Understanding GitHub Action Developer Information Needs: An Empirical Study

1st Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

2nd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

3rd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

4th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

5th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

6th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

***Abstract*—The software industry extensively employs Continuous Integration and Delivery (CI/CD) practices to streamline software development. GitHub Actions (GA) provides a powerful automation framework within the GitHub ecosystem, allowing developers to automate workflows triggered by specific events. However, GA's complexity and varied use cases often require supplementary information beyond official documentation. This study examines GA developers' information needs as expressed on Stack Overflow. By analyzing numerous posts, we identified recurring themes in developers' queries, including troubleshooting, configuration specifics, and optimization strategies. Our findings show a significant increase in interest in GA over time, surpassing other CI/CD tools in post frequency and repository adoption rates. These insights highlight the critical need for comprehensive and accessible documentation, as well as robust community support, to address the complex challenges developers face. This research provides valuable insights for tool creators and documentation authors, offering a detailed categorization of developers' information needs and identifying key areas for enhancement, thereby improving the overall usability and effectiveness of GA resources.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

The software industry has widely adopted Continuous Integration and Delivery (CI/CD) practices to automate software engineering tasks [1]. These practices help minimize integration issues, enable frequent integration, automatically deploy changes, and speed up feedback loops for software developers [2]–[4].

Within the GitHub ecosystem, developers can leverage GitHub Actions (GA) to specify CI/CD. GA enable the automation of actions based on specific triggers, such as commits, pull requests, issues, comments, and more. This automation simplifies the process of building, testing, and deploying software projects, as developers can share actions across multiple repositories within the GitHub ecosystem.

Despite GA's robust automation capabilities, developers frequently need supplementary information to exploit its potential. This necessity arises because GitHub Actions (GA), like many complex tools, often involves intricate configurations and diverse use cases not comprehensively covered by the official documentation [5]. Consequently, developers seek information to troubleshoot issues, optimize workflows, or implement best practices. They often articulate these information needs in technical forums such as Stack Overflow (SO), asking specific questions and seeking answers to clarify various aspects of GA' s usage. Liu *et al.* emphasizes that developers express information needs at the sentence level [6], focusing on specific, granular details to address their issues effectively.

Consider the post titled *"Unexpected bash readable test result with GitHub Actions"*[1]. Here, the author expressed multiple needs, including understanding why a previously functioning GA workflow suddenly failed and identifying the cause of the error related to the configuration file. The author detailed the specific issues encountered, such as receiving an error indicating that `/etc/makepkg.conf` was not found, despite being able to `cat` the file. They also described steps taken, like adding debugging commands to inspect file permissions and directory structure, specifically noting an issue where a GA workflow that previously worked started failing without any changes. This example aligns with previous studies, which show that developers often seek information to understand how to implement specific tasks, indicating a need for additional guidance that is not readily available and hinders their progress [7].

Previous research has extensively explored developer information needs. Ko *et al.* found that developers often sought information about artifacts and coworkers [8], while Buse and Zimmermann highlighted the variability of these needs, emphasizing tailored tools [9]. Studies on similar software artifacts, such as Infrastructure as Code (IaC) and Configuration

[1]https://stackoverflow.com/q/73128465

as Code (CaC), have also been conducted, regarding the need for better support and documentation [10], [11]. Liu *et al.* stressed the importance of sentence-level analysis to capture developers' specific needs accurately, revealing multiple and complex needs within single posts [6]. Zhang *et al.* investigated GitHub Actions, presenting a comprehensive taxonomy of GA problems and solutions, but did not focus on sentence-level analysis or developers' information needs [7]. Therefore, none of these studies have been explicitly conducted regarding GA developers' information needs, nor at the granular level of sentences.

Through an empirical study, we present the identification and analysis of the diverse information needs of GA developers. Leveraging Liu *et al.*'s definition [12], we define GA developer information needs as questions related to GA posed on SO by individuals from diverse backgrounds (e.g., students, professional developers), collectively referred to as developers. We identify and categorize these information needs at the sentence level to provide a precise understanding of developers' queries. We conjecture that this fine-grained approach can better capture the essential features of GA developer needs, especially when multiple information needs are involved within a single SO post.

Our analysis revealed several key results. Among these, the most common information needs pertained to troubleshooting errors in GA workflows. Developers often faced issues with permissions, configuration files, and unexpected behavior in their CI/CD pipelines. Additionally, many posts highlighted the need for more detailed and accessible documentation, particularly regarding advanced configurations and optimization techniques. Moreover, developers frequently sought community best practices to improve the efficiency and reliability of their workflows, indicating a gap in the current official resources provided by GA.

Developers who use GitHub Actions for their CI/CD workflows are the primary beneficiaries of this research. By understanding their specific information needs at a granular level, tool creators and documentation writers can enhance the quality and accessibility of their resources. Additionally, this study benefits the broader software engineering community by providing insights into common challenges and solutions, thereby improving best practices, workflow optimizations, and overall developer productivity.

Our contributions are threefold: we provide a detailed categorization of GA developers' information needs based on sentence-level analysis, identify specific areas where GA documentation and community support can be improved, and offer insights that can guide future research and development of tailored support tools for GA users. This study addresses the gap between general documentation practices and the specific needs of developers, providing insights for more effective and efficient use of GitHub Actions in the software development process.

II. Research Questions

The goal of this study is to characterize the information needs of GA developers. Specifically, the purpose is to identify information need patterns that developers express on the SO platform. We propose two concrete research questions to guide our study, inspired by prior empirical research in the SO context [13]. With these RQs, we aim to better understand (i) the *current level of interest of CI/CD tools* (RQ1), and (ii) the *types of information needs* of GA developers (RQ2).

- **RQ1: How has the interest in GitHub Actions evolved within the GitHub ecosystem, and how does this compare to other CI/CD tools?** Building upon the work of Haj *et al.* [13], but with a specific focus on GA, this question aims to analyze the trend in SO posts related to GA and compare it with trends for other popular Continuous Integration tools. We hypothesize that interest in GitHub Actions has increased significantly over time, as reflected in a growing number of SO posts and repository adoption rates compared to other CI/CD tools. This analysis will help quantify the interest in GitHub Actions within the developer community.
- **RQ2: What are the most common information needs and challenges expressed by GitHub Actions users in Stack Overflow posts?** This research question seeks to characterize and categorize the types of information needs and challenges faced by GA developers. We hypothesize that the most frequently expressed needs relate to troubleshooting workflow errors, optimizing pipeline performance, and integrating GA with other tools and technologies, as these areas are critical for maintaining efficient and effective CI/CD pipelines. Understanding these needs can inform the development of targeted assistance tools and documentation, ensuring developers receive timely and relevant support throughout their GA implementation process.

III. Study Design

To establish the context for our research, we build on the method used by Liu *et al.* [6]. As illustrated in Figure 1, the relationships between the primary concepts discussed in this paper are central to understanding our approach. An SO [post] may include a [question] (composed of a title and body) along with multiple [answers]. The [question] can express various [developer information needs]. Each [developer information need] corresponds to a particular [developer information need type] and is articulated through [describing sentences] extracted from the [question]. These [describing sentences] contain the [relevant information] required to express the [developer information need]. Each piece of [relevant information] represents a specific [relevant information type]. Multiple [developer information needs] might be presented within a [question], with each serving a distinct [role] in describing the [developer information need] or its solution. TO DO ►*generar diagrama conceptual, agregar ejemplo que de cuenta de este esquema*◄

With this conceptual framework in place, we now provide an overview of our study design, structuring Sections III-A

and III-Baccording to the individual phases required to address each research question.

A. RQ1 - Level of interest on CI/CD Tools

Data collection: We collected SO posts related to CI/CD tools by querying the Stack Exchange Data Explorer[2] (SEDE) from January 2019 to October 2023. The extracted posts were saved in CSV files and grouped by month. Our tool selection criteria were based on the 2023 JetBrains TeamCity survey, which identified Jenkins, GitHub Actions, GitLab CI, Azure DevOps, CircleCI, and Travis CI as the six most popular CI/CD tools[3]. We identified the associated posts using the primary tags linked to each of these tools on Stack Overflow: "github-actions," "gitlab-ci," "jenkins," "azure-devops," "circleci," and "travis-ci." Although additional derived tags related to these tools exist, we focused on the parent tags to ensure consistent trend comparisons across posts.

As a result of this process, we extracted a total of 60,389 posts, distributed as follows: 9,622 with the tag "github-actions", 19,354 with "jenkins", 6,581 with "gitlab-ci", 22,797 with "azure-devops", 1,095 with "circleci", and 940 with "travis-ci.".

Approach: To address RQ1 we examined the evolution of interest in GA compared to other CI/CD tools. We tailored the approach of Yahmed *et al.* [14] to focus on GA and its counterparts.

To assess the growing interest in GitHub Actions (GA), we analyzed the frequency of questions related to various CI/CD tools (e.g., Jenkins, CircleCI, Travis CI) on Stack Overflow. Our main objective was to identify trends in the adoption and interest in GA over time. We leveraged two key metrics: popularity and difficulty.

Following the approach outlined by Ahmed *et al.* [14], we computed the number of related questions posted annually on Stack Overflow to measure popularity. Difficulty was similarly assessed using the percentage of questions with no accepted answer (%nAA) and the response time (RT) to receive an accepted answer, as established by previous studies [15]–[17]. These metrics were used to evaluate the community's engagement and the challenges associated with using GA.

To ensure that we were specifically capturing the interest and challenges associated with GA, we established a baseline by comparing these metrics to those of other CI/CD tools like Jenkins, GitLab CI, Azure DevOps, Travis CI, and CircleCI. The baseline was calculated by determining the average metrics for these other tools, which provided a reference point for comparison. For instance, the baseline for popularity was calculated as the average number of questions posted for Jenkins, GitLab CI, Azure DevOps, Travis CI, and CircleCI. Similarly, we calculated the baseline for difficulty by averaging the %nAA and RT for these tools.

Following previous work [14], we compared the metrics of GitHub Actions with a baseline to determine whether GA-related questions were more frequent, had higher percentages of questions without accepted answers, or took longer to receive accepted answers. For instance, if the average number of questions for GA was higher than the baseline, this would indicate that GA is more popular than other tools. Conversely, if GA had a higher percentage of questions without accepted answers or longer response times compared to the baseline, it would suggest that GA-related questions are more challenging or take longer to resolve.

This comparison allowed us to understand how GitHub Actions stands relative to other CI/CD tools, providing insights into whether GA poses distinct challenges or enjoys a higher level of interest within the developer community. By identifying these differences, we can better understand where additional support or resources may be needed to enhance the effectiveness of GA.

In the same fashion of Yahmed *et al.* [14], we conducted a correlation analysis using Spearman's rank correlation coefficient to examine the relationship between the popularity of GA-related questions and their corresponding difficulty. This analysis aimed to determine how community interest aligns with the challenges faced in using GA, helping us identify potential areas where additional support or resources may be necessary. **PV** ►*tuve que cambiar esta parte, para que el estudio sea coherente con la RQ2*◄

B. RQ2 - Information Needs

To address RQ2, we collected Stack Overflow (SO) posts related to GitHub Actions (GA). We downloaded the Stack Overflow data dump from the official Stack Exchange repository in December 2023[4], which served as our primary source. Following Yahmed *et al.*'s approach [14], we cloned a local database to perform detailed queries. We queried tags, titles, and body content to extract relevant posts, ensuring comprehensive coverage and relevance.

We used Stack Overflow's search interface[5] to identify relevant tags. Then we queried the tags table with the term "github actions," which yielded nine relevant tags: "github actions," "including: github-actions," building-github-actions," "github-actions-self-hosted-runners," "github-actions-runners," "github-actions-services," "github-actions-artifacts," "github-actions-reusable-workflows," "github-actions-workflows," and "github-actions-marketplace."

We also queried post titles and bodies independently for mentions of GitHub Actions or its variations. We applied text normalization rules to account for different capitalizations and hyphenations (e.g., "github actions," "github-actions," "Github actions," "Github-actions," "Github Actions," "Github-Actions"). From the results, we generated CSV files for further analysis.

We created three datasets based on the queries: tags, post titles, and post bodies. The dataset for tags contains posts identified by relevant tags, while the dataset for post titles includes posts where the title mentions GitHub Actions or its

[2]https://data.stackexchange.com/

[3]https://blog.jetbrains.com/teamcity/2023/07/best-ci-tools/

[4]https://archive.org/download/stackexchange_20231208

[5]https://stackoverflow.com/tags

variations. Similarly, the dataset for post bodies includes posts where the body mentions GitHub Actions or its variations. We intersected these three datasets using post IDs, merging them by ID, and discarding duplicates.

This process resulted in a total of 2,903 posts, encompassing posts from January 2019 to December 2023 (See Table I). Each entry provided details such as the type of post (e.g., question, answer, or date), creation date, tags, title, and body. Questions included between one and five tags related to their topics and may have an accepted answer, indicating a satisfactory response from the question's owner. Table I summarizes the dataset statistics.

TABLE I: Summary of Dataset Statistics. PV ►*Sandro, podrías completar esta tabla?*◄

Statistic	Value
Total posts	2,903
Time span of posts	XXXX to Present
Number of tags per question	1 to 5
Posts with accepted answers	[Statistical value]
Posts identified by tags	[Number]
Posts identified by titles	[Number]
Posts identified by bodies	[Number]

From the mentioned dataset we select a random sample of 400 posts. After removing posts with broken links, our final set ends with 340 posts. We parsed `HTML` post content into plain text and segmented into sentences. The parsing process accounted for code blocks, enumerations, hyperlinks, and figures. Finally, the content was divided into individual sentences.

Analysis techniques: To characterize the information needs of GA developers, we perform an open coding [18] , followed by an open card sorting analysis [19]. We leveraged the aforementioned dataset sample of 340 SO posts related to GA. From this dataset, we randomly selected a subset of 50 posts to ensure a manageable and representative sample. This subset included 463 sentences. PV ►*Confirmar estos numeros con Sandro*◄

Two classifiers manually analyzed these sentences. Initially, sentences that were irrelevant for expressing a DN or too implicit to understand without significant context were excluded. Through open coding, the classifiers identified 146 sentences as relevant. These sentences were then grouped into ten preliminary categories after two rounds of classification.

We subsequently reviewed the remaining sentences from other posts to ensure comprehensive coverage. During this review, we noticed that some sentences needed re-grouping as they did not fit neatly into the initial categories. We iteratively refined the process by re-evaluating and reclassifying sentences as necessary, ensuring that all sentences were accurately categorized. We continued this approach until we achieved a stable and coherent classification.

We ultimately organized the sentences into Relevant Information (RI) categories and refined their definitions for clarity. We then grouped these RI categories into corresponding Developer Needs, each with a specific definition.

Evaluation: To ensure the reliability of our taxonomy, we conducted an evaluation procedure outlined by Kaplan *et al.* [20], which comprises three comprehensive stages. First, we assessed the structural suitability by evaluating five key parameters: laconicity, completeness, lucidity, soundness, and orthogonality. Laconicity ensures that the taxonomy is concise without unnecessary complexity. Completeness guarantees that all relevant categories are covered. Lucidity confirms that the taxonomy is clear and easily understood. Soundness checks the internal consistency of the categories, and orthogonality ensures that the categories are mutually exclusive and collectively exhaustive.

Second, we measured inter-annotator agreement and calculated precision and recall. In this stage, three experts independently classified the categories, and their results were compared with ours. Additionally, the experts completed a survey to assess the practical usability of the taxonomy, providing insights into its effectiveness and ease of use. This evaluation not only validated the taxonomy's consistency but also identified areas for improvement.

Third, we examined the fraction of relevant classes, innovation, adaptation, and classification delta. Assessing the fraction of relevant classes ensures the taxonomy focuses on pertinent categories. Innovation measures the taxonomy's ability to incorporate new concepts. Adaptation evaluates how well the taxonomy adjusts to different contexts, and classification delta measures changes over time, indicating the taxonomy's flexibility and relevance. PV ►*Sandro: al parecer el último paso no es lo que hicimos?*◄

IV. Analysis & Results

PV ►*working on this section, please wait*◄

A. RQ1: Level of interest

The monthly number of questions posted by developers on StackOverflow is illustrated in Figure 1. We can highlight two distinct periods: between 2019 and early 2023, and from mid-2023 onwards. In the first period, we observed a decline in the number of posts for tools like Jenkins and Azure DevOps, while GitHub Actions, released on November 13th, 2019, surged in relevance, eventually matching or surpassing these tools. This rapid growth in popularity among the developer community corroborates the findings from [21], which identified GitHub Actions as the most used tool in Continuous Integration. Notably, the number of questions for GitHub Actions reached over 350 per month.

In the second period, starting from mid-2023, there was a general decline in the number of questions across all tools, including GitHub Actions, which dropped to about 200 questions per month. This trend is not unique to GitHub Actions; similar pattern were observed for other CI tools. This decline could be attributed to the increasing popularity of large language model-based tools, such as ChatGPT. Therefore, despite the reduction in the number of questions about GitHub Actions, this could be a global effect of new querying tools rather than a decrease in interest in GitHub Actions itself.

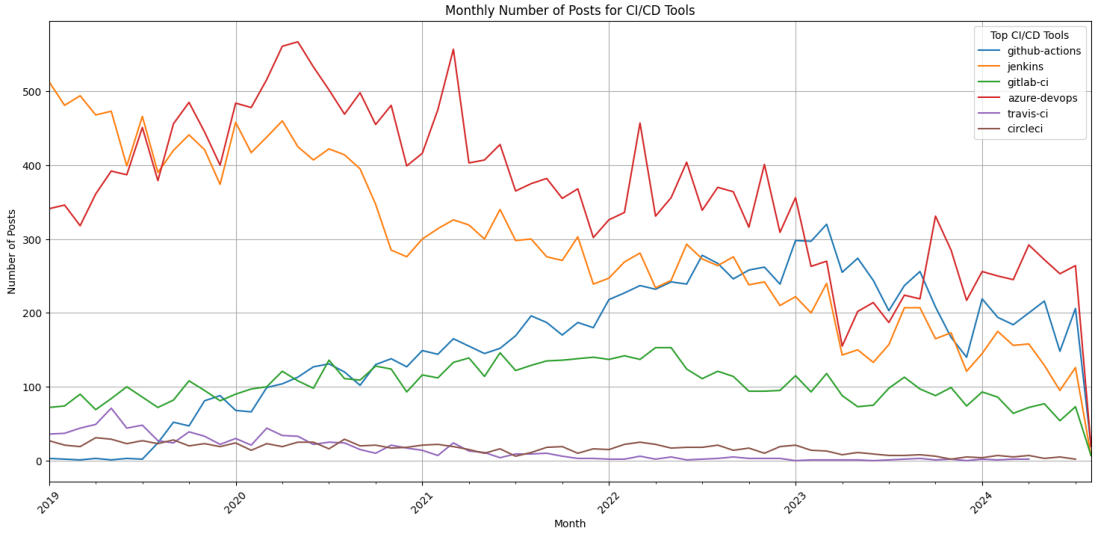

Fig. 1: Monthly number of questions with 'github-actions', 'jenkins', 'gitlab-ci', 'azure-devops', 'circleci', and 'travis-ci' tags correspondly in Stack Overflow. PV ►*probando tamaño*◄

B. RQ2 - Information Needs

In our study, we analyzed a total of 340 posts, which comprised 3,176 sentences. This sample size provides us with a 95% confidence level and a 5% margin of error for our conclusions. Out of these sentences, we identified 1,007 that corresponded to one or more types of Relevant Information (RI). As illustrated in Table V, 1 post did not contain any sentences that could be classified into an RI category. In contrast, 4 posts contained sentences with up to 5 different types of RIs, while the majority of posts (152) contained sentences with 2 types of RIs.

For instance, post 68346302 did not have any identifiable RI due to the loss of information in the form of images, snipets of code, or links during the coding process. Similarly, post XXX1 lacked clarity regarding the questioner's intentions, making it difficult to categorize. Utilizing these posts, we examined the occurrence and frequency of various RI categories. The findings, presented in Table **??**, show the number of sentences and posts associated with each RI category. It is noteworthy that some posts contained multiple sentences with the same RI type. For additional context, Table III provides definitions and examples of these categories.

Our analysis revealed that the most common RI category is *Implementation Goal* (FI1), present in over 152 posts. In this category, users describe their objectives or directly inquire about how to implement specific functionality. The second most common category is *Learning Specific Functions* (LE1), identified in 111 posts, where users seek to learn how to perform specific tasks using GitHub Actions.

The relationship between RIs and Developer Needs (DNs) can be observed in Figure 2, while Table II provides detailed definitions, the number of posts, and the corresponding percentages for each category. *Error Handling* (EH) emerged as the most frequently occurring DN, found in 179 out of 340 posts (52.65%).

Following EH, the next most common DNs were *Functionality Implementation* (FI), *Orientation* (OR), and *GHA Learning* (LE), with percentages of 44.71%, 38.82%, and 35.88% respectively. These results suggest that developers often articulate their implementation goals or seek to learn how to use the tool in their posts. However, considering the overall frequency of DNs, FI and LE, although commonly mentioned, are not the most prevalent; other needs such as EH and OR are more dominant.

EH typically involves the developer's need to address errors or failures within their implementation. This is likely because EH is a cross-cutting DN that frequently appears alongside other DNs, as evident from the co-occurrence measurements. In Figure 4, we calculated the overlap coefficient, also known as the Szymkiewicz-Simpson coefficient, to measure the similarity between two sets. This coefficient is defined as the size of the intersection divided by the size of the smaller set:

$$O(A, B) = \frac{|A \cap B|}{\min(|A|, |B|)}$$

where $|A \cap B|$ is the number of posts labeled with both DNs A and B, and $\min(|A|, |B|)$ is the number of posts in the smaller set. The coefficient ranges from 0 to 1, where 1 indicates complete overlap and 0 indicates no overlap.

We observed that EH frequently co-occurs with other DNs, with the highest overlap percentage being with *Incompatibility* (IN) at 83%. This might be because Incompatibility can be considered a type of EH, but we chose to separate it as a special case of error handling due to its specific relevance to GitHub Actions, where issues arise locally but not in the tool.

For instance, in post 7281996, sentences like "It says that it can't find the -CODE- package, but the package is there." are categorized as EH5. Similarly, "I'm setting up the CI/CD

process for a NextJS application and everything works fine locally, but when I run the workflow in GitHub Actions, the command -CODE- fails." represents an implementation incompatibility between local and GA execution. To maintain a comprehensive taxonomy, we decided to keep EH and IN separate.

Additionally, there is a significant overlap between LE and MI (58%). *Migration* (MI) involves the developer's need to transition from one CI/CD platform to GA, which can coincide with LE or OR. The overlap with OR is also relatively high (33%). Nonetheless, we deemed it important to identify MI separately as a special case relevant to GA.

V. Discussion

VI. Related Work

VII. Threats to Validity & Limitations

VIII. Conclusion

Acknowledgment

References

[1] M. Golzadeh, A. Decan, and T. Mens, "On the rise and fall of CI services in GitHub," in *2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE, 2022, pp. 662–672.

[2] M. Fowler and M. Foemmel, "Continuous integration," 2006.

[3] J. Humble and D. Farley, *Continuous delivery: reliable software releases through build, test, and deployment automation*. Pearson Education, 2010.

[4] B. Fitzgerald and K.-J. Stol, "Continuous software engineering: A roadmap and agenda," *Journal of Systems and Software*, vol. 123, pp. 176–189, 2017.

[5] M. Wessel, J. Vargovich, M. A. Gerosa, and C. Treude, "Github actions: the impact on the pull request process," *Empirical Software Engineering*, vol. 28, no. 6, p. 131, 2023.

[6] M. Liu, X. Peng, A. Marcus, S. Xing, C. Treude, and C. Zhao, "Api-related developer information needs in stack overflow," *IEEE Transactions on Software Engineering*, vol. 48, no. 11, pp. 4485–4500, 2021.

[7] Y. Zhang, Y. Wu, T. Chen, T. Wang, H. Liu, and H. Wang, "How do developers talk about GitHub actions? evidence from online software development community," in *Proceedings of the 46th IEEE/ACM International Conference on Software Engineering*, 2024, pp. 1–13.

[8] A. J. Ko, R. DeLine, and G. Venolia, "Information needs in collocated software development teams," in *29th International Conference on Software Engineering (ICSE'07)*. IEEE, 2007, pp. 344–353.

[9] R. P. Buse and T. Zimmermann, "Information needs for software development analytics," in *2012 34th International Conference on Software Engineering (ICSE)*. IEEE, 2012, pp. 987–996.

[10] A. Ouni, I. Saidani, E. Alomar, and M. W. Mkaouer, "An empirical study on continuous integration trends, topics and challenges in stack overflow," in *Proceedings of the 27th International Conference on Evaluation and Assessment in Software Engineering*, 2023, pp. 141–151.

[11] A. Rahman, A. Partho, P. Morrison, and L. Williams, "What questions do programmers ask about configuration as code?" in *Proceedings of the 4th International Workshop on Rapid Continuous Software Engineering*, 2018, pp. 16–22.

[12] M. Liu, X. Peng, A. Marcus, S. Xing, C. Treude, and C. Zhao, "Api-related developer information needs in stack overflow," *IEEE Transactions on Software Engineering*, vol. 48, no. 11, pp. 4485–4500, 2022.

[13] A. H. Yahmed, A. A. Abbassi, A. Nikanjam, H. Li, and F. Khomh, "Deploying deep reinforcement learning systems: a taxonomy of challenges," in *2023 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 2023, pp. 26–38.

[14] Z. Chen, Y. Cao, Y. Liu, H. Wang, T. Xie, and X. Liu, "A comprehensive study on challenges in deploying deep learning based software," in *Proceedings of the 28th ACM joint meeting on European software engineering conference and symposium on the foundations of software engineering*, 2020, pp. 750–762.

[15] A. Abdellatif, D. Costa, K. Badran, R. Abdalkareem, and E. Shihab, "Challenges in chatbot development: A study of stack overflow posts," in *Proceedings of the 17th international conference on mining software repositories*, 2020, pp. 174–185.

[16] M. Alshangiti, H. Sapkota, P. K. Murukannaiah, X. Liu, and Q. Yu, "Why is developing machine learning applications challenging? a study on stack overflow posts," in *2019 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*. IEEE, 2019, pp. 1–11.

[17] S. Ahmed and M. Bagherzadeh, "What do concurrency developers ask about? a large-scale study using stack overflow," in *Proceedings of the 12th ACM/IEEE international symposium on empirical software engineering and measurement*, 2018, pp. 1–10.

[18] K. Charmaz, *Constructing grounded theory: A practical guide through qualitative analysis*. sage, 2006.

[19] P. Morville and L. Rosenfeld, *Information architecture for the World Wide Web: Designing large-scale web sites*. " O'Reilly Media, Inc.", 2006.

[20] A. Kaplan, T. Kühn, S. Hahner, N. Benkler, J. Keim, D. Fuchß, S. Corallo, and R. Heinrich, "Introducing an evaluation method for taxonomies," in *Proceedings of the 26th International Conference on Evaluation and Assessment in Software Engineering*, 2022, pp. 311–316.

[21] T. Blog. (2023) Best continuous integration tools for 2023 – survey results. [Online]. Available: https://blog.jetbrains.com/teamcity/2023/07/best-ci-tools/

DN_id	Developer Need	Definition	N° of Posts	Percentage
EH	Error Handling	The developer identifies an issue within their code that causes unexpected behavior or failures and seeks solutions to diagnose and resolve the error.	182	53.53%
FI	Functionality Implementation	The developer aims to design and implement new features or enhancements within their project using GitHub Actions to automate workflows and processes.	153	45.00%
OR	Orientation	The developer looks for advice, best practices, or recommendations on how to proceed with a particular task or decision within their project, using GitHub Actions.	133	39.12%
LE	GHA Learning	The developer is looking to acquire knowledge and understanding of GitHub Actions, requiring documentation, tutorials, or examples to learn how to effectively use its features and capabilities.	123	36.18%
II	Insufficient Implementation	The developer finds that their current implementation falls short of the desired functionality or specifications, necessitating further enhancements or modifications.	86	25.29%
IN	Incompatibility	The developer's code functions correctly in their local environment but encounters issues or fails to execute as expected when run within GitHub Actions.	48	14.12%
MI	Migration	The developer seeks to transition their continuous integration and continuous deployment (CI/CD) processes from another platform to GitHub Actions, ensuring compatibility and functionality during the migration.	12	3.53%
AS	Alternative Solution	The developer has an existing solution in place but is interested in exploring different methods or tools that might offer better performance, efficiency, or simplicity.	8	2.35%

TABLE II: Detailed Taxonomy of Developer Needs with Post Counts and Percentages

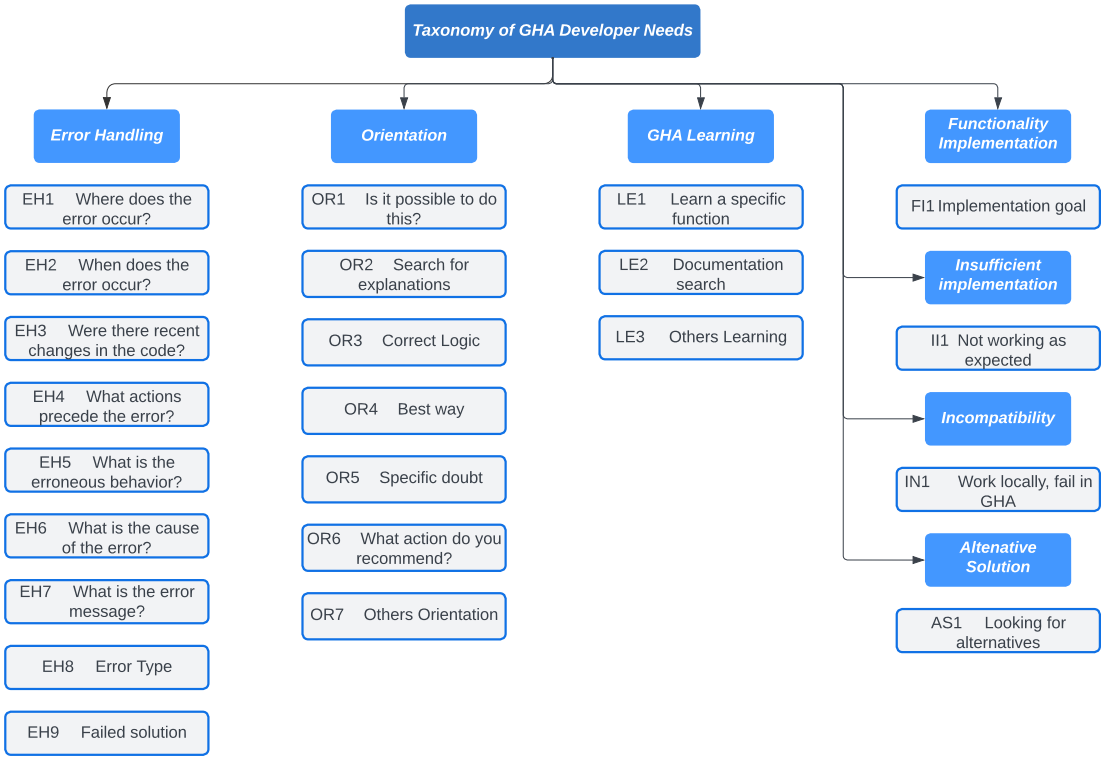

Fig. 2: Information Needs (IN), their Definitions, and Associated Relevant Information (RI)

RI_id	Relevant Information	Definition	Example
EH1	Where does the error occur?	This includes specific locations in the code such as functions, steps, jobs, stages, or modules where the error manifests.	The line that fails is: -CODE-.
EH2	When does the error occur?	This details the timing of the error, whether it happens at execution, after a certain period, or at the end. It also includes whether the error is constant or intermittent.	The 'strange' thing is that sometimes the test passes and sometimes it doesn't.
EH3	Were there recent changes in the code?	This involves describing any recent changes made by the developer to the code before the error appeared. These changes could be relevant to understanding the cause of the error.	I'm switching a Python project over to poetry for dependency and packaging management, and am running into issues getting my GitHub Actions unit tests working.
EH4	What actions precede the error?	This includes the specific actions or parts of the code that are executed just before the error occurs.	The build job passes but the deploy one fails.
EH5	What is the erroneous behavior?	This describes the incorrect behavior exhibited by the code that indicates an error.	The problem is when I try to use cache I see that the version in -CODE- and -CODE- are always changed so I can't use real cache here.
EH6	What is the cause of the error?	This provides a description of the possible reason or cause behind the error.	Running tests on GitHub Actions with FastAPI fails due to it trying to connect to hosted DB first.
EH7	What is the error message?	This includes the exact error message received, which can help in diagnosing the problem.	This is the error I am getting: -CODE-.
EH8	Error Type	Specifies the type of error encountered (e.g., syntax error, runtime error, etc.).	Running Angular e2e tests using GitHub Actions throws 'DevToolsActivePort file doesn't exist' error.
EH9	Failed solution	This includes explicit or implicit information about any attempted solutions that failed to resolve the issue.	I tried adding -CODE- in the Workflow YAML file, but it did not get reflected in how the container was created and the command still failed.
FI1	Implementation goal	This describes the specific features or goals of the implementation.	I'm trying to set up codecov monitoring for a public R package, where GitHub Actions will run -CODE-.
OR1	Is it possible to do this?	This includes queries about the feasibility of performing a specific action.	Is there a way to install ODBC drivers on github actions?
OR2	Search for explanations	This involves looking for explanations or reasons why something is not happening as expected.	Why does GitHub actions rest API download artifacts by creating a temporary URL?
OR3	Correct logic	Asking if the logic followed is correct. Questioning whether the approach or assumptions and actions are appropriate.	Should I create another entry in my matrix that only relies on the second branch?
OR4	Best way	Asking what the best or correct path is to take to accomplish something.	What's the best way to test my app using GitHub actions?
OR5	Specific doubt	Asking about a specific question related to GitHub Actions functions, policies, or behaviors.	Do I maybe need to add an authToken or something else?
OR6	What do you recommend?	Asking for recommendations for their specific need.	If I wanted to run an arbitrary command and make a PR to the repository, which GitHub Actions should I be looking at instead of reinventing my own Actions?
LE1	Learning specific functions	Asking how to perform specific actions using GitHub Actions.	How can I reference multi-line secrets in GitHub Actions?
LE2	Documentation search	Searching for documentation or examples related to specific GitHub Actions features or commands.	Could you please provide an example for me to refer?
II1	Not working as expected	This describes situations where the developer's implementation does not meet their requirements or functions as expected.	I've found a starter workflow that runs the Gradle build on a commit but I haven't found a way to report the Checkstyle errors as pull-request annotations.
IN1	Work locally, fail in GHA	This indicates that the code works correctly in the local environment but fails when executed in GitHub Actions.	Running it locally works fine but once I push it to our repository in our company domain, I get error due to some extra headers missing.
MI1	Change CI/CD platform	This involves migrating the CI/CD code to GitHub Actions.	I am transitioning my CI/CD over to GitHub Actions and noticed that my prior scripts for deploying Firebase do not work.
AS1	Looking for alternatives	Seeking for alternative approaches to achieve a goal or solve a problem without doing what is already known.	I know that I can split them up into separate repositories and solve it that way, but I am looking for a solution where I don't have to do that.

TABLE III: Detailed Taxonomy of Relevant Information for GitHub Actions

RI_id	Relevant Information	N° of Sentences	N° of Posts
EH1	Where does the error occur?	98	78
EH2	When does the error occur?	8	6
EH3	Were there recent changes in the code?	4	4
EH4	What actions preceded the error?	14	14
EH5	What is the erroneous behavior?	80	70
EH6	What is the cause of the error?	11	10
EH7	What is the error message?	67	61
EH8	Error Type	63	50
EH9	Failed solution	61	50
FI1	Implementation goal	195	153
OR1	Is it possible to do this?	69	60
OR2	Search for explanations	39	32
OR3	Correct Logic	18	17
OR4	Best way	9	9
OR5	Specific doubt	25	21
OR6	What action do you recommend?	2	2
OR7	Others Orientation	16	16
LE1	Learning Specific Functions	142	112
LE2	Documentation Search	6	6
LE3	Others Learning	23	14
II1	Not working as expected	102	86
IN1	Work locally, fail in GHA	63	48
MI1	Change CI/CD platform	14	12
AS1	Looking for alternatives	8	8
NR	Non-Relevant sentences	2169	—
—	Sentences with any RI	1007	—
—	Total number of sentences	3176	—

TABLE IV: Number of sentences and posts associated to a Relevant Information.

N° of DN Classes	Post Count
0	1
1	75
2	153
3	84
4	23
5	4

TABLE V: Number of DN Categories and Corresponding Post Counts

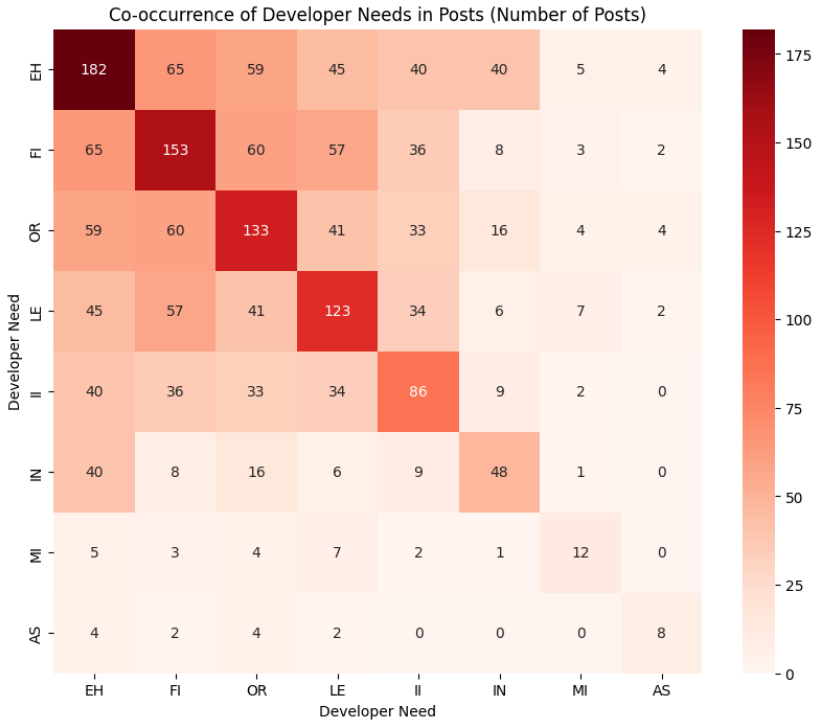
Co-occurrence of Developer Needs in Posts (Number of Posts)
Developer Need
EH
FI
OR
LE
II
IN
MI
AS
182 65 59 45 40 40 5 4
65 153 60 57 36 8 3 2
59 60 133 41 33 16 4 4
45 57 41 123 34 6 7 2
40 36 33 34 86 9 2 0
40 8 16 6 9 48 1 0
5 3 4 7 2 1 12 0
4 2 4 2 0 0 0 8
175
150
125
100
75
50
25
0

Fig. 3

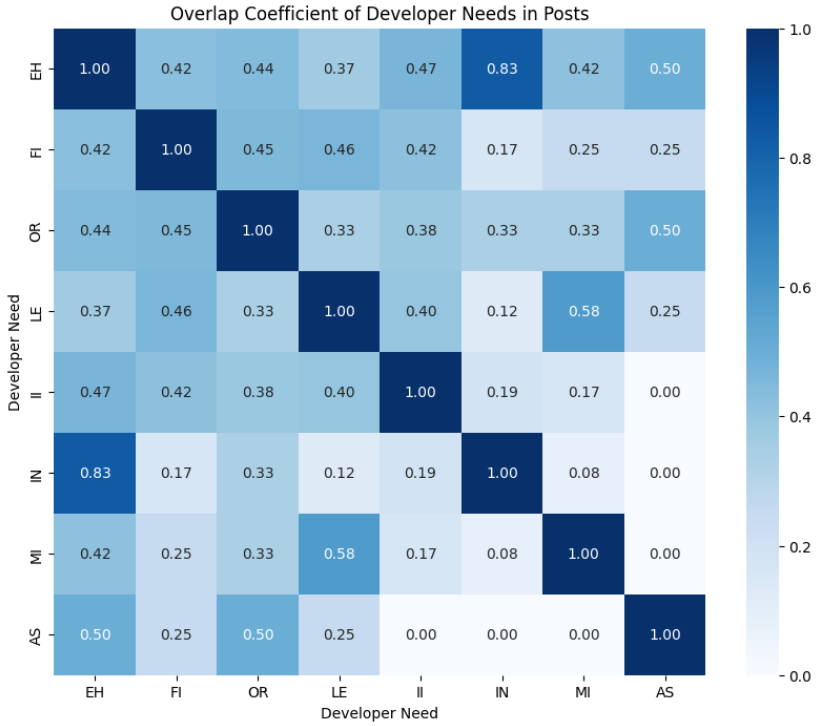
Overlap Coefficient of Developer Needs in Posts
Developer Need
EH
FI
OR
LE
II
IN
MI
AS
1.00 0.42 0.44 0.37 0.47 0.83 0.42 0.50
0.42 1.00 0.45 0.46 0.42 0.17 0.25 0.25
0.44 0.45 1.00 0.33 0.38 0.33 0.33 0.50
0.37 0.46 0.33 1.00 0.40 0.12 0.58 0.25
0.47 0.42 0.38 0.40 1.00 0.19 0.17 0.00
0.83 0.17 0.33 0.12 0.19 1.00 0.08 0.00
0.42 0.25 0.33 0.58 0.17 0.08 1.00 0.00
0.50 0.25 0.50 0.25 0.00 0.00 0.00 1.00
1.0
0.8
0.6
0.4
0.2
0.0

Fig. 4

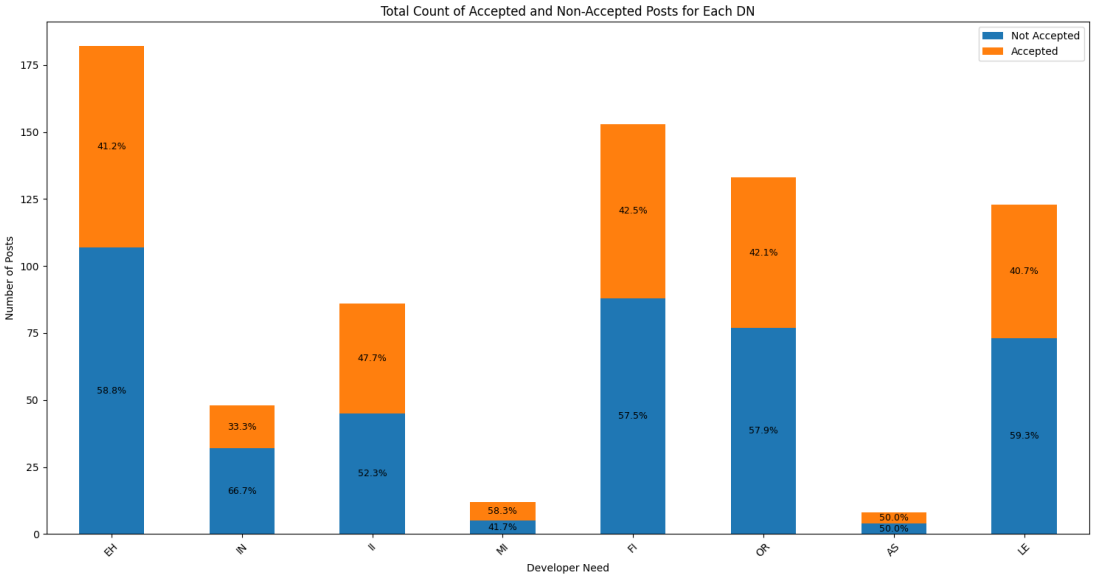

Fig. 5

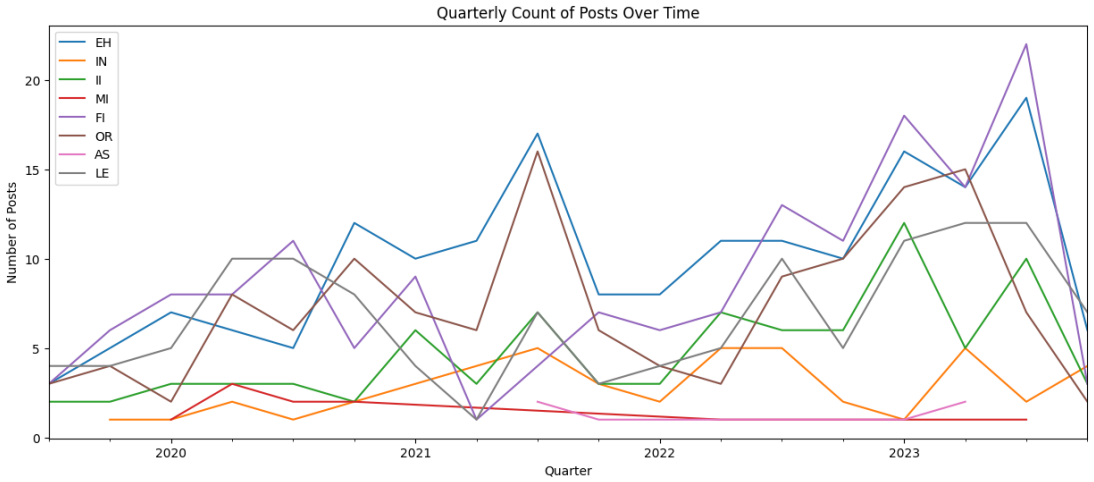

Fig. 6: Line plot showing the quarterly count of posts for each Developer Need (DN) category over time.

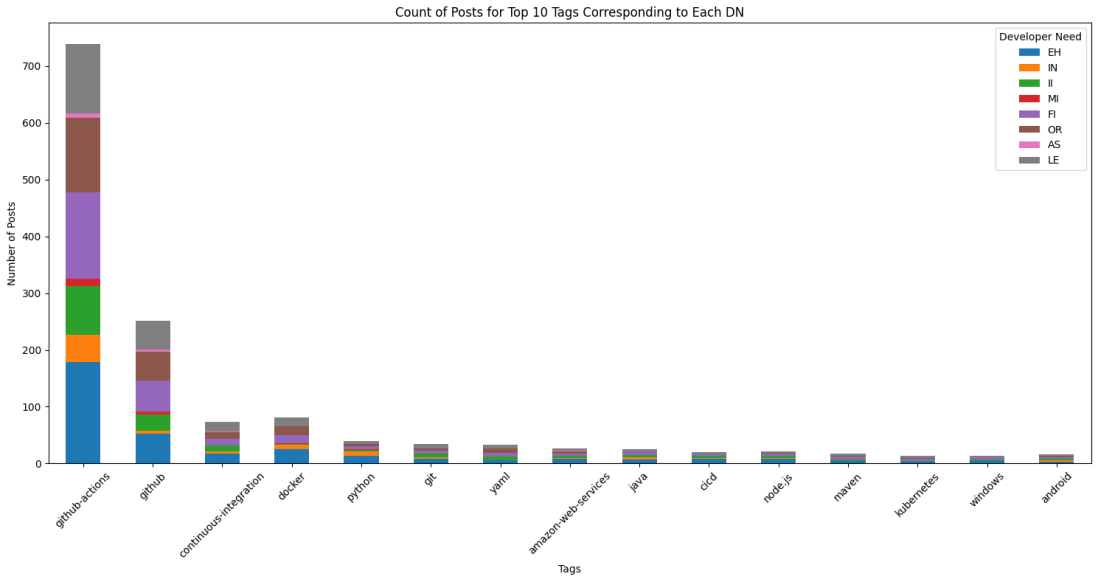

Fig. 7: Count of Posts for Top 10 Tags Corresponding to Each Developer Need (DN). The stacked bar plot shows the distribution of posts associated with each Developer Need (EH, IN, II, MI, FI, OR, AS, LE) across the top 10 most common tags.

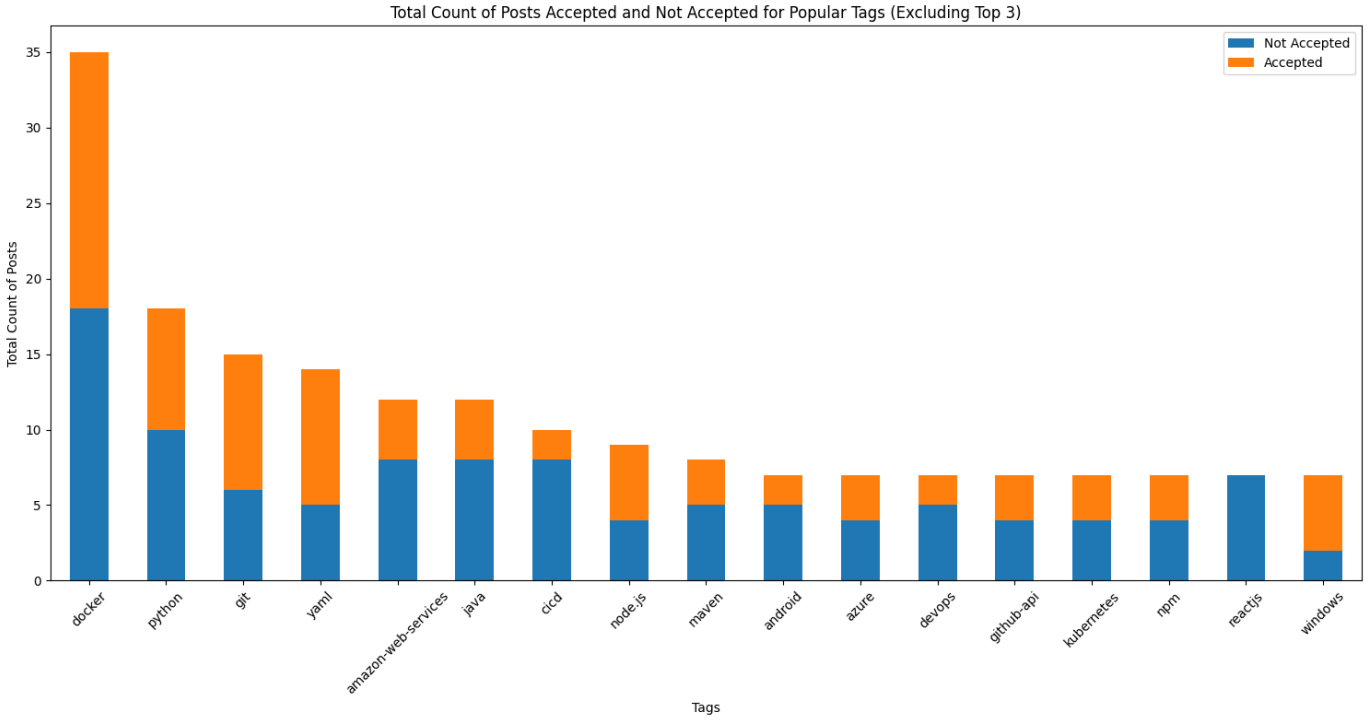

Fig. 8: Total Count of Posts Accepted and Not Accepted for Popular Tags (Excluding Top 3). This stacked bar plot shows the distribution of accepted and not accepted posts for the top popular tags, excluding the three most popular tags.

TABLE VI: Most Common Tags for Each Developer Need (DN)

DN	Tag	Count
EH	github-actions	182.0
	github	53.0
	docker	25.0
	continuous-integration	18.0
	python	13.0
IN	github-actions	48.0
	python	9.0
	docker	8.0
	github	6.0
	continuous-integration	4.0
II	github-actions	86.0
	github	29.0
	continuous-integration	11.0
	git	6.0
	yaml	6.0
MI	github-actions	12.0
	github	4.0
	docker	1.0
	python	1.0
	git	1.0
FI	github-actions	153.0
	github	56.0
	docker	14.0
	continuous-integration	13.0
	yaml	7.0
OR	github-actions	133.0
	github	51.0
	docker	15.0
	continuous-integration	11.0
	yaml	7.0
AS	github-actions	8.0
	github	4.0
	continuous-integration	1.0
	docker	1.0
	amazon-web-services	1.0
LE	github-actions	123.0
	github	51.0
	continuous-integration	17.0
	docker	16.0
	git	8.0