An Exploratory Study on How Software Reuse is Discussed in Stack Overflow

Eman Abdullah AlOmar[1] (✉), Diego Barinas[1], Jiaqian Liu[2], Mohamed Wiem Mkaouer[1], Ali Ouni[3], and Christian Newman[1]

[1] Rochester Institute of Technology, Rochester, NY, USA
{eaa6167,dh7445,mwmvse,cdnvse}@rit.edu
[2] University at Buffalo, Buffalo, NY, USA
jliu275@buffalo.edu
[3] ETS Montreal, University of Quebec, Montreal, QC, Canada
ali.ouni@etsmtl.ca

Abstract. Software reuse is an important and crucial quality attribute in modern software engineering, where almost all software projects, open source or commercial, no matter small or ultra-large, source code reuse in one way or another. Although software reuse has experienced an increased adoption throughout the years with the exponentially growing number of available third-party libraries, frameworks and APIs, little knowledge exists to investigate what aspects of code reuse developers discuss. In this study, we look into bridging this gap by examining Stack Overflow to understand the challenges developers encounter when trying to reuse code. Using the Stack Overflow tags "code-reuse" and "reusability", we extracted and analyzed 1,409 posts, composed of questions and answers. Our findings indicate that despite being popular, reuse questions take relatively longer than typical other questions to receive an accepted answer. From these posts, we identified 9 categories that group the different ways developers discuss software reuse. We found Java and ASP.NET MVC to be the most discussed programming language and framework, respectively. Based on the programming languages and frameworks mentioned in the posts, we noted that Web software development is the most frequently targeted environment. This study can be utilized to further analyze aspects about software reuse and develop guidelines to be practiced in industry and taught when forming new developers.

Keywords: Reusability · Software Reuse · Stack Overflow.

1 Introduction

When dealing with unexpected coding or design problems, developers typically refer to online forums and ask questions. One of the most used sites where developers converge to ask and answer questions of a multitude of topics related to programming, software engineering and related areas is Stack Overflow[4] (SO).

[4] https://www.stackoverflow.com

Since its conception in 2008, SO has seen a yearly increase of its user base with over 10 million users as of 2019. Because of the growing importance of *SO* as a community of developers, a wide number of studies have been performed to understand the ways developers discuss different topics, and about aspects of *SO* and the community themselves.

In this study, we utilize *SO* to understand what are the challenges that drive developers to ask and discuss software reuse. We extract what aspects of this topic are discussed and what technologies are involved in these discussions. To do this, we mined from *SO*, all the questions that are tagged as *software reuse*. We found nine high-level categories that encompass the type of discussions that take place in *SO*, namely *Development, Third-party Software, Design Principles, Refactoring, Legacy System Wrapping, Application Product Lines, Program Generators, Configurable Vertical Applications*, and *General*. We also found that software reuse questions take longer to receive an accepted answer when compared to Web questions Among the discussed technologies, we identified Java as the most used programming language, ASP.NET MVC as the most discussed framework, and Web as the target environment for most of the questions. We provide the software reuse community with a set of well structured and classified questions and answers, for the purpose of analysis, replication and extension[5].

2 Study Methodology

The main goal of this work is to obtain and share insights to the Software Engineering community regarding how software reuse is discussed in practice by analyzing Stack Overflow (SO) posts. Figure 1 depicts an overview of our methodology.

2.1 Data Extraction

To conduct our study, we plan on extracting our data from Stack Overflow. The data should capture all the necessary information needed for our study, starting with questions, answers, discussions, and all their corresponding metadata (publication date, authors, TAGs, source code attached, etc.). We start the mining process through the Data Explorer web interface[6], which releases a regularly updated data dump of the SO network's content. We considered all questions, since SO creation, from August 2008, all the way to December 2019.

To distinguish reuse related questions, we started with using the tags "*reuse*" and "*reusability*" as they are intuitive. We have avoided using any extra tags mainly because it has been shown that developers misuse tags as an attempt to increase views, and the large number of tags tend to cause confusion among posters [4]. Additionally, in order to mitigate tag misuse issues, the relevancy of the posts and tags were verified during the manual analysis described in Section 2.3, so that we only keep posts related to software reuse.

[5] `https://smilevo.github.io/self-affirmed-refactoring/`
[6] https://data.stackexchange.com/stackoverflow/queries

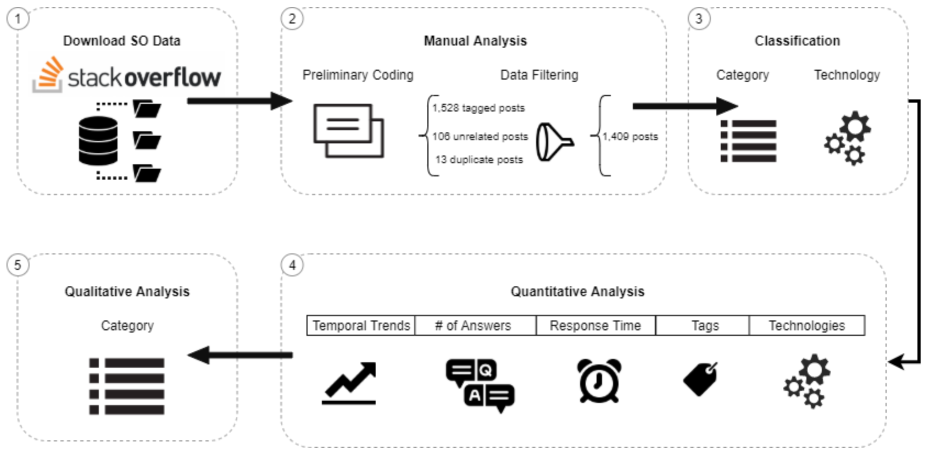


Fig. 1: **Overview of the our study methodology.**

2.2 Manual Analysis

Posts Filtering. We included all posts that referred to reuse of code, and reuse of different resources that are relevant to any of the stages of the Software Development Life-Cycle. In addition, we also included questions that were related to general aspects of software reusability. We excluded all questions that were not related to the aforementioned aspects. Two authors have performed the manual verification of the questions. This process consisted of manually reading the title and question description for each post to assess its relevancy. When the description is ambiguous or short, we refer to any additional comments made to the question, and then we refer to the proposed answers, which may provide a clearer description of the problem. For instance, we found multiple posts where the question was vague, but the answers were clearer based on assuming what the question poster meant, and these assumptions were validated by the poster through either choosing one as a correct answer or by adding comments to further clarify their question.
Duplicate Posts Removal. During the manual validation, we found one duplicate post, 10 posts that were removed (upon looking for them in Stack Overflow), and 2 posts when accessing them redirected to unrelated questions.

To summarize, our data extraction process initially mined 1,528 reuse tagged posts, from which 106 were unrelated to reuse, and 13 were duplicate. After the filtering process, we retained 1,409 posts, created between August 2008 and December 2019.

2.3 Manual Classification

The classification process consists of reading *title, questions, answers* and *comments* to classify them based on various aspects, including the technology used, programming language, and category of reuse. The reuse categories were extracted from the systematic literature review, performed by Ahmaro et al.[1], on papers published from 1977 to 2013 on the different concepts surrounding Software Reusability. In this work, Ahmaro et al.[1] identified the following categories: *design patterns, component-based development, application frameworks, legacy system wrapping, service-oriented systems, application product lines, commercial off-the-shelf integration, program libraries, program generators, aspect-oriented software development and configurable vertical applications.* This set of categories was also supported by Younoussi et al. [5] whose work consisted of a another systematic literature review on *All about Software Reusability.* During the classification process, we combined questions related to *components*, *services*, and *aspects* into *Development*, we also combined *program libraries*, *application frameworks*, and *commercial off-the-shelf integration* into *Third Party Systems.* We also noticed the existence of questions that do not fit into the existing categories, so we have added two categories, namely *Refactoring*, which can be similar to design patterns, except that developers discuss refactoring opportunities without explicitly mentioning any patterns. The second added category was *General*, where questions were open-ended and their answers can belong to various categories. Table 1, enumerates each category, its related categories, its description, and an example from the classified questions.

Our categorization is not mutually exclusive, meaning that one post can be associated with more than one category at the same time. We also classified the posts regarding the technologies involved in the question. We looked into what programming language, framework and operating systems. Although most of the times these technologies were identified by the posters using the Stack Overflow provided tags, we still focus on the content of the question as the source for this classification due to previously identified issues involving tags usage. When looking at the content of the question, we not only referred to explicit mentions of the technologies, but also at implicit ones. Questions that we were not able to associate to specific technologies, were assigned with N/A (Not Available) but still considered for the categories classification.

3 Results and Discussion

RQ_1: How challenging are reuse questions to answer?

Approach. Similar to previous studies that analyze *SO* [2,4], we take as indicators to measure the difficulty of the questions: 1) the percentage of questions without answers, 2) the percentage of questions without accepted answers, and 3) the average response time taken to receive an accepted answer. We present the average response times using both the mean and the median, which reduces the impact of outliers, as we noticed that some of the questions took more than five years to receive an accepted answer. Using these three indicators, we use Web Development (Web) questions as a baseline to compare our SO questions with, because Web questions are found to be the most popular questions in SO [4]. We randomly selected 1,409 posts from web development questions, in the

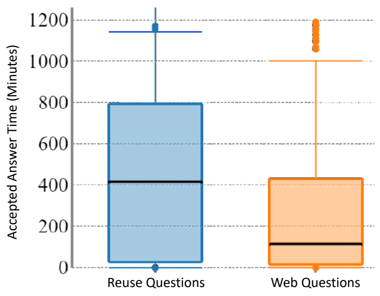
Table 1: **Description of categories for software reuse.**

Category	Description
Development	**Related Categories.** *Aspect-oriented software development, Component-based development, Service-oriented systems* **Description.** Questions related to all types of programming, namely component-based, service or aspect-oriented. Questions in this categories are typically related to programming paradigms. **Example.** In C++, there isn't a de-facto standard logging tool [...] This creates a bit of a problem, however, when trying to create reusable software components. If everything in your system depends on the logging component, this makes the software less reusable [...] (QuestionID 39304).
Third-party Software	**Related Categories:** *Program libraries, Application frameworks, Commercial off-the-shelf integration* **Description.** Question related to creating and consuming APIs and third party libraries, and reusing existing frameworks. **Example.** I am deploying a mod-wsgi application on top of Apache, and have a client program that uses Curl. [...] What do I need to do on the Apache/Mod-wsgi side to enable connection re-use?. (QuestionID: 1056643).
Design Patterns	**Description.** Question related to implementing/gaining knowledge about design principles/patterns. Since the goal of applying design patterns is to achieve quality attributes, this category also includes questions that actively discuss these attributes. **Example.** This class is immutable and I want to make it a singleton for every distinct set of values, using the static factory pattern. [...] I am looking for the best way to cache and reuse previous instances of this class. [...] (QuestionID: 3557620).
Refactoring	**Description.** Questions that actively discuss how to refactor code or the usage of tools to accomplish refactoring. **Example.** I was wondering if there is a tool that will take a bunch of source code and find similar blocks of code so that they could be identified for possible refactoring. [...] (QuestionID: 1104543).
Legacy System Wrapping	**Description.** Questions related to using the features of an already developed system to avoid rewriting the functionality again or creating such type of systems. In a nutshell, reusing a system that does not provide APIs. **Example.** I would like to ask, has anyone successfully reused Audacity's source code in their own program? [...] I am thinking of a few concepts: 1. Compile the whole Audacity as a dll and call it in my C# program (is it possible?) 2. Extract out the functions of playing, recording and displaying waveform and compile them into dll [...] (QuestionID: 5360249).
Application Product Lines	**Description.** Questions discussing the continuous integration of various products, following a given line. Questions are typically around version controlling and repositories organization for better software reuse. **Example.** [...] I have a body of Java code, it used to be for one project. Then a second project, code was based on the first project. So a copy of the first one with many changes.[...] Now there is a third project, something like the first and the second one mixed together. How do I structure the code to have one code base/ git repo that contains all three projects code? [...] (QuestionID: 17202211)
Program Generators	**Description.** Questions regarding the use of code generators to avoid writing the code manually. These programs usually generate code following commonly known algorithms and patterns. Few instances of these questions were found mainly because of the existence of a "code-generation" tag on Stack Overflow which was out of scope for our research. **Example.** When building ASP.NET projects there is a certain amount of boilerplate, or plumbing that needs to be done, which is often identical across projects. [...] I would like to know what approach you find best for 'reusing' this plumbing across projects? [...] Code generation tools, such as T4? [...] (QuestionID: 2551849)
Configurable Vertical Applications	**Description.** Questions that discuss the creation of systems that can be configured to meet the requirements of different users. In other words, one system that can be adapted to different business needs, this way avoiding the creation of multiple similar systems sharing almost identical code. **Example.** I developed a rails app for a school alumni site. Now another school wants me to develop a similar site for them. I want to reuse the app. Data structure will be same but the actual data will be different. Design layout will be similar but design itself will be different. [...] Do you have any experience with such a case? If so, can you share it with me? [...] (QuestionID: 1329824)
General	**Description.** Questions that discuss aspects of software reuse from a high-level point of view and cannot be placed individually in any of the other categories. **Example.** [...] How do you make code reusable? What are the requirements for code being reusable? What are the things that reusable code should definitely have and what things are optional? (QuestionID: 268258)

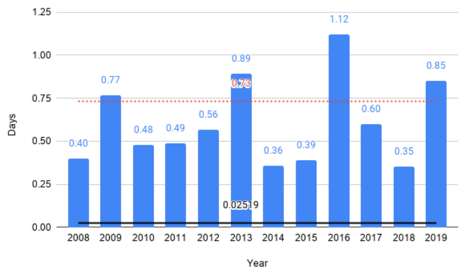
same period that we selected our reuse questions in, and using the same other tags that were also found in our reuse questions. The purpose of choosing the baseline questions, in the same period, and using similar tags, is to design a comparison sample that is as close as possible to our reuse sample, in terms of language and framework, to reduce any related bias.

Table 2: **Percentages of questions without answers, and questions without accepted answers.**

	Reuse Questions	Web Questions
Without Answers	16.21%	14.90%
Without Accepted Answer	58.97%	46.61%



(a) **Average median time.**



(b) **Average response time.**

Fig. 2: (a) Average median time to receive an accepted answer for Reuse questions vs Web questions. (b) Average response time for accepted answers.

Results. Table 2 shows the percentage of questions without answers, and without accepted answers, for both Reuse and Web questions. Results indicate that most of the software reuse questions (83.79%) received at least one answer, reflecting that the Reuse community is highly active, and close enough to the highest active community, *i.e.,* Web, whose percentage of unanswered questions is 85.10%. Furthermore, Asaduzzaman et al. [3] analyzed around 1.3 million *SO* questions and found that 7% of the total corpus of questions was unanswered. This shows that the precentage of unanswered Reuse questions (*i.e.,* 16.21 %) is high.

Although most of the questions received an answer, 58.97% of the questions were not marked with an accepted answer. It is important to note that the percentage Reuse questions with no accepted answer is significantly higher than the percentage of Web questions with no accepted answer (46.61%). This hypothesizes that Reuse questions are relatively harder to answer, in comparison with the Web questions. Such hypothesis itself is insufficient to affirm that Reuse questions are more challenging to answer, because *SO* users are not always available to mark a good answer as accepted. Therefore, to further investigate this finding, we refer to the third indicator, namely the average response time taken to receive an accepted answer.

Figure 2a presents the distribution of time to receive an accepted answer for Reuse questions for Web questions. We observe that the time needed to mark an answer as accepted, for Web questions is typically below 400 minutes, in contrast with Reuse questions, which have a much wider spread that goes up to 800 minutes. Moreover, the median time for a Reuse question to mark an answer as accepted, is 410 minutes, while the median accepted answer's time for a Web question is 230 minutes. To verify the statistical significance of the difference, we use the Mann-Whitney U test, a non-parametric test that checks continuous or ordinal data for a significant difference between two independent groups. The Mann-Whitney test accepted the alternate hypothesis, and the difference between the two groups is found to be statistically significant (*i.e.,* $p < 0.05$). This shows that Reuse questions require more time for users to find the appropriate answer.

Figure 2b shows the average response days for the accepted answers. From the graph, we can see that the year where the responses took longer to be posted was 2016 with an average of 1.12 days, this was followed by 2013 with 0.89 days. Given the time when the data was collected (December 2019), 2019 might have the most unanswered questions and thus this average might be affected. The red dotted line represents the mean response time for the accepted answers of all years having a value of 0.73 days (17.52 hours).

RQ_2: **What are the most popular technologies discussed?**

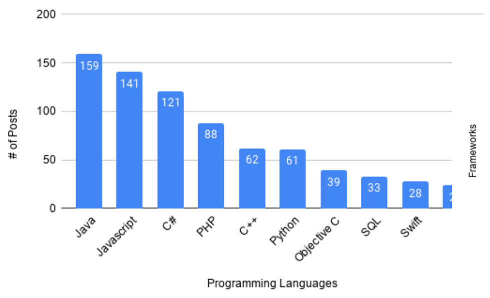
Approach. When performing manual classification, we noted that a lot of the times posters did not tag their questions with all the used technologies or they tagged questions with technologies that had nothing to do with the content of the question. As the tags did not really represent accurate technologies, we manually classified the technologies used in terms of programming language, framework and environment. We also note that because of the nature of software development, one post could have been tagged with multiple target programming languages, frameworks or environments.

Results. Figure 3a shows the top 10 programming languages used when discussing software reuse in *SO*. We can see that Java was the most used one with 159 posts, followed by JavaScript and C# with 141 and 121 posts respectively. This findings suggest a similar list to the ones we can find on different programming languages ranking indexes such as Tiobe[7] and PYPL[8] where the leading positions tend to be occupied by Java, Python and C in recent years. Although our results do not show C as one of the highest used languages, it is understandable as it is not an object-oriented language where software reuse is often encouraged and one of the goals of using such type of languages. In our list, SQL stands with 33 associated posts this was formed by grouping together all

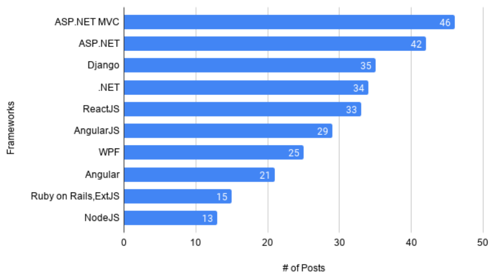
[7] https://www.tiobe.com/tiobe-index/

[8] http://pypl.github.io/PYPL.html

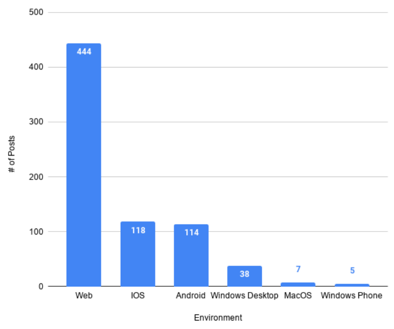
found SQL dialects such as: MySQL, PLSQL, PostgreSQL and TSQL. We would also like to reflect on the number of questions that were not tagged with any programming language, this category accounted for 475 posts (around 34% of classified data). However, not all of these posts were really left without a programming language as, at times, we were simply not able to identify the used language. The list of programming languages with the highest amount of posts with no accepted answers highly resembles the list at the top 10 used programming languages. The list indicates Java and JavaScript with 67 and 61 posts respectively, as the most challenging languages, although this might be highly influenced by the number of posts related to these languages.



(a) **Top 10 most used programming languages.**



(b) **Top 10 most used frameworks.**



(c) **Most targeted environments.**

Fig. 3: Most popular technologies discussed alongside software reuse.

Figure 3b presents the most used frameworks when discussing software reuse in *SO*. As can be seen on the graph, the .NET family of framework stands as the most mentioned ones with ASP.NET MVC at the top of the list with 46 associated posts, followed by ASP.NET with 42 posts. Django, with 35 posts, is third on the list being the most used framework for the Python programming language. The JavaScript programming language stands as the language with the most related frameworks on this list, having ReactJS, AngularJS, NodeJS and ExtJS, which when added together would be placed 2nd only behind the .NET family.

Figure 3c displays the most targeted environments. It shows a great difference between the first and second place, having Web with 442 posts followed by IOS with 118. Although when looking at the data on a higher level (Web, Mobile and Desktop), the difference between the first and second place would be reduced. When combined, the Mobile environment stands with a total of 243 posts. The desktop environments are distributed between the Windows platform and the MacOS being the target of 38 and 7 posts respectively. This result indicates how relevant the development of mobile application has been in recent years, with an overwhelming difference when compared to the desktop environments. Moreover, the posts that target Web environments, due to their nature, might have had an underlying intention of being utilized exclusively in mobile devices which would increase their count.

RQ_3: **What is the trend of software reuse questions since the creation of Stack Overflow?**

Approach. In order to obtain the temporal trends of the overall number of posts for each year, we queried our obtained dataset and plotted the results in a graph. To assess the evolution of the technologies, we considered the top three most used programming languages according to our previous findings. For the frameworks, we grouped together the different families of frameworks belonging to the programming languages associated with the most used frameworks, resulting in .NET family, JavaScript Frameworks and Python Frameworks. Similarly, for the environments, we created the following groupings: Web, Mobile and Desktop.

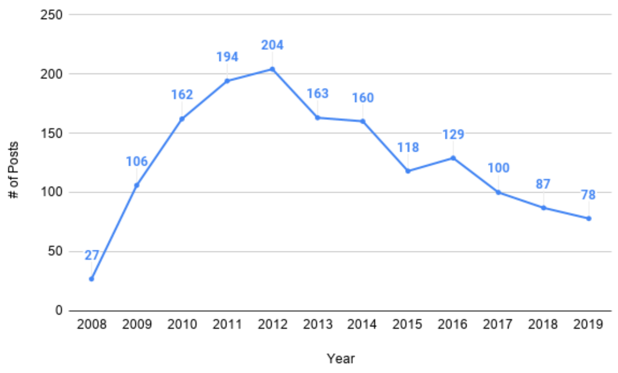


Fig. 4: **Temporal trend of software reuse posts.**

Table 3: **Temporal trend of the most used technologies**

Technologies	Total	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019
Java	159	0	5	14	24	25	17	18	16	8	14	14	4
Javascript	141	0	3	8	11	11	20	13	13	19	20	7	16
C#	121	0	10	19	13	18	15	18	9	8	5	3	3
Total	421	0	18	41	48	54	52	49	38	35	39	24	23
ASP.NET Family	95	4	13	12	21	12	9	3	4	7	6	4	0
Python Frameworks	43	0	2	13	5	4	3	6	3	1	3	2	1
JS Frameworks	125	0	1	3	6	11	17	12	10	17	17	12	19
Total	263	4	16	28	32	27	29	21	17	25	26	18	20
Web	444	5	30	49	62	46	52	38	27	39	41	32	23
Mobile	243	0	4	22	42	40	34	32	27	14	11	11	6
Desktop	56	2	9	12	4	6	6	2	4	6	5	0	0
Total	743	7	43	83	108	92	92	72	58	59	57	43	29

Results. Figure 4 shows the temporal trends regarding the number of posts for each year. As can be seen, there is an increasing trend from 2008 to its peak in 2012. From there until December 2019 when the data was collected there has been a decrease in the number of posts with the exception of 2016 which had more posts than the previous year. This trend suggests that due to the wide adoption of code reuse and how it is more and more taught as a fundamental programming practice, developers don't need much help to achieve software reuse as compared with other aspects of software engineering. It is hard to determine the reason for the peak in 2012, because as our *complete dataset* shows, almost all technologies presented an increase in their numbers for this year. However, when looking at the original tags that the posters tagged their questions with, IOS showed an increase from 7 posts in 2011 to 25 in 2012. Furthermore, when querying the complete *SO* data dumps to search for the temporal trends of the usage of the *"ios"* tag, we can see that the number of posts from 2011 to 2012 almost doubled (from 41,139 to 80,276), an increase that was not seen for any other two consecutive years. This might be in relation to new features released by Apple regarding their development kit in the early months of 2012 or during 2011 (as some technologies might take some time to be fully adopted by the developers).

As seen on Table 3, almost all of the technologies had the less amount of occurrences in 2008, this is clearly due to 2008 being the year with the less amount of posts (27). The JavaScript programming language, as well as the JS Frameworks and the Web environment (where other two technologies are almost always used), presented the highest number of posts for 2019 showing how relevant these technologies are nowadays.

RQ_4: **What categories emerge when discussing software reuse?**

Approach. We showcase the results of our manual classification, performed in Section 2.3.

Results. Figure 5 shows the number of posts associated with each category. As can be seen, Development is the most discussed category in our dataset with 874 posts. This suggests that achieving software reuse is relevant within a single

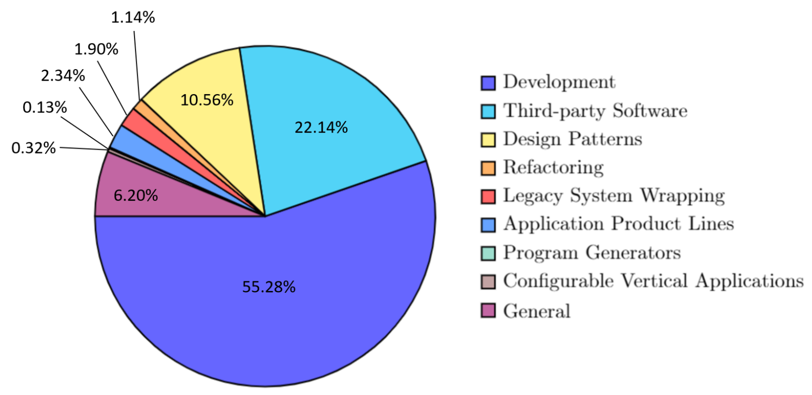


Fig. 5: **Number of posts per category.**

project or application. When analyzing the posts, we noted that a lot of times, in order to better receive answers from the community, posters extract away most of the aspects of their projects, thus leaving a precise and concise problem that falls under this category but that if completely described might have completely fallen under a different one.

Reusability is one of the quality attributes that speeds up the development process, because of this, it is not surprising that the second category with the most posts (350) is about reusing between projects or systems, as one of the goals in the industry is always to achieve fast releases of their products. Moreover, we expected the Third-party Software category to be one of the most discussed as it includes posts regarding the usage of program libraries and APIs which are widely used nowadays in software development.

Design Principles or Patterns are created to provide general solutions to common problems, which is why we expected this category to be among the most discussed ones, it consisted of 167 posts. We noted that a lot of the questions under this category were pertaining to the Inheritance design principle which, in simple terms, focuses on grouping coming functionality in a super class, resulting in avoiding to have the same pieces of code in multiple child classes. For the classification process, we did not take into consideration the DRY (”Don’t Repeat Yourself”) principle as basically all posts belonging to software reuse are about achieving this principle.

Although Refactoring is widely implemented to reduce code duplication (some IDEs even provide automatic ways of achieving this) we did not find many posts (18) that actively discussed this topic. When looking at the tags used by the posters, we found 28 occurrences of the refactoring one, but only 9 of those posts were classified as belonging to this category.

The Legacy System Wrapping and Application Product Lines category had a moderate amount of posts associated with them with 30 and 37 respectively. Because of how specific the Configurable Vertical Applications category can be, there were not many posts (only 5) belonging to this category. The category with the least amount of posts (2) was the one regarding the creation of code through code generation tools.

The General category contained 98 posts. It was interesting to see that many posts in this category, although we can not say that we were not expecting them to exist. Some of the questions in this category were related to people asking about how to achieve software reuse for personal knowledge and interestingly enough, others were asking about how to implement code reuse practices or achieving reusability in a company wide scheme.

4 Conclusion

In this paper, we presented insights regarding how developers discuss software reuse by analyzing Stack Overflow. These findings can be used to guide future research and to assess the relevancy of software reuse nowadays. Our findings show that software reuse is a decreasing trend in Stack Overflow which might indicate that developers have widely adopted this practice and thus few questions regarding it emerge as it is well grasped by the community. On the other hand, they might indicate that, since software reuse is so deeply embedded in achieving good quality software and is often taught as an essential quality attribute, that it has spread to a wide variety of topics and thus the questions are not posted solely as reusability or code reuse issues. Our study opens the door for software reuse researchers to further understand the software reuse challenges.

Acknowledgements This material is based on work supported by the National Science Foundation under Grant No. 1757680.

References

1. Ahmaro, I., Abualkishik, A., Yusof, M.: Taxonomy, definition, approaches, benefits, reusability levels, factors and adaption of software reusability: A review of the research literature. Journal of Applied Sciences **14** (11 2014)
2. Alshangiti, M., Sapkota, H., Murukannaiah, P., Liu, X., Yu, Q.: Why is developing machine learning applications challenging? a study on stack overflow posts. In: International Symposium on Empirical Software Engineering and Measurement. pp. 1–11 (09 2019)
3. Asaduzzaman, M., Mashiyat, A.S., Roy, C.K., Schneider, K.A.: Answering questions about unanswered questions of stack overflow. In: 10th Working Conference on Mining Software Repositories. p. 97–100 (2013)
4. Barua, A., Thomas, S.W., Hassan, A.E.: What are developers talking about? an analysis of topics and trends in stack overflow. Empirical Software Engineering **19**(3), 619–654 (2014)
5. Younoussi, S., Roudies, O.: All about software reusability: A systematic literature review. Journal of Theoretical and Applied Information Technology **76**, 64–75 (01 2015)