On the Use of Agentic Coding: An Empirical Study of Pull Requests on GitHub. *ACM Transactions on Software Engineering and Methodology* (2026). doi:10.1145/3798166