e-Informatica Software Engineering Journal, Volume 13, Issue 1, 2019, pages: 105–139, DOI 10.5277/e-Inf190104

A Three Dimensional Empirical Study of Logging Questions from Six Popular Q & A Websites

Harshit Gujral*, Abhinav Sharma*, Sangeeta Lal*, Lov Kumar**
*Jaypee Institute of Information Technology, Noida, Uttar-Pradesh, India
**BITS-pilani Hydrabad Campus, Hydrabad, India
harshitgujral12@gmail.com, sharma1997abhinav@gmail.com, sangeeta@jiit.ac.in, lovkumar505@gmail.com

Abstract

Background: Q & A websites such as StackOverflow, Serverfault, provide an open platform for users to ask questions and to get help from experts present worldwide. These websites not only help users by answering their questions but also act as a knowledge base. These data present on these websites can be mined to extract valuable information that can benefit the software practitioners. Software engineering research community has already understood the potential benefits of mining data from Q & A websites and several research studies have already been conducted in this area.
Aim: The aim of the study presented in this paper is to perform an empirical analysis of logging questions from six popular Q & A websites.
Method: We perform statistical, programming language and content analysis of logging questions. Our analysis helped us to gain insight about the logging discussion happening in six different domains of the StackExchange websites.
Results: Our analysis provides insight about the logging issues of software practitioners: logging questions are pervasive in all the Q & A websites, the mean time to get accepted answer for logging questions on SU and SF websites are much higher as compared to other websites, a large number of logging question invite a great amount of discussion in the SoftwareEngineering Q & A website, most of the logging issues occur in C++ and Java, the trend for number of logging questions is increasing for Java, Python, and JavaScript, whereas, it is decreasing or constant for C, C++, C#, for the ServerFault and Superuser website 'C' is the dominant programming language.

Keywords: classification, debugging, ensemble, logging, machine learning, source code analysis, tracing

1. Introduction

Logging is an important programming practice that is performed by inserting log statements in the source code. These log statements are used to record important runtime information about the program execution. Software developers can use this runtime information at the time of debugging. In addition to debugging, logging is important in several other software development activities such as anomaly detection [1], performance problem diagnosis [2]. For example, Fu et al. [1] use log messages timings to differentiate normal and anomalous executions. Nagaraj et al. [2] purpose a system that compares the state of normal execution sequence (normal performance) and bad execution sequences (bad performance) and identify the states that are different between the two execution sequences. Logging is an important activity for software development, however software developers often face challenges In logging due to changing nature of source code as well as logging libraries. For example, software developers face difficulty in identifying code constructs that needs to be logged [3, 4], log level that needs to assigned

Submitted: 4 Aug 2018; Revised: 28 Feb 2019; Accepted: 28 Feb 2019; Available online: 31 May 2019

to log statements [5] or issues in migrating log libraries [6]. Hence, recently several techniques have been proposed by the software engineering research community to help software developers in source code logging [3–5, 7].

The techniques proposed in the literature for helping software developers in logging are useful, but, at present there is little understanding about the major logging concerns of the different software practitioners like software developers, system administrators, database administrators etc. A detailed study of the most frequent logging concerns of the software practitioners can be beneficial in further improving the existing logging techniques or tools. Information present on the technical Q & A websites can be a great resource for identifying the logging concerns of the software practitioners. Table 1 shows 6 logging questions from six popular StackExchange Q & A websites, i.e., StackOverflow (SO) [8], ServerFault (SF) [9], SuperUser (SU) [10], Database Administrators (DB) [11], Android Enthusiast (AE) [12], and Software Engineering (SE) [13]. Each question in Table 1 received thousands of views from the software development community. For example, question 1 received 192,320 views. This indicates the impact and reach of Q & A websites in software development community. In the question 1, the user has asked a questions on SF website which is related to 'Enabling MySQL logging'. It shows that users face issue in enabling MySQL logging. In question 6, the user has asked about 'best practices of logging user actions in production'. In this question, user wants more information about logging practices of user action. We believe that a detailed characterization study of the logging questions asked on these websites can provide a valuable insights about the logging needs of software development community.

The software engineering research community has already recognized the potential of the Q & A websites in various applications [14, 15]. For example, Pinto et al. [14] analyze the SO questions to find application-level energy consumption related issues. Mario et al. [15] analyze SO questions to find mobile development related issues. Barua et al. [16] analyze questions on the SO website to find software development related trends. All these studies analyze important aspects of software development. However, at present there is no research study that analyzes the data from the Q & A website for identifying source code logging issues. In this paper, we take the first step towards analyzing the logging concerns of software developers from popular Q & A websites.

The overall goal of our research is to improve the understanding about the logging issues that software practitioners face the most. In particular, we aim at systematically analyzing the questions from Q & A websites. We hypothesize that Q & A websites represents an important knowledge base and can be beneficial in identifying the source code logging concerns of the software developers. The findings of this paper can be beneficial to software practitioners in many ways. Product manager can use this study to perform market analysis to find logging tools that are gaining popularity. Software practitioners can use this study to find logging tools/libraries that are commonly used by other software practitioners. These findings can be used by the StackExchange team for site moderation/archiving purpose. Additionally, software engineering research community can use the results presented in this paper to further improve the current logging prediction or improvement studies.

In this work, we perform a three dimensional, large scale and an in-depth empirical study of logging questions asked on six popular community based Q & A websites from the StackExchange network. We analyze more than 82K questions from six popular programming Q & A websites with respect to three different research dimensions, i.e., *statistical analysis*, *programming language analysis*, and *content analysis* and answer a total of 7 research questions. The results of our empirical analysis show several interesting insights such as logging questions are pervasive in all the programming websites. It shows that nearly 1.06–11.6% of all the logging questions invite a great amount of discussion. The work presented in this paper is a significant extension of our previously accepted work *Empirical Analysis of the Logging Questions on the StackOverflow Website* at *Conference on Software Engineering & Data Sciences (CoSEDS-2018).*

Table 1. Example of logging related questions from various websites: WN: Website Name

S. No	WN	Question Id	Title	View Count	Tags
1	SF	71071	How to enable MySQL logging?	192,320	MySQL, **logging**
2	SO	56628	How do you clear the SQL Server transaction log?	965,799	sql-server, **transaction-log**
3	SU	176165	Where Linux places the messages of boot?	139,650	Linux, boot, centos, **logging**
4	AE	14430	How can I view and examine the Android log?	355,050	**logging**
5	DB	4043	Can I see Historical Queries run on a SQL Server database?	173,358	sql-server, SQL, **logs**
6	SE	168059	Best practices for logging user actions in production	49,435	C#, asp.net, **logging**

The remainder of the paper is organized as follows. In Section 2, we describe the closely related studies in context to the work presented in this paper and the novel research contribution made by this work. In Section 3, we describe the various research dimensions and their respective research questions, research method that we followed, the experimental dataset, and the results of the empirical study. In Section 4, we give various threats to validity related to the finding of this paper. In Section 5 we conclude the paper and provide details about future directions and finally, in Section 6, we give acknowledgment.

2. Related Work

In this section, we review the closely related work to our research and list down our specific research contributions. We divide related work into multiple lines of research, i.e., 1) Empirical analysis of logging statement, 2) Logging prediction studies, and 3) Empirical analysis of Q & A websites.

2.1. Empirical analysis of logging statement

Logging is a cross-cutting software development concern and has attracted attention of many researchers. Logging statements present in the source code have been analyzed with respect to several dimensions, such as **type of changes in log statements** ([17]), **reasons of migrating from one logging library to other** ([6]), **source code constructs that are logged more frequently as compared to others** ([18,19]), **relationship between code quality and logging statements** [20], **uses of different log levels in source code** ([5]). These analysis provide important information to software developers. For example, Yuan et al. [21] analyze four open source projects written in C\C++. They identify type of changes to logging statements where software developers spend most of their time. Shang et al. [20] analyze modifications done to logging statements for Java projects. They report four major reasons for logging modification, i.e., debugging, feature change, inaccurate logging level, and redundant logging. Chen et al. [17] replicate the study performed by Yuan et al. [21] for Java projects and report several differences in the results. For example, in Java projects deleting and moving log printing code accounts to 26% and 10% to all logging modifications whereas in C\C++ it accounts to only 2%. Kabinna et al. [6] identify reasons for logging library migration on Java software projects. They report two major reasons, i.e., flexibility and performance improvement, for logging library migration. Li et al. [5] analyze log-levels of various open-source Java projects and report several interesting findings. They report that no single log level dominates. They also report that different projects show varying distribution of log levels. In another study, Li et al. [22] analyze four Java software projects and identify 20 reasons for logging change in source code. They categorize these 20 reasons into four categories: *changing context code*, *improving logging*, *dependency-driven changes*, and *fixing logging issues*. Yuan et al. [23] analyze 250 randomly sampled bug reports from five large C\C++ projects and report the most frequently occurring error patterns that need to be logged. Fu et al. [18] work on analyzing logged and non-logged code constructs. They analyze log statements and their logged code snippets from two closed-source systems at Microsoft (written in C#). They categorize the log statements in five categories: *assertion-check, return-value-check, exception, logic-branch* and *observing-point* logging. They further perform a detailed study of 70 non-logged catch-blocks and find reasons of not logging. Lal et al. [19] analyze logged and non-logged catch-blocks. They report several distinguishing characteristics between logged and non-logged catch-blocks. For example, try-blocks associated with logged catch-blocks have much higher complexity (measures using SLOC, number of operators and number of method calls) as compared to that of non-logged catch-blocks.

All of the above studies analyze logging statements present in the source code. In contrast to these studies, in this work, we analyze logging questions from six StackExchange sites to get insights about the logging issues that software developers face most frequently.

2.2. Logging prediction studies

Logging is crucial for software development and hence, in past researchers spent a great amount of effort for providing software developers with tools and techniques that can help them in source code Logging. For example, Fu et al. [18] and Zhu et al. [24] propose a tool *LogAdvsior* to help software developers in logging prediction for exception types and return value check code snippets for C# projects. Lal et at. [3, 4] propose *LogOpt* and *LogOptPlus*, machine learning models for catch-blocks and if-blocks logging prediction for Java projects. Li et al. [5] propose model for log level prediction. Kabinna et al. [25] propose a model for log statement stability prediction for Java projects. Lal et al. [7] propose a method *LogIm* for predicting logging statement for if and catch-blocks for imbalanced dataset. In another study, Lal et al. [26] use ensemble of classifiers for doing cross-project logging prediction.

The work presented in this paper, is complementary to above studies. We work on identifying the most frequent logging issues of the software developers. Hence, the findings of this work can be beneficial in further improving these logging prediction models. For example, researchers can select in which language software practitioners face most of the logging issues and can provide logging tools for the same. Researchers can identify which are the most frequent libraries in which software practitioners facing the issues and hence, can provide solutions to apply logging prediction for these logging libraries.

2.3. Empirical analysis of Q & A websites

The StackExchange is network of popular Q & A sites and is actually a knowledge base, several research studies have already been conducted using the data from StackExchange websites. Pinto et al. present an empirical study on analyzing 300 StackOverflow questions and 550 StackOverflow answers on problems related to application-level energy consumption [14]. They study distinctive characteristics, most common problems, main causes and solutions recommended on software energy consumption [14]. Mario et al. apply topic modelling to discover hot-topics on mobile development by mining questions and answers on StackOverflow [15]. Their findings reveal that most of the questions are on compatibility issues, crash reports and database connection [15]. Beyer et al. conduct a manual categorization of Android app development related issues on StackOverflow [27]. They investigate 450 Android related posts and conclude that developers mainly have issues related to usage of API components such as User Interface and Core Elements [27].

Yang et al. study security related questions on StackOverflow and cluster security related questions (such as cryptography and mobile security) based on their text [28]. They discover that security related questions belong to five main categories, i.e., web security, mobile security, cryptography, software security, and system security [28]. Malik et al. manually analyse 1000 posts on Android energy consumption [29]. Their study reveals that most of the questions are related to improper implementation, sensor and radio utilization [29]. Nagy et al. present a study in mining StackOverflow for discovering error-prone patterns in SQL queries [30]. Their study reveals that the SQL statements of the code blocks can be automatically analyzed to identify error-prone patterns which can be used in a recommendation system [30].

Above studies analyze one aspect of programming or software development and none of these studies focus on analyzing questions related to logging. In contrast to these studies, the work presented in this paper focuses on analyzing source code logging questions on six StackExhange websites.

3. Empirical Study

3.1. Research Dimensions and Research Questions

Table 2 shows three main research dimensions (RDs) and respective research questions (RQs) considered in this work. Following is a brief description of each RD and respective RQs:
RD1: Statistical Analysis of Logging Questions on the StackExchange Sites: In RD1, we explore *how* software development communities use StackExchange websites for asking logging related issues. For this, we analyze several parameters related to logging questions. We analyze the trend of logging question with accepted answers (**RQ1**), number of answers posted for each logging question (**RQ2**), and time taken by each logging question to get the accepted answer (**RQ3**).

Successful questions (question with an accepted answer) depict satisfaction of the programmer. These trends are essential for development of logging tools and libraries. We conducted this analysis on six websites and hence, it broadens the observation of satisfaction of logging users across various platforms. For an instance, significantly more accepted questions were observed in Database Administrator (DB) than Android (AE). This also gives the sense of how alive is logging today and how much more research and development is required in order to satisfy the needs of programmers dealing with source-code logging

Number of answers per question is the sign of the amount of discussion source-code logging is attracting on these six-websites. Additionally, if a question is attracting a large amount of discussion then it may symbolize the presence of some widely occurring error or some ambiguity faced by the programmers. Study of these cases will aid in developers and researchers in developing tools and methods that would be easy to use and debug. For example, *logcat, alogcat* and *adb* questions invite great amount of discussion in Android.

Time taken to get an accepted-answer to a question corresponds to the time taken to solve user's posted issue. Lesser the time, quicker the solution. If some questions are taking large time in getting accepted-answer, it can depict the presence of some esoteric (lesser known) issues that need to be researched in order to present a palatable solution. If a logging tool or library is associated with large time-taken then concerned developers should intervene with a solution or some version update in order to fix such problems. For example, our analysis shows that questions asked on SE website invite a great amount of discussion. On SE website the user is asking fundamental questions like which methods of better for logging *file* or *database*.
RD2: Analysis of Logging Questions with Respect to Different Programming Languages: In RD2, we analyze trend of logging questions with respect to different programming languages. First, we analyze how pervasive software logging questions are on the community based Q & A websites across programming languages (**RQ4**). Second, we analyze, the distribution of logging questions with respect to different programming language for each Q & A website (**RQ5**).

Analysis of RQ4 provides insight into the development of source-code logging tools and libraries. The results of this RQ will be helpful to analyze the dependency of the programming languages with source-code logging. The results will help in estimating programmer's interest and discussion with respect to various languages. This will help developers to understand emerging trends in programming languages and they would be able to wisely choose a programming language for building logging tools. For example, our results show that maximum number of logging questions are asked in Java and C++. Companies can use this information to build new logging tools.

The RQ5 which is an extension of RQ4. In this RQ, we analyze programming language distribution across six-websites. This would aid developers to choose programming language based upon various environment and platforms. For Example, Server (SF) and super-user (SU) oriented applications suggest a large interest in C-based logging tools while for software engineering, C#, Java, and C++ seems to be a viable option.

RD3: Content Analysis of Logging Questions on the StackExchange Sites: We analyze the information present in logging questions. We perform two types of analysis in this: First, we identify the main topics present in the title and description of logging questions (**RQ6**). Second, we analyze the tags associated with the logging questions (**RQ7**).

Results of this dimension provide an overview of most discussed logging topics. This insight will help developers to keep in mind these discussions while developing logging tools and libraries. It would also aid to keep a track of logging-related issues and needs of programmers. In RQ6, we focus on analyzing the content of the post. Answer of this RQ, provides important insight like Android users face logging issues in network connections. Research community can use this information to further improve logging functionality of network related functions in Android OS.

The analysis of RQ7, provide information about cross-discipline logging tools and practices, for example, the transaction log is used in both server environment (SF), StackOverflow (SO) and database (DB) while event-logging practice is observed in super-user (SU) and StackOverflow (SO). This knowledge is the use-case for researchers and developers to select logging practices and tools that are compatible with multiple platforms and environments.

3.2. Research method

In this subsection, we describe the research methods followed in this work. There are several Q & A websites such as StackExchange [31], Quora [32], where people can post their questions and other people or experts can reply to their questions. In this work, we select StackExchange websites for our analysis because it is a network of so many popular Q & A sites. At the time of this study, there were a total of 133 websites present in the StackExchange network. The StackExchange network consists of websites related to various domains such as software development, tourism, academia. Our aim in this work is to analyze questions related to logging. Hence, analysis of all of these websites is not required and is out-of-scope of this paper. Thus, we carefully selected six technical Q & A websites from all these websites. Following is the criteria and essential properties that we took into account while selecting websites for our study:

Type – Software development/Uses: In this work, we are analyzing questions related to logging. Hence, we select websites related to software development and programming.

Number of users – At least 1000: We select websites having at least 1000 users in order to draw statistically significant conclusions.

Number of questions – At least 1000: We select websites having at least 1000 questions so that we can draw statistically significant conclusions.

Age of the website – At least 2 years old: We select websites having at least 2 years of history. Website which are not so old or are in there beginning phase may not be appropriate for our study as they may not have enough logging questions to infer any statistically significant conclusion.

3.3. Experimental dataset details

Matching to our selection criteria we select following six popular websites that are frequently used

Table 2. Details of research dimensions and research questions

Research dimension	Research questions
Statistical analysis	1. What is the trend of successful and ordinary or unsuccessful questions on logging across years and across StackExchange sites? 2. What is the trend of logging question in terms of quantity of answers per question across years and across websites? 3. How much time it takes to get the accepted answer of logging questions?
Programming language analysis	4. How pervasive is software logging related questions on community based Q & A websites across programming languages? 5. What is the distribution of logging questions with respect to different programming language for each Q & A website?
Content analysis	6. What are the main discussion logging topics in various websites? 7. What is the distribution of logging-related tags across various Stack Exchange websites? And how persuasive is the commonality between these tags along various Stack Exchange websites?

by software practitioners. All the six websites are actively used by thousands of users.

StackOverflow (SO): SO is a Q & A website created for professional and enthusiast programmers [8]. It is created in the year 2008, i.e., ≈ 10 years old. At the time of this study, it consisted of ≈ 8.2 million users, ≈ 14 million total questions and ≈ 75 K logging questions.

Serverfault (SF): SF is a Q & A website for system and network administrators [9]. It is created in the year 2009, i.e., ≈ 9 years old. At the time of this study, it consisted of ≈ 0.3 million users, ≈ 0.2 million questions, and ≈ 4.2 K logging questions.

Superuser (SU): SU is a Q & A website for computer enthusiasts and power users [10]. It is created in the year 2009, i.e., ≈ 9 years old. At the time of this study, it consisted of ≈ 0.6 million users, ≈ 0.3 million questions, and ≈ 1.2 K logging questions.

Database Administrators (DB): DB is a Q & A website for database professionals who wish to improve their database skills and learn from others in the community [11]. It is created in the year 2009, i.e., ≈ 9 years old. At the time of this study, it consisted of ≈ 0.1 million users, ≈ 60 K questions, and ≈ 1.1 K logging questions.

SoftwareEngineering (SE): SE is a Q & A website for professionals, academics, and students working within the systems development life cycle [13]. It is created in the year 2010, i.e., ≈ 8 years old. At the time of this study, it consisted of ≈ 0.2 million users, ≈ 47 K questions, and 198 logging questions.

Android (AE): AE is a Q & A website for enthusiasts and power users of the Android operating system [12]. It is created in the year 2010, i.e., ≈ 8 years old. At the time of this study, it consisted of ≈ 0.1 million users, ≈ 46 K questions, and 183 logging questions.

3.4. Dataset preparation

In this subsection, we describe the steps that we used to extract the relevant dataset for our study. For this study, we have used the data dump provided by StackoverFlow community. This dataset is in XML format and consists of details of all the questions asked by users. For each website, it provides 7 files: badges.xml, comments.xml, posts.xml, posthistory.xml, postlinks.xml, user.xml, votes.xml. For this study we have used **posts.xml** file. This file consists of information each post. For example, if for a give question there are three answers, then a total of four post will be included in this file. This file consists of information like, title of the questions, description of the questions, date on which the questions asked etc. Next, we extract all the logging questions. Manual identification of all the logging questions can be a tedious task. Hence, we adopt a method to automatically find the logging questions. We use tags

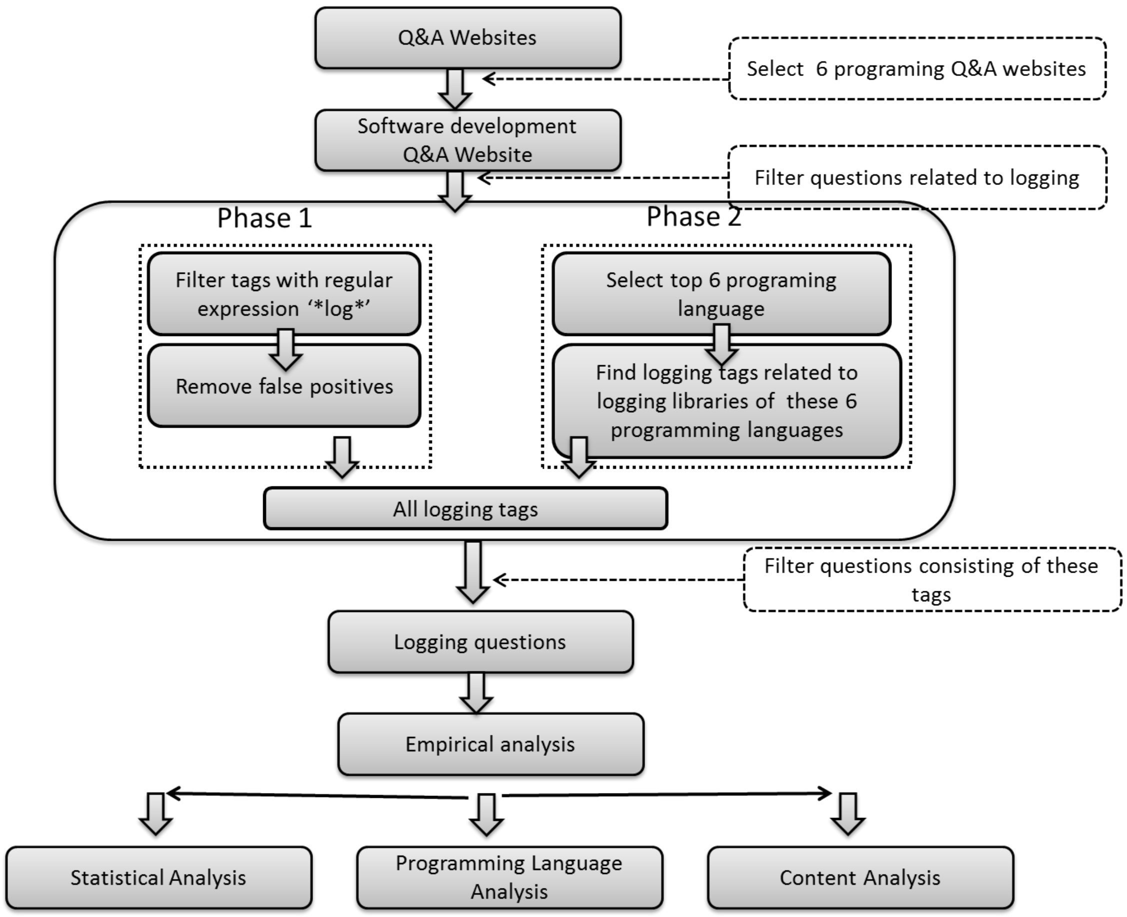

Q&A Websites
Select 6 programing Q&A websites
Software development Q&A Website
Filter questions related to logging
Phase 1
Phase 2
Filter tags with regular expression '*log*'
Remove false positives
Select top 6 programing language
Find logging tags related to logging libraries of these 6 programming languages
All logging tags
Filter questions consisting of these tags
Logging questions
Empirical analysis
Statistical Analysis
Programming Language Analysis
Content Analysis

Figure 1. Research method followed in this study

assigned to questions to identify logging questions. The StackExchange community assigns a set to tag to each question. These tags are chosen carefully to describe the domain of the question. We use a **2-phase** method to select all the logging tags. Figure 1 present the main steps of our **2-phase** method. Below we describe our **2-phase** method:
Phase 1: In phase 1, we define a regular expression, i.e., **log** to retrieve all the logging tags. We notice that this approach is very effective in finding logging tags, as we were able to retrieve several logging tags using this approach. For example, we are able to retrieve tags such as *transaction-log*, *syslog*, *syslog*. However, we notice that this approach results in lots false positives also. For example, tags like *'login'*, or *'logins'* were also outputted. Hence, we manually remove false positives from the dataset.
Phase 2: We noticed that the phase 1, is not able to retrieve logging questions because the regular expression used in the phase 1 is not able to retrieve tags such as *SLF4J*. Hence, we decided to add these kinds of tags manually. We select top 6 programming language from the tiobe index, i.e., Java, C, C++, Python, C#, and JavaScript. We perform an exhaustive Google search to identify all the logging libraries used for these six programming languages. For example, *Log4J* and *SLF4J* for Java, *Log4C* for C, C++. We add tags related to these libraries in our list.

Using process followed in Phase 1 and in Phase 2, we retrieve a total of 169 tags.We extract all the questions consisting of any of these tags. We extracted 82199 logging from these six websites. We made all our dataset publicly available to allow replication of the results by software engineering research community (https://github.com/newtein/StackExLogging). For each website, we compute the percentage that logging questions have with respect to total questions.

Table 3. Experimental dataset details of StackExchange website – StackOverflow: SO, SuperUser: SU, ServerFault: SF, DBA: DB, SoftwareEngineering: SE, Android: AE

Field	SO	SU	SF	DB	SE	AE
Total Number of Unique Users	8287574	630516	346259	114789	241851	154687
Total questions	14995834	363915	252963	60948	47362	46559
Total questions with accepted answer	8034235	154322	125601	29400	27762	13316
Total logging questions	75185	1275	4227	1131	198	183
Total logging questions with accepted answer	39674	541	2163	555	110	50
Percentage of logging questions to total questions	0.19	0.14	0.62	0.78	0.10	0.17
Timestamp of the First Question	8/1/2008	7/15/2009	4/30/2009	10/22/2009	9/27/2010	9/1/2010
Timestamp of the Last Question	12/3/2017	11/30/2017	12/1/2017	12/1/2017	11/19/2017	11/28/2017

Table 3 shows that SF and DB have the highest percentage of logging questions. This table also shows that a large number of questions are asked on logging.

3.5. Research contributions

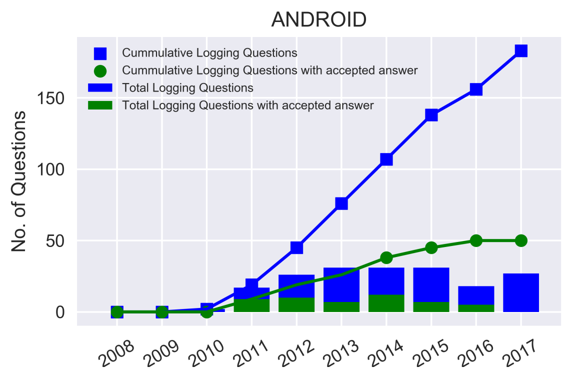
In context to work done in literature, in this work, we perform the first study (to the best of our knowledge) of logging questions on six popular StackExchange websites with respect to three dimensions. We identify several RQ's related to statistical and content analysis of logging questions. We answer each RQ by conducting empirical analysis on more than 82 K logging questions.

3.6. RD1: Statistical analysis

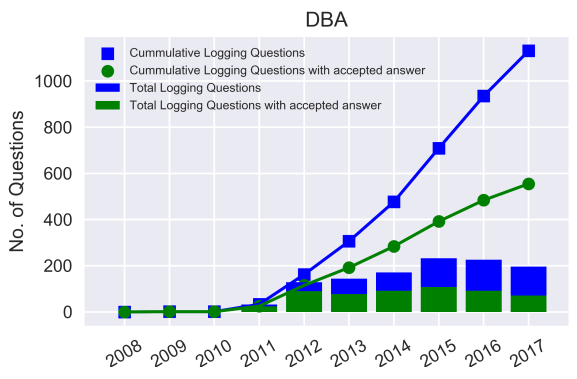
In this subsection, we present the results of various RQ's related to statistical properties of logging questions. This research dimension provides information about how the behavior of programmers is changing over time. Statistical trends are observed from posted questions and answers. This dimension of research may not have a direct application for a user but it is in-fact quite essential to understand how source-code logging is evolving over the years? This gives a sense of how alive is logging today and how much more research and development is required in order to satisfy the needs of programmers dealing with source-code logging.

3.6.1. RQ1: What is the trend of successful and ordinary or unsuccessful questions on logging across years and across StackExchange sites?

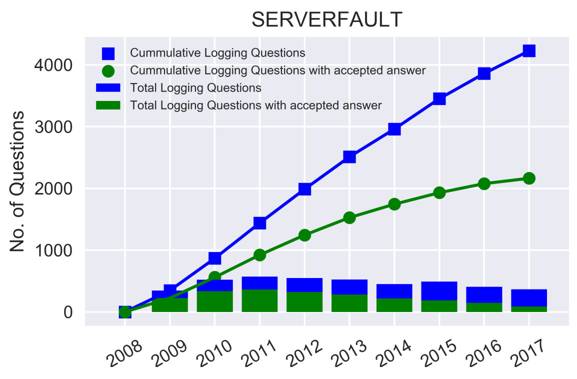
Motivation: On StackExchange sites a question can receive multiple answers. The user who has asked the question can review these answers. If he is satisfied with one of these answers, he can mark that answer as *accepted* [33]. However, if none of these answers, answer the question correctly, the user has the right to not select any of these answers as accepted. Each question can have only one *accepted* answer. In the literature [14], StackExchange questions are classified into three categories: *successful* (questions with accepted answer), *ordinary* (questions that have answers but none of them is accepted) and *unsuccessful* (questions that do not have any answer). In this RQ, we analyze the trend of logging questions and logging questions with accepted answers on various StackExchange websites. Accepted an-



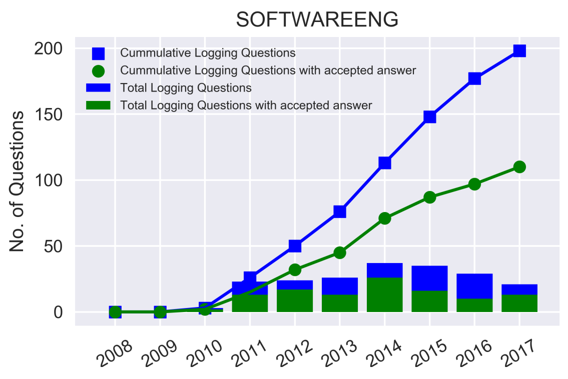
(a) AE



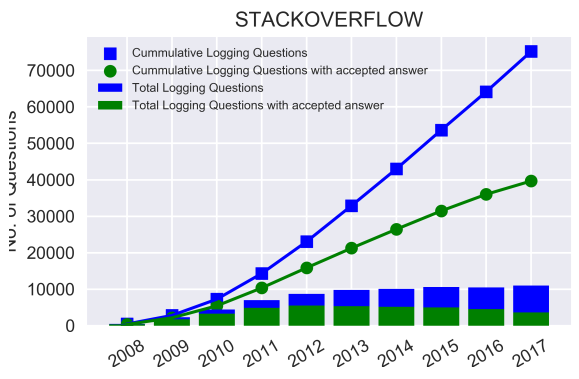
(b) DB



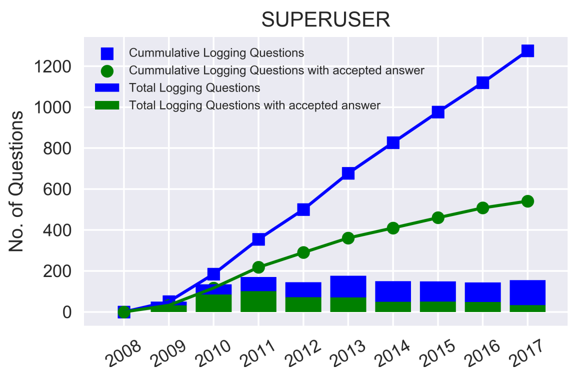
(c) SF



(d) SE



(e) SO



(f) SU

Figure 2. Distribution of logging questions with accepted answers

swers are indicators of quality of responses to questions. Accepted answers shows that questions on logging are receiving helpful responses. We believe that identifying number of logging questions that are successful can be beneficial in identifying the behavior and satisfaction of software development community towards logging.

Approach: In this RQ, we extract total logging questions and logging questions with accepted answers (i.e., successful logging questions) for each of the six websites. We extract this data for all the years from 2008 to 2017. Using this information, we plot histogram showing the total number logging questions and total successful logging questions for each of the six website. We also plot cumulative logging questions and cumulative successful logging questions for all the six websites.

Results: Figure 2 shows the histogram of total logging question and successful logging questions. It also shows the trend of cumulative logging questions and cumulative successful logging questions. From Figure 2, we draw several interesting observations. First, it shows that irrespective of the website logging questions occur consistently across all the websites. For example, on SU website users had asked 50–177 questions in each year between 2009 to 2017. Second, we observe that the frequency and intensity of questions differ across the websites. For example, a total of 75185 logging question are asked on SO whereas 1131 logging questions are asked on DBA in the years 2008–2017. This huge difference in the number of logging question between DBA and SO does not necessarily means that database have less logging issues. It can be also be due to difference in the popularity and user base of the websites. For example, the SO website has much bigger user base and is much more popular than other StackExchange websites, and hence, has much more logging questions as compared to other sites. Third, we observe that there is no trend (increasing or decreasing) in number of logging questions asked over the years for all the websites. Fourth, we observe that all the websites have a large number of successful logging questions. For example, 33–78% of logging questions have received an accepted answer on the SO website.

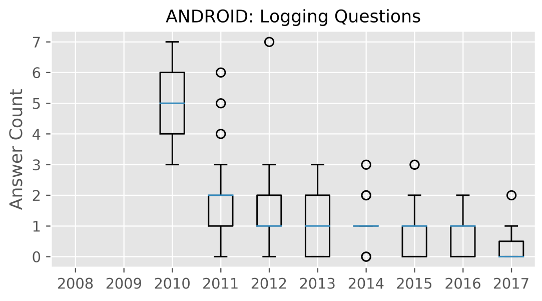
RQ1 conclusions: Logging is an important concern that occurs frequently in different domains. We observe a large number of successful logging questions. However, we do not observe any trend in terms of frequency of total logging questions and successful logging questions across the years for any website.

3.6.2. RQ2: What is the trend logging questions in terms of quantity of answers per question across years and across websites?

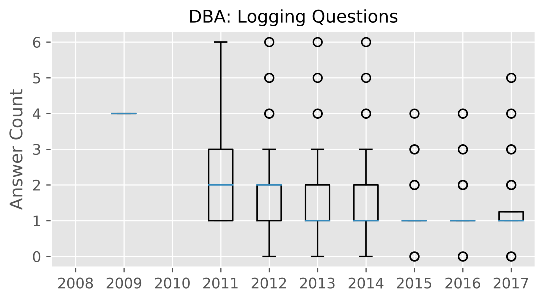
Motivation: In this RQ, we analyze the number of answers posted for each logging questions. On StackExchange sites, users can post any number of answer to each questions. We consider answer count posted for each question as a measure of **discussion**. Increase in number of answers can be an indication towards increase in discussion required for logging questions.

Approach: To answer this RQ, we compute number of answers received for each logging question for each year for all the six websites. Using this information, we compute descriptive statistics such as Quartile-1 (Q1), Median, Quartile-1 (Q3), Min and Max and create box-plots. We compute descriptive statistics to gain insight on the data characteristics and its basic features.

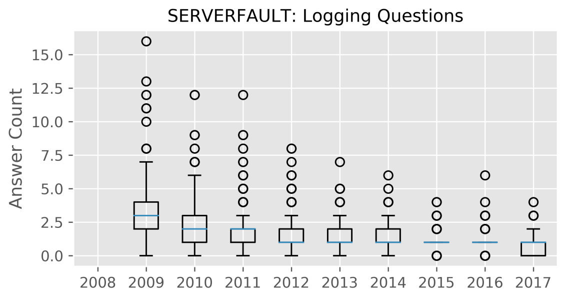
Results: Figure 3 shows box-blot of number of answers received for each logging question for all the websites. We study the central tendency of the data in-terms of the median values. The median values of the answer count for the sites AE, DB, SF, SE, SO and SU in the year 2014 are 1, 1, 1, 2, 1, and 1, respectively. The box-plot in Figure 3 reveals the dispersion in the data which is the spread of the values around the median. We draw several interesting observations from the Figure 3. First, we notice that for all the websites, the median value of the number of answers received for each question is higher in the initial years (2008–2011) as compared to later years (2012–2017). For example, for SE website the median values of answer count is 5 (2010), 3 (2011), 2.5 (2012) and 2 (2013–2017). For this outcome, one reason can be that old questions receive more answers over the period of time.



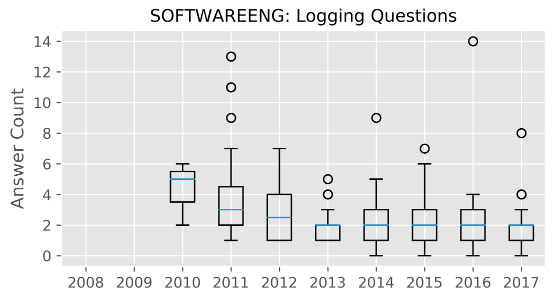
(a) AE



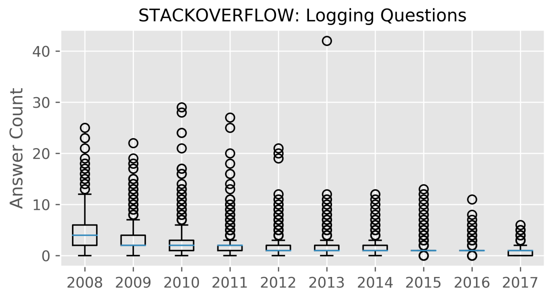
(b) DB



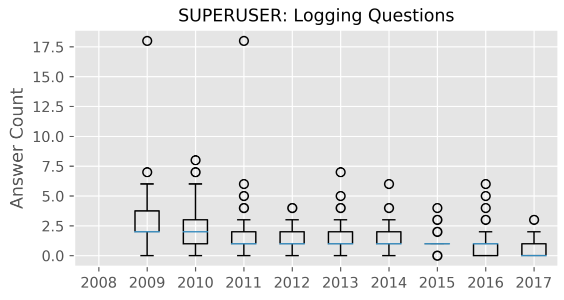
(c) SF



(d) SE



(e) SO



(f) SU

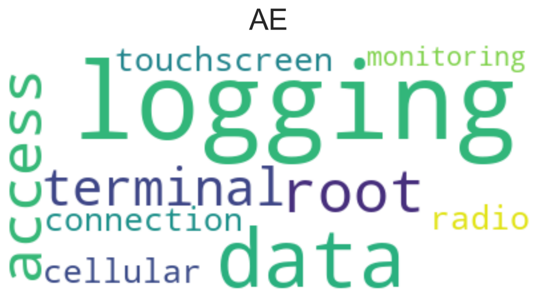
Figure 3. Box plot showing answer count of all the logging questions

Second, we observe presence of several outliers in all the websites. We show the outliers in Figure 3 using dots so that they are clearly visible and displayed separately and do not exaggerate the range values. We observe that the SO website has the highest number of outliers as compared to any other website. To get insight about the questions receiving a large number of answers, we further analyze logging questions that received $\geq$ 5 answers. Table 4 shows the percentage of logging questions that receive $\geq$ 5 answers. Our analysis shows that $\approx$1.06–11.61% of questions in all the websites received $\geq$ 5 answers. It is interesting to find that the SE website has the highest percentage of questions that received $\geq$ 5 answers.

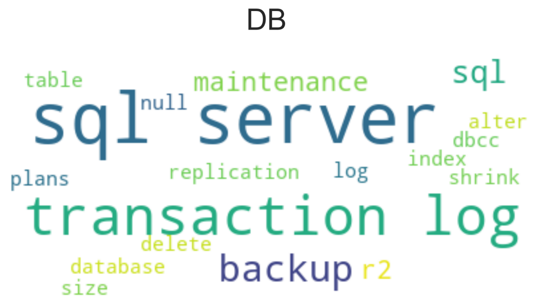
Third, we analyze tags assigned to logging questions that received $\geq$ 5 answers. We build word cloud of tags associated with these questions (refer to Fig. 4). We observe the word cloud of each website highlights a different set of tags. For example, in Android logging questions are related to *logcat*, *alogcat* and *adb*. In DB website, all the 12 questions are related to *transaction-logs*. The SE websites word-cloud highlights tags like *exception, practice*. The logging questions in SE website are related to *logging practices*. The most discussed logging questions on the SO website

Table 4. Percentage of the questions that received ≥ 5 answers – StackOverflow: SO, SuperUser: SU, ServerFault: SF, DBA: DB, SoftwareEngineering: SE, Android: AN

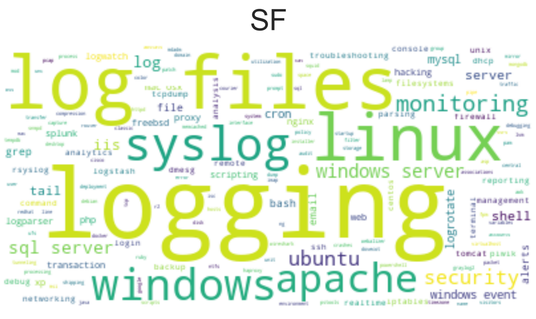
SO	SU	SF	DB	SE	AN
2.8	2.35	3.76	1.06	**11.61**	2.18



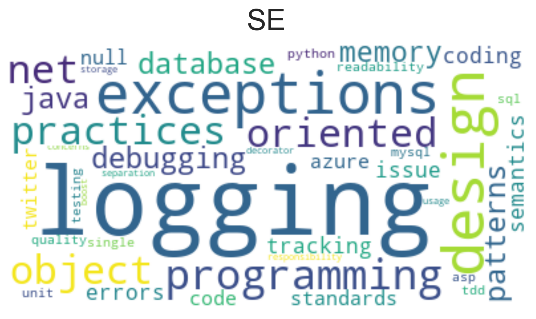
(a) AE



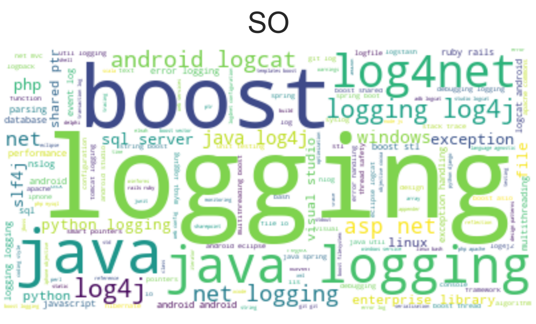
(b) DB



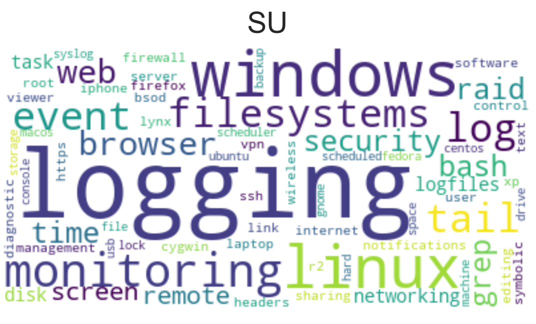
(c) SF



(d) SE



(e) SO



(f) SU

Figure 4. Word cloud of tags of logging questions that received more than 5 answers

are related to logging libraries such as *log4net, log4j, SLF4J.* Table 5 provide more details about the questions that we analyzed.

RQ2 conclusions: Approximately 1.06–11.61% of all the logging questions invite a great amount of discussion.

Table 5. Analysis of the questions that received ≥ 5 answers: StackOverflow: SO, SuperUser: SU, ServerFault: SF, DBA: DB, SoftwareEngineering: SE, Android: AN

S. No.	WB	Tag	ID	Context
1	AE	root-access	157	A rooted device can monitor the **logcat (a command line tool that dumps a log of system messages)** stream on the phone for this he needs root access.
2	AE	touchscreen	13992	is there an existing app that could be installed and could **record touch interactions** on the background?
3	AE	Android logging	14430	Android logging can be viewed and examined by use of **adb logcat(android debug bridge it can control device over USB from a computer)**, **alogcat(software testing tool that control full control over intents)**, **logcat extreme (user interface that records and monitors logcat)**.
4	AE	data connection, data-monitoring, celluar-radio	35702	**Logging information about data-connection**, rate along with the location can be monitored by using phone's radio without sending any additional data which is available through logcat or through network buffer on user's phone using alogcat or adb.
5	DBA	sql-server, transaction-log, dbcc, database-size, shrink	41215	Shrinking of database on SQL server using DBCC while **transaction log** showed that no transactions where open can be done using log_reuse_wait_desc query and further sp_removedbreplication query to remove replication related objects.
6	DBA	sql-server, transaction-log, r2, backup	45876	The only difference between full backup and copy-only-full-backup is that the full backup does not break up the differential log chain while neither of them breaks the **transaction log** chain nor they truncate the transaction log file.
7	DBA	sql-server, transaction-log, backup, delete	13757	Prior taking the backup, to safely remove the SQL server transaction log file use sp_detach_db procedure. This procedure make sure that SQL server records the fact that database was shut down cleanly.

Table 5 continued

S. No.	WB	Tag	ID	Context
8	DBA	sql-server, backup, transaction-log	162628	Taking a **transaction log** backup and truncating the log and then deleting the **transaction log** backups this would only work if the tool is using its own tracking to know which log files are to be restored for the proper functioning of log-chain.
9	DBA	sql-server, transaction-log, maintenance	12474	During the maintenance of **transaction-log** file it should be make sured that there are no error in backing up transaction log otherwise file size would grow and system would run out of space.
10	SF	monitoring	1845 24428 53000 53699 53894 437369	Here monitoring of different types of **log files, systems** and their different features is being made.
11	SF	syslog, log-files	96720 42527 49042 62687	Different tools like logrotate, splunk linked to **syslog log** files are studied.
12	SF	Apache	322116 355311	To catch all access log with Apache virtual hosts, to write useful awk and grep scripts for Apache logs and such other log based features based are provided by Apache.
13	SE	exceptions	15502 20109 130250 272771 306032	Different features to handle different types of **logging exceptions**.
14	SE	object	82, 499 230, 131	It provides features like **best design perspective for logging**, need of logging while doing TDD(Test-driven development).
15	SE	debugging	84, 301 225, 903	It explains use of the concept like **timestamping, maintaining transactions log and logging** for the purpose of debugging.
16	SE	design	27, 595 782, 499 153	It provides different design perspective for **best logging practices**.

Table 5 continued

S. No.	WB	Tag	ID	Context
17	SE	programming	2, 727 713, 729 415, 500	It tells us to write **efficient programs** which make significant use of logging.
18	SU	monitoring	103, 222 143, 658 226	It explains **monitoring concepts of logging** from the network perspective.
19	SU	filesystems	22, 674 420, 321 236, 100	It **explains filesystems with respect to logging issues**.
20	SU	Linux	106073 222912 226744 330590 351387	It helps in **logging different processes, files** in operating system Linux (Ubuntu).
21	SU	Windows	153 106073 145086 219401	It helps in **logging different processes and files** in OS Windows.
22	SO	log4net	192456 756125 50599689 50591008	The **Apache log4net** library is a tool to help the programmer output log statements to a variety of output targets.
23	SO	log4j,	1140358 12532339 728295	**Apache Log4j** is a Java-based logging utility.
24	SO	logcat	7959263 2250112 3280051 19897628	**Logcat** is a command-line tool that dumps a log of system messages, including stack traces when the device throws an error and messages that you have written from your app with the Log class.
25	SO	boost	34362871 17844085 39247778 34394896	Boost is a set of libraries for the C++ programming language that provide support for tasks and structures such as linear algebra, pseudorandom number generation, multithreading, image processing, regular expressions, and unit testing.
26	SO	SLF4J	11916706 7421612 8965946 14024756 11639997	**SLF4J** or Simple Logging Facade for Java provides a Java logging API by means of a simple facade pattern.

3.6.3. RQ3: How much time it takes to get the accepted answer for logging questions?

Motivation: In this RQ, we compute time taken by the logging questions to get an accepted answer. We believe that answer to this RQ can be beneficial in identifying type of logging issues that are most time consuming. For example, server related issues can be more time consuming as compared to others.

Approach: We perform three types of analysis to answer this RQ. First, we compute the average time taken by the logging questions to get accepted answer. Second, we compute average time taken by the logging questions to get accepted answer for each year separately for all the six websites. We also compute standard deviation for both the analysis. Third, we create box plot of the time taken by the logging questions to get the accepted answer. We compute both average time graph and box-plot because average computation is affected by outliers and can give mis-leading results, whereas in box-plot analysis all the outliers are clearly visible.

Results: Figure 6 shows the average acceptance time for all the logging questions for all the years combined. Figure 6 shows that mean time to get accepted answer for SU and SF websites are much higher as compared to other websites. The mean time of acceptance of SU and SF websites in 36761.93 (in hours) and 26034.32 (in hours), respectively. The standard deviation of SU (198665.32) website is much higher as compared to other websites, i.e., AE (76106.25), DB (119476.32), SF (159452.76), SE (167535.09), and SO (138574.83).

Figure 5 shows the average acceptance time for all the logging questions for all the years separately. We observe that for each website whenever there is an increase in the mean acceptance time, there is corresponding increase in the standard deviation, which indicates presence of potential outliers, i.e., questions that took a large amount of time to get the accepted answer in all the websites. Additionally, we observe presence of several questions which took almost 0 time in getting accepted answer. For example, SO has 156 questions that took 0 second to get an accepted answer. Figure 5d shows that the mean time taken by the SE questions is very less for all the years. In year 2012, there are some question in SE that took much longer to get the accepted answer. The SE website invites questions on programming practices. If there is some question which is taking a large amount of time, it can indicate some fundamental programming issue or concern with logging in which software developers are facing problem. We analyze five questions on SE website that took large amount of time to get accepted answers (refer to Table 6). Following is the detailed analysis of two such questions:

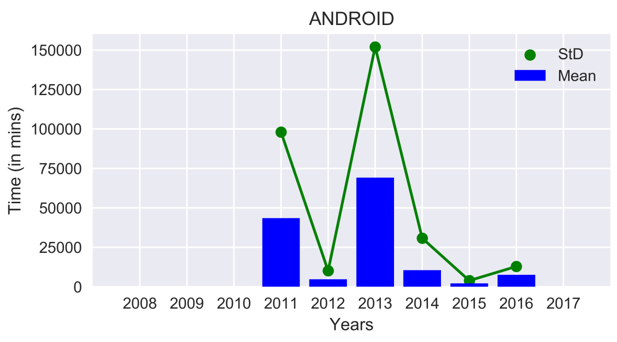
In question 1 (id: 291757), the user is asking about "a better method to handle precondition and logging". The experts suggested the user to use *throw* and *assert* statements. Following a snippet of expert comment:

> When implementing this, you should rarely decide on how to handle the error, at the place where it occurs; instead, you should throw an exception and let client code decide.

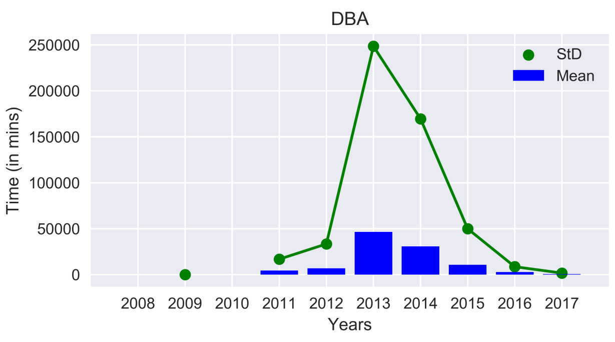
In question 2 (id: 208471), the user was asking a fundamental question about a better method between *file* or *database* to use for logging. The expert suggested the user to use file as he was not needing any complex processing of the log used. Following is the snippet of expert comment:

> Both options seem valid to me. In such cases, a useful rule to apply is to do the Simplest Thing That Could Possibly Work. Text files are easier to get started with and are expected to work reasonably well at least in the beginning. Once requirements arise that are better satisfied using a database, it will be trivial to import them. Using this strategy, you postpone design decisions as long as possible (but not longer than that). As such, you don't do unnecessary work. When, if ever, it will be needed, you will have a much better understanding of what exactly it is you need. Hence, you are more likely to build the Right Thing and not waste time building the Wrong Thing.

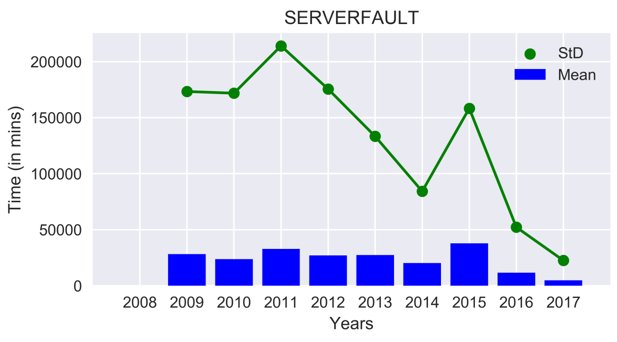
Figure 7 shows the box plot of the time taken by the logging questions to get the ac-



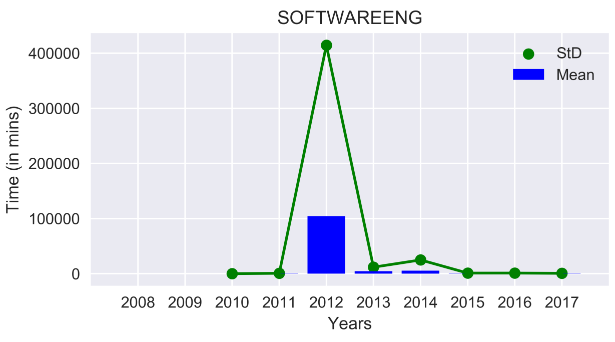
(a) AE



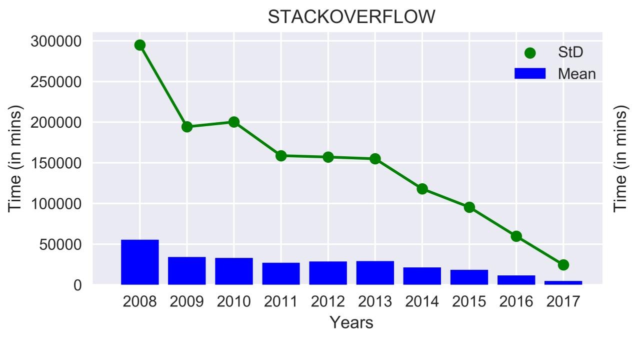
(b) DB



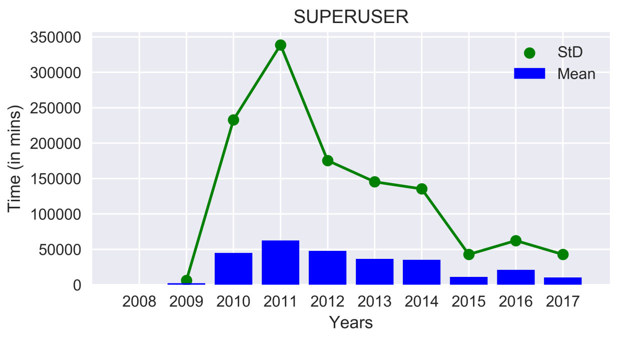
(c) SF



(d) SE



(e) SO



(f) SU

Figure 5. Mean time to get accepted answer for logging questions

cepted answer. Figure 7 shows that the median acceptance time of logging questions on the AE website is much higher as compared to that of other websites. For example, the median acceptance time (in minutes) for logging question on the AE website is 3093.39 (2011), 113.2 (2012), 4475.32 (2013), 292.30 (2014), 790.49 (2015), 277.74 (2016) whereas for the SO website is the median acceptance time is 46.96 (2011), 65.41 (2012), 78.43 (2013), 111.94 (2014), 139.91 (2015), 151.86 (2016). There can be several reasons for this kind of behavior for example may be AE logging questions are more complex as compared to other logging questions or there can be lack of user participation of the AE website as compared to other websites.

RQ3 conclusions: The *mean time* to get accepted answer for logging questions on SU and SF websites are much higher as compared to other websites, whereas, the *median acceptance time* of logging questions asked on AE is much higher as compared to the other 5 websites.

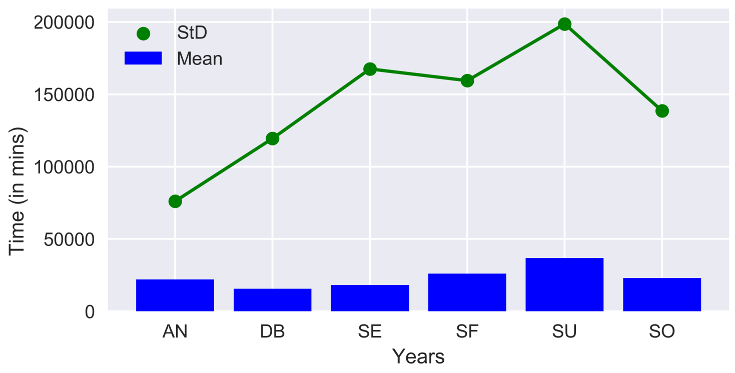


Figure 6. Combined mean

Table 6. Top 5 most time consuming questions on the SE website

S. No.	Question Id	Answer Id	Question	Time (in minutes)
1	291757	292108	Better way of handling pre conditions and logging	4 647.95
2	208471	209261	Is SQLite a sensible option for data logging?	10 154.03
3	220557	223273	Finding patterns in logs	44 275.5
4	232143	244595	Strategy to store/average logs of pings	130 175.27
5	149346	298292	Logging asynchronously – how should it be done?	1 761 970.55

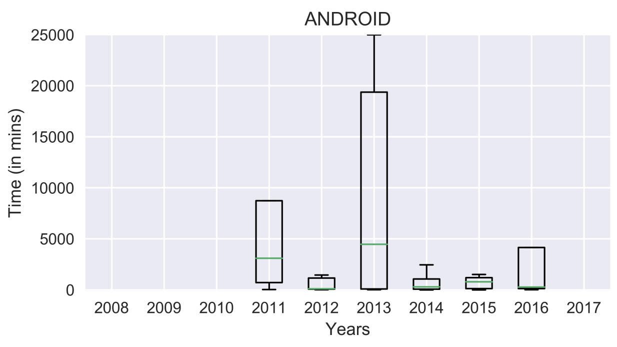
3.7. RD2: Programming language analysis

This research dimension provides an insight into the development of source-code logging tools and libraries. Results would be helpful to analyze the dependency of programming language with source-code logging across various platform. Results estimate programmer's interest and discussion with respect to various languages. This will help developers to choose a programming language for developing logging tools based upon various environment and platforms. For Example, Server (SF) and super-user (SU) oriented applications suggest a large interest in C-based logging tools while for software engineering, C#, Java, and C++ seems to be a viable option.

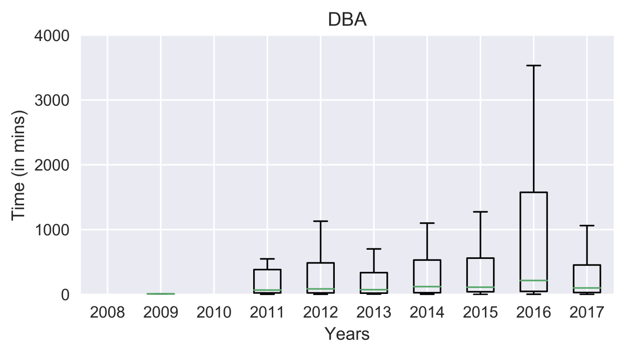
3.7.1. RQ4: How pervasive is software logging related questions on community based Q & A websites across programming languages over the years?

Motivation: In this RQ, we analyze the distribution of logging question in different programming languages over past several years. Logging frameworks for various programming languages can vary in-terms of their capabilities and performance with respect to features such as type-safety, thread-safety, flexibility and portability. Our objective is to gain insights on the quantity of questions asked on logging frameworks for multiple programming languages. We believe that answer to this RQ, can be beneficial in identifying the programming language(s) in which software developers face most of the logging issues. It can also be beneficial in identifying the languages in which logging interest is increasing or decreasing. Answer to this RQ, can be used to tune future logging automation or prediction tools.

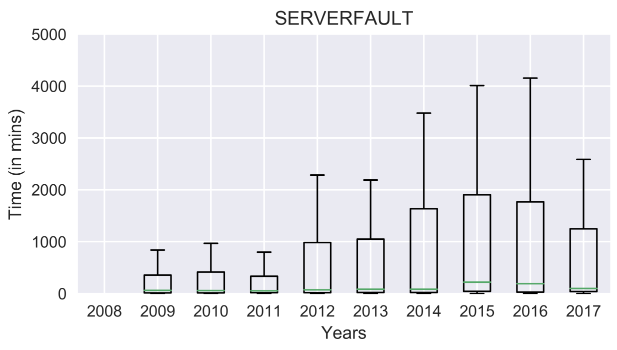
Approach: To answer this RQ, we select top 6 programming languages: C, C++, C#, Java, Python, and JavaScript. Next, we extract all the logging related questions for these six programming languages. We collect data from all the six websites considered in this work, i.e., SO, SU, SF, DB, SE, and AN. We extract these questions in two steps:



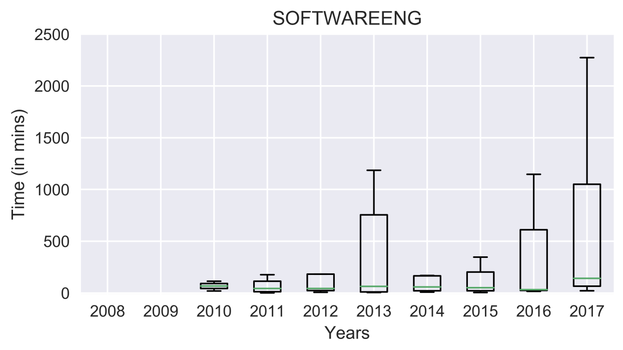
(a) AE



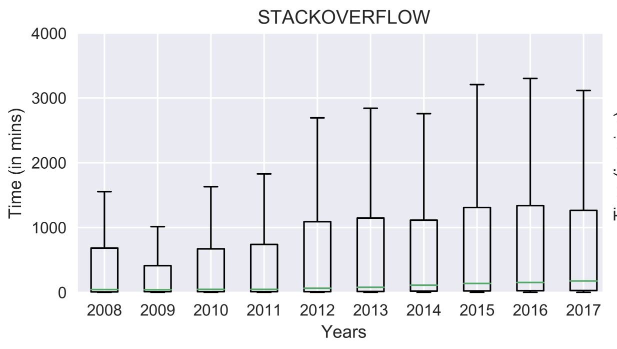
(b) DB



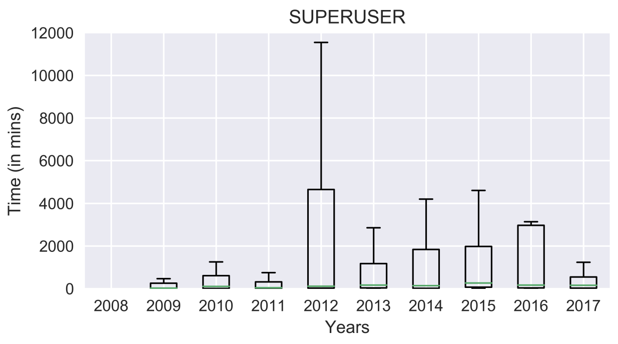
(c) SF



(d) SE



(e) SO



(f) SU

Figure 7. Box Plot showing time to get accepted answer for logging questions

– First, we manually identify the tags related to popular logging libraries used in these six programming languages. Table 7 shows the list of tags that we identified. We select all the questions consisting of any of these tags. Logging questions are then assigned to their respective programming languages.
– Second, we select all the questions that consist of any of the programming language tag, i.e., 'C', 'C++', 'C#', 'Java', 'Python', 'JavaScript'. From these questions, we filter all the questions that consists of any 'logging tag'. The logging questions are then assigned to the respective programming language.

Results: Figure 8 present the number of logging questions asked in each year (2008–2017) for all the six programming language considered in this work. In this Figure, y-axis represents the number of logging questions asked in each year. We kept the y-axis scale same for all

Table 7. Tags that we used to extract questions related to programming languages

Programming language	Tags
Java	Log4J, SLF4J, tinylog, logback, Apache Commons Logging, google-cloud-java, commons-logging, jboss-logging, syslog4j, otroslogviewer, log4j, log4j2, log4jdbc, jul-to-slf4j, hyperloglog.java, java.util.logging
C	Log4C, sclog4c, syslog, zlog, zf_log, log4c
C++	glog, log4cplus, pantheios, boost::log, easylogging++, log4cxx, boost, boost-log, boost-logger, boost.log, spdlog, log4cpp
Python	pygogo, Logbook, google-cloud-python, django-logging, logger-python, hyperloglog.python, unified-log, auth.log, graylog, graylog2
C#	log4net, NLog, Enterprise Library, Common.Logging, log4net-configuration, log4net-appender, log4net-filters
JavaScript	js-logging, Log4js, log4JavaScript, JSNLog, Node-Loggly, Bunyan, Winston, Morgan, Angular-Loggly, loglevel, jsnlog, log-level, logsene-js, node-nslog, truncate-logs-js

the programming languages for better visualization of logging questions trend across the website. We draw several interesting observations from the Figure 8. First, it shows that users have asked the highest number of logging related questions for Java and C++. A total of 51723 logging questions are asked on these websites out of which 73% (i.e., 37935) questions belong to Java and C++. Second, it shows that the number of logging questions are increasing for Python and JavaScript. For example, the number of logging questions asked in Python are : 11 (2008), 66 (2009), 149 (2010), 247 (2011), 323 (2012), 465 (2013), 512 (2014), 571 (2015), 701 (2016), 789 (2017). We also observe an increasing trend for logging questions in Java (except in the year 2016). Whereas, we observe a decreasing trend for logging question in C++ and C# after the year 2013.

RQ4 conclusions: A majority of logging questions belong to C++ and Java programming language. The trend of number of logging questions is increasing for Java, Python, and JavaScript, whereas it is decreasing or constant for C, C++, C#.

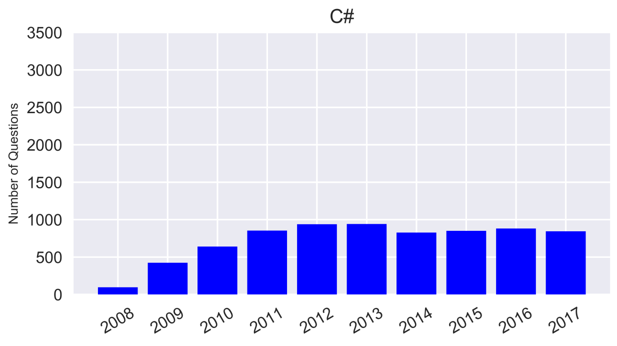
3.7.2. RQ5: What is the trend of logging questions with respect to different programming language for each Q & A website?

Motivation: In this RQ, we analyze the distribution of logging questions with respect to different programming languages for different websites. Each website represents a specific domain. Analysis of the logging questions asked in different programming language with respect to different website can be beneficial identifying the programming language in which most of the questions arise on that domain.

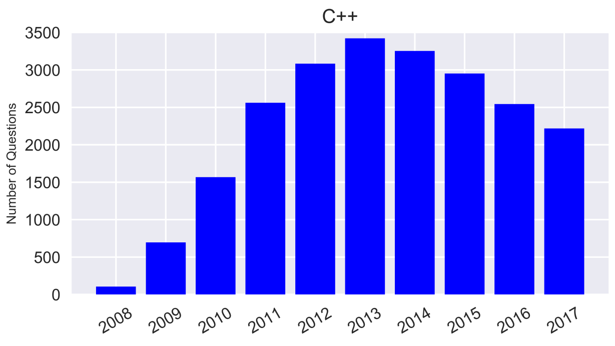
Approach: To answer this RQ, we compute total number of logging questions asked in each year on each website. For each website, we grouped the logging questions with respect to each programming language.

Results: Figure 9 shows the programming language wise trend of logging question with respect to each website. The results of the RQ4, show that Java, C++ and C# are dominant programming language in which most of the logging questions are asked. However, results of the RQ5 show that different programming language show dominance (language in which the highest number of logging questions are asked) for different website. For example, for the SO website C++ and Java are dominant language whereas for the SF and SU website, ‘C’ is the dominant programing language. For the SE website, we did not observe any particular trend with respect to different programming languages. The AE and DB websites consists of very few logging questions and hence, we not able to extract any significant insight.

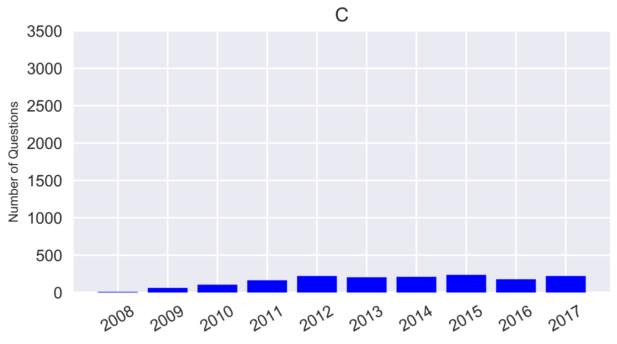
RQ5 Conclusion: Different websites have different programming language that is dominant. For the SO website C++ and Java are dominant



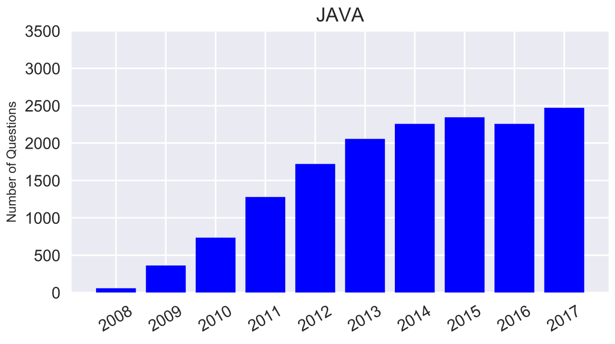
(a) C#



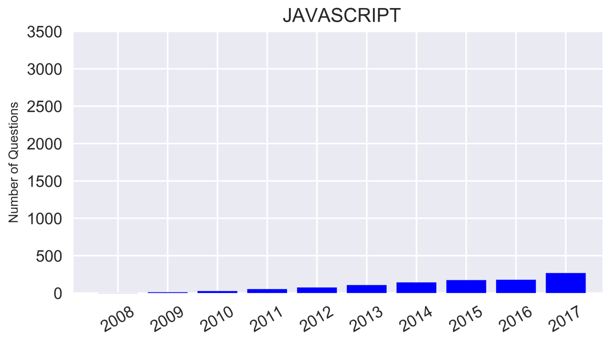
(b) C++



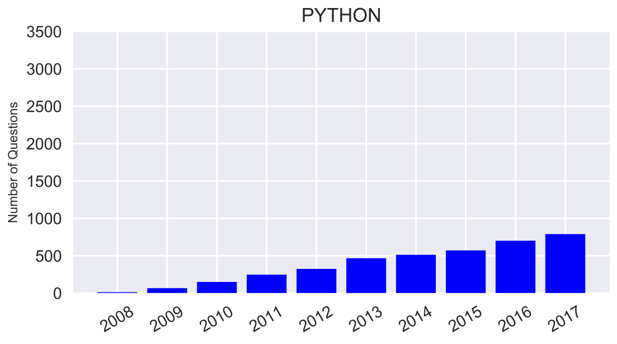
(c) C



(d) Java



(e) JavaScript



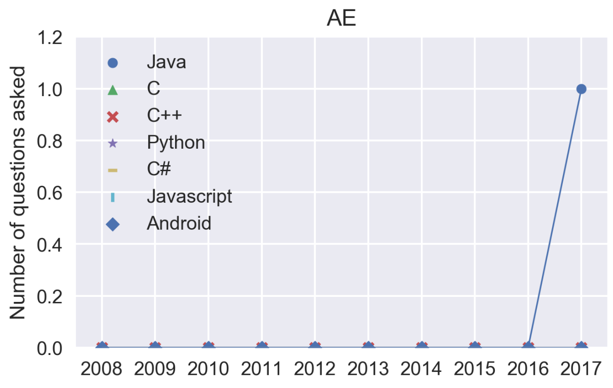
(f) Python

Figure 8. Count of logging questions for various programming languages

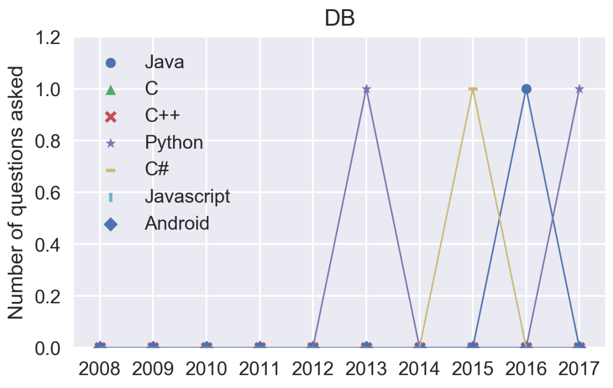
language whereas for the SF and SU website, 'C' is the dominant programming language.

3.8. RD3: Content analysis

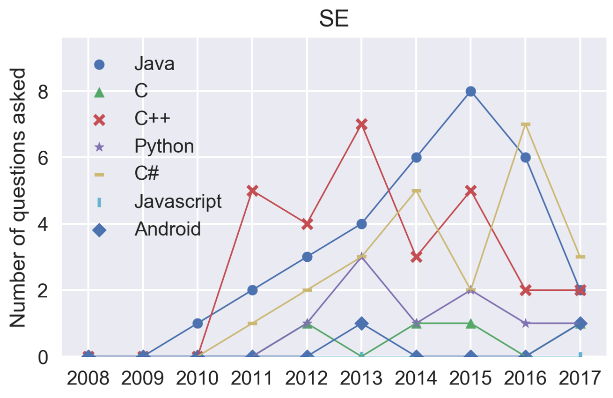
Source code logging exists in various forms and across various disciplines. Results of this dimension provide an overview of most discussed logging topics. This insight will help developers tokeep in mind these discussions while developing logging tools and libraries. It would also aid to keep a track of logging-related issues and needs of programmers. Additionally, it also provides insight about cross-discipline logging tools and practices, for example, the transaction log is used in both server environment (SF), StackOverflow (SO) and database (DB) while event-logging practice is observed in super-user (SU) and StackOverflow (SO).



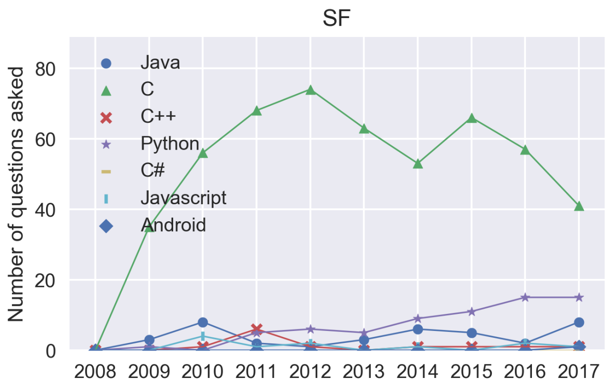
(a) AE



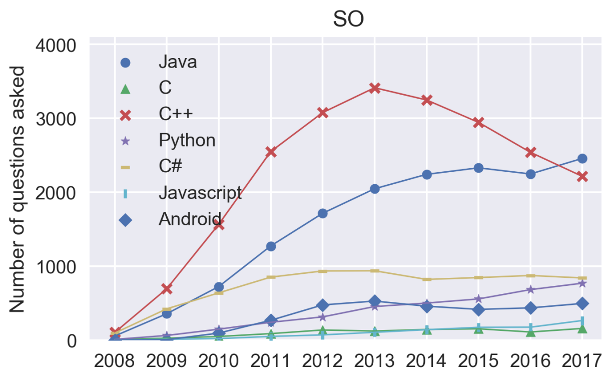
(b) DB



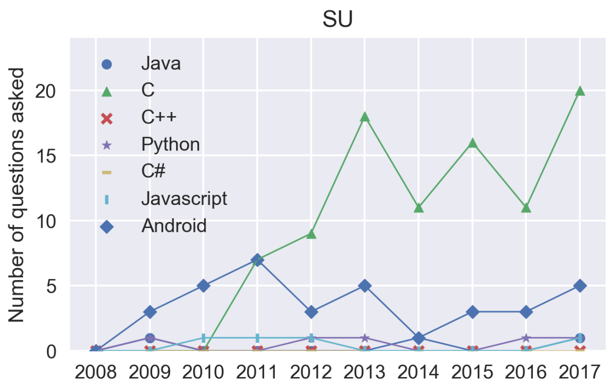
(c) SE



(d) SF



(e) SO



(f) SU

Figure 9. Count of logging questions for various programming languages for each of the six websites

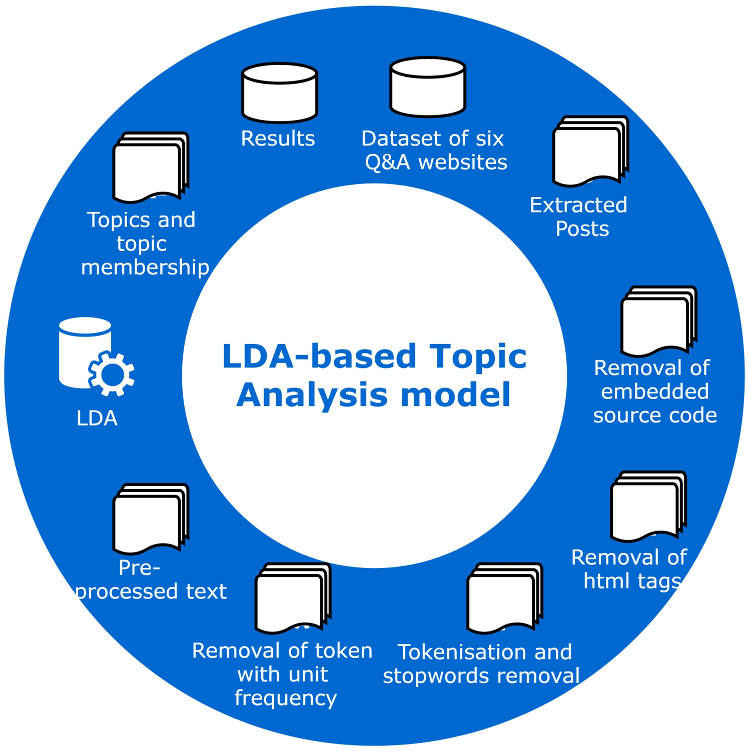

LDA-based Topic Analysis model
Results
Dataset of six Q&A websites
Extracted Posts
Removal of embedded source code
Removal of html tags
Tokenisation and stopwords removal
Removal of token with unit frequency
Pre-processed text
LDA
Topics and topic membership

Figure 10. Pre-processing steps of LDA

3.8.1. RQ6: What are the main discussion logging topics in various websites?

Motivation: In this RQ, we analyze main *discussion topics* present in the logging questions. Identifying major logging discussion topics can be beneficial in finding logging tools and libraries in which software developers face most difficulties. Since, we are analyzing logging topics on six websites, it can also be beneficial in identifying types of problems that developers face on each website.

Approach: We use LDA algorithm for identifying topics present in the logging questions in each website. We use Python library Gensim [34] in our work. LDA algorithm require three input parameters: *corpus*, *number of topics*, and *number of iterations*. Hence we first create our corpus. We extract *title* and *description* of each question to build the initial corpus. We apply five pre-processing steps to clean the initial corpus (refer to Figure 10 for details). First, we remove all the source code snippets from the description of the questions, i.e., we remove all the content between '<code>' and '<code>' tag. Second, we remove all the HTML tags such as '<p>, <a herf... >' from the description of the questions. Third, we remove all the English stop words, i.e., 'the', 'is' etc. Stop words are non-content bearing terms that do not add meaningful insight towards our goal. Fourth, we apply stemming to convert words to their root form. For example, the term 'programmers' is converted to 'programmer'. We use Porter Stemmer algorithm for stemming [35]. Fifth, we remove all the words that occur only once in the corpus. LDA takes *number of topics, i.e., K,* as the second parameter. Since, there is no best value known for K that is suitable for all types of dataset. We vary the value of K from 10 to 50 and select the value giving the best results. The third parameter to LDA algorithm in the number of iterations. In this work, we set the number of iterations equal to 500.

Results: We analyze results given by the LDA algorithm. The results showed that for all the websites we were getting meaningful topics for $K = 50$, hence, we selected this value. Table 8 shows some of the topics discovered for each website. Following is the detailed discussion about the topics obtained in each website:

SO: We discovered logging topics related to different domains on SO website. For example, logging topics in OOP, classes etc. Logging topics in different programming languages like Java, Python, C#, C. We also find topics related to programming features like object oriented programming language, file handling, class features. Table 10 shows an illustrative example of a logging question for *Python* programming language asked on the SO website.

SF: In SF website, we observe logging topics related to issues to various servers like email server, tomcat server, Apache server. In addition, we observe topics related to networking and syslog. Table 10 shows an illustrative example of a logging question related to *networks* asked on the SF website.

Table 8. Popular logging topic identified in all the six websites

Android Enthusiast, $K = 50$	
Topics	Words
document processing in system	document, write, store, readonly, writing, system
log in Android applications	adb, logcat, applications, Android, device, bootloader
log in tablet	adb, tablet, file, files, log, applications
log in Android phone	log, phone, app, Android, apps, time
log in device	log, logs, apps, storage, device, problem
apps	apps, google, exception, phone, install, exit
log in network connection	log, WiFi, network, IP, DHCP, connect, logcat
Database Administrators, $K = 50$	
Topics	Words
Backup	Restore Backup, log, transaction, backups, restore, recovery
Binarylog in MySQL	Binlog, MySQL, Binlogs, Binary, Logs
databases	null, bigint, int, varchar15, commit, set
rebuliding indexes	index, rebuild, progress, task, query, source, end
SQL features	table, rows, column, insert, values, id
memory allocation	reserved, allocated, commited, pages, kb, node
transaction-log in databases	select, query, transaction, log, table, join
log in databases server	logs, db2, server2, server, databases, transaction
binlog in MySQL server	binlog, mysqlbinlog, query, MySQL, server, endlogpos
PostgreSQL	master, slave, Postgres, PostgreSQL, archive, wal
log in SQL databases server	log, SQL, database, databases, server, mirroring
log in Oracle databases	redo, Oracle, database, log, logs, files
Serverfault, $K = 50$	
Topics	Words
log in network	TCP, UDP, log, port, lo, accept
syslog in messaging	syslogng, log, destination, source, get, messages
log in email server	ip, email, mail, Logwatch, server, postfix
log in Tomcat server	log, file, Tomcat, logs, server, files
log in Apache server	Apache, log, errorlog, logs, acesslog, server
different syslog in messages	rsyslog, syslog,varlogmessages, log, logging, logs
log in SQL server	server, log, database, server, backup, transaction
StackoverFlow, $K = 50$	
Topics	Words
OOP	int, include, class, void, char, const

Table 8 continued

features of class	public, void, static, class, private, new, import, null, return, string
log in Python	logging, logger, import, Python, log, logback
log in Java	log4j, spring, maven, class, log4j2, log4jwarn, logger
file handling	file, included, main, line, appender, stdout
programming features	undefined, reference, const, function, stdallocator, external
log in files	file, log, files, logs, line, write
log in Java Tomcat	file, log, log4j, logger, Tomcat, log4jproperties
C programing	include, const, return, int, typedef, function
log in Python file	logging, logger, file, log, Python, logback
log in file using C#	log4net, appender, file, log, using, config
Superuser, $K = 50$	
Topics	Words
log in Windows computer	Windows, log, application, logs, computer, service
log using USB	USB, device, plugged, logs, file, drive
syslog in message server	syslog, server, log, syslogd, mesaages, information
log in opengl file	system, opengl, log, logs, file, extension
log in file using command line	file, log, command, log, commands, output
Software Engineering, $K = 50$	
Topics	Words
Java	public, void, static, try, catch, throw, class, exception
log in client server	log, client, server, clients, logger,request
log in OOP's	exception, log, throw, method, catch, logging
software	software, license, disclaimer, copyright, warranty, warranties
log in databases	log, database, table, SQL, file, user
log in user application	log, user, application, logging, logged, data
log and exceptions	logger, log, exceptions, exception, logging, logginggetlogger
logging in project	project, logging, compiled, library, shared, core
log in files	log, files, appender, file, tests, application
multithreading in C in Unix	multithreading, Unix, C, code, library, boost
log in applications	logs, log, message, loglevel, application, information
log in systems	system, logging, libraries, log, developer, exception
logging in languages and OS	log4cxx, log4j, C, Java, Windows, Linux

Table 9. Categorical detail of observed logging-related tags

Category	Name	Tags	Websites	Description
General	Logging	logging	AE, DB, SE, SU, SO	Consider base of our analysis. Logging tag is used in all the six websites.
	Log-shipping	log-shipping	DB, SF, SO	It is a process of restoring transaction-log files on a standby server after creating a backup of transaction-logs on a primary database
	Log Files	logfiles, log-files, transaction-log	SO, SF, SU	It is a record of events that occurs in an operating system, software runs etc. Transaction Log and Event logs forms a sub category of log files [36].

Table 9 continued

Category	Name	Tags	Websites	Description
Types of Logging	Transaction-log	transaction-log	DB, SO, SF	A transaction log is a log of communication or transactions between a system and clients of that system.
	Event Log	event-log	SO, SU	Event logs aims to provide an audit trail that can be employed to understand the activity of the system and to diagnose problems. They forms the basis of understand activities of complex systems such as server applications.
	Error-log	error-log, error-logging	SO, DB, SU	Error-log is the collection of errors encountered during execution of a program.
	Binary-log	binlog, binarylog	SF, SO, DB	A binary log consist of binary log files and an index and is similar to transaction-log. They are used to restore data after backup.
Syslog and Syslog utilities	Syslog	syslog	SO, SF, SU	Syslog is the standard protocol for message logging.
	Rsyslog	ryslog	SF, SO	It is widely used in Unix and Unix-like operating system as a utility for transferring log messages in an IP network. It extends basic syslog protocol to content-based filtering along with providing features such as using TCP for transport.
	syslogd	syslogd	SF, SU	It provides provision for system logging and kernel message trapping. It supports both local and remote logging.
	syslog-ng	syslog-ng	SO, SF	syslog-ng extends syslogd model by adding content-based filtering, flexible configuration, TCP transport etc.
Logging Tools and Libraries	Graylog	graylog, graylog2	SO, SF	Graylog is an Open-Source log capture tool and provides analysis solution for operational intelligence.
	NXLog	nxlog	SO, SF	NXLog is a log collector and supports log collection from multitude of sources and formats e.g. event logs from TCP, UDP, file, databases, syslog, Windows event log etc.
	Logparser	logparser	SO, SF, DB	Logparser is a command line tool designed to automate tests for Internet Information Services (IIS) logging.
	Logwatch	Logwatch	SF, SU	Logwatch is a log parser and analyser.
	Logstash	logsash	SO, SF	Logstash is used for managing events and logs. It deals with log processing, storage and searching.
	Hugo	hugo	SO, SU, SF	The Hugo logging plugin is used to log debug statements with the help of annotations.
	Log4j	log4k	SO, SF	Log4j is Java-based logging utility developed by Apache Software Foundation.
	Lynx	lynx	SU, SO	Lynx is the Android logging library.
	mysqlbinlog	mysqlbinlog	DB, SO	It is a utility used to process binary and relay logs.

Table 9 continued

Category	Name	Tags	Websites	Description
	Boost	Boost	SO, SF, SE	Boost is a C++ based logging library.

SU: In SU website, we find logging topics related to Windows, USB, message server, command line etc. Table 10 shows an illustrative example of a logging question related to *USB* asked on the SU website.
DB: In DB website, we observe logging topics related to various domains like backup, SQL features, indexes, memory allocation. Additionally, we observe topics related to DB servers like Oracle and MySQL servers. Table 10 shows an illustrative example of a logging question related to *transaction-log* asked on the DB website.
AE: In AE website, we observe topics related to document processing, tablet, Android phone, and logs in network connection. Table 10 shows an illustrative example of a logging question related to *networking* asked on the AE website.
SE: In SE website, we obtain a wide variety of logging topics related to OOP, Java, files, databases etc. Additionally, we observe topics related to exception and multi-threading. Table 10 shows an illustrative example of a logging question related to *object oriented programming* asked on the SE website.
RQ6 conclusions: For each website, we obtain logging topics related to different features such as programming language, transaction log, networking (Tab. 9).

3.8.2. RQ7: What is the distribution of logging-related tags across various Stack Exchange websites? And how persuasive is the commonality between these tags along various Stack Exchange websites?

Motivation: In this RQ, we analyse logging-related tags. Logging related tags mostly represent logging libraries, tools and technologies. This is a pressing need of our analysis to investigate distribution of these logging tags across various Q & A websites. This can be beneficial to find common logging libraries across six-websites as tags are the medium of classification on these websites. Distribution of these logging tags across various websites will provide us cues regarding common logging tools and libraries employed across various environments their trends. This may help the developers to design a common tool across different platforms capable of solving multiple problems.
Approach: In order to provide a compendious analysis, for this RQ, we have considered logging-related tags that are present in at-least two websites.
Results: Results of this analysis is divided into four categories, first, General Logging tags, second, Syslog and Syslog-based utilities, third, types of logging, and fourth, logging tools and libraries. Figure 11 depicts the observed results of our analysis. Center of the bubble depicts logging-related tags while its diameter corresponds to observed frequency. Following are the results of each category:
General Logging tags: This category consists of tags namely logging, log-files and log-shipping. The chief objective of log shipping is to ensure high availability of database by creating backup server that can replace production server quickly. It is used by both server engineers and database administrator on SF and DB respectively along with that this technique is further supports by well-known servers and databases namely Microsoft SQL Server, 4D Server, PostgreSQL and MySQL [37–39].
Syslog and Syslog-based utilities: Syslog is the standard protocol for message logging. Syslog is commonly used for system management, security auditing and debugging analysis [40]. It provides provision for system logging and kernel message trapping. Many utilities extends syslog-based models, two of them are namely Rsyslog and Syslog-ng. These are commonly used

Table 10. Example post from various websites of the selected topic: WS: Website, QID: Question Id

WS	Topic	QID	Title	Body
SO	log in Python	41666158	log4j/logback pass logger level as a parameter	I want to do something which seems really straightforward: just pass a lot of logging commands (maybe all, but particularly WARN and ERROR levels) through a method in a simple utility class. I want to do this in particular so that during testing I can suppress the actual output to logging by mocking the method which does this call.
SF	log in network	508349	Rsyslog not logging from remote server	I am trying to set up a centralized log server. I have central server (A) receiving logs via a remote server (B) on port 514. I know it is receiving these. Here are a few entries from a tcpdump on port 514... I have made sure to restart rsyslog every time I edit rsyslog.conf and I am running the start daemon with the -r and -t flags, even though they are deprecated in my current version. So why isn't anything coming in on port 514 being written to test.log?
SU	log using USB	849950	Logging when someone connects or removes a USB device to/from a Windows machine	I am currently trying to find a way to log all of the connections and disconnections of USB devices from all of the Windows machines on our network. This information needs to automatically be logged to a file on the machine, this file can then be read by nxlog and then get shipped to our centralised logging platform for processing. I was hoping that this information would be logged by Windows logs automatically, but I found that while some information about USB removable storage appears to get logged to Event Viewer, this is quite limited information and doesn't pick up when USB keyboards and mice are connected and disconnected...
DB	Transaction log	6996	How do I truncate the transaction log in a SQL Server 2008 database?	How do I truncate the transaction log in a SQL Server 2008 database? What are possible best ways? I tried this from a blog as follows: 1) From the setting database to simple recovery, shrinking the file and once again setting in full recovery, you are in fact losing your valuable log data and will be not able to restore point in time. Not only that, you will also not be able to use subsequent log files. 2) Shrinking database file or database adds fragmentation. There are a lot of things you can do. First, start taking proper log backup using the following command instead of truncating them and losing them frequently.
AE	log in network connection	85114	Does Android save a log of its own IP addresses?	I have a WiFi network to which I connect at work. The IP address has always been DHCP, but today the DHCP server is down. If I can figure out what IP address I had, I can set it statically (after checking to make sure another device hasn't already taken it, via ping from my desktop). Does Android have a log anywhere of the IP addresses it is leased? I have root and thus can look at any file on my phone.
SE	log in OOP's	255372	Logging exception in multi-tier application	I'm building a multi-tier enterprise application using Spring. I have different layers: Controller, Business and Provider. Within the application I've built a custom error handling mini-framework that is based on a single RuntimeException which has an error code to discriminate different kind of errors...

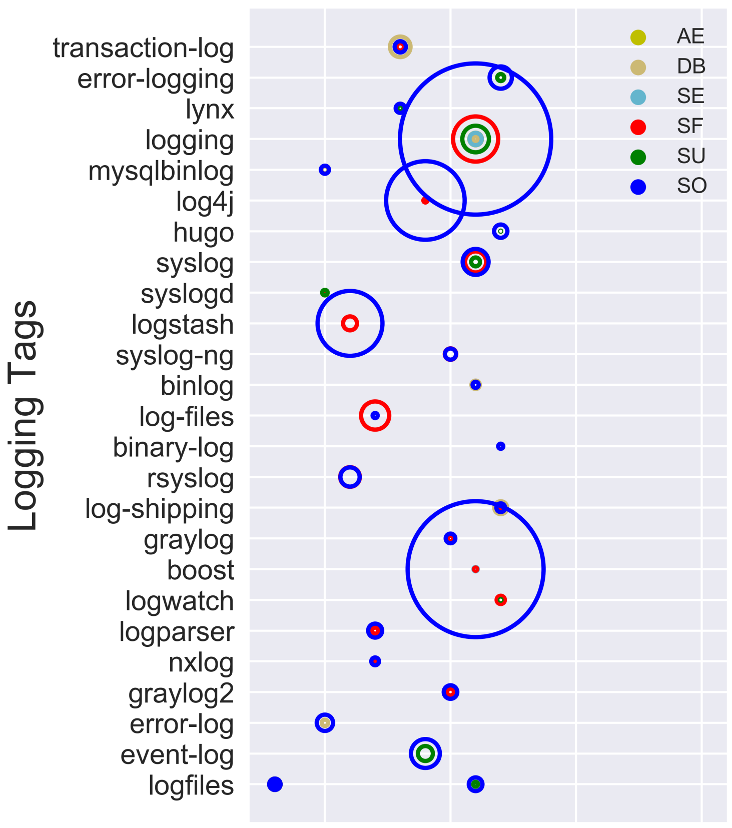


Figure 11. Bubble diagram of tags that are present in more than one website

by system programmers of SO and SF. Rsyslog website claims it to be Swiss army knife for logging [41]. Rsyslog extends basic syslog protocol to content-based filtering along with providing features such as using TCP for transport while Syslog-ng extends syslogd-model and thereby enhancing existing features [42, 43]. Both of these utilities are used for system logging while it is observed that Rsyslog is more popular on SO and SF than syslogd depending upon their tag frequency.
Types of Logging: Error Logging, Transaction logging, Event-logging and Binary-logging are some of the popular types of logging used by programmers of SO, SF, DB and SU. Error-logs are widely used for troubleshooting and bug fix [44]. Error logging is observed on SO, DB and SU. Event logs aims to provide an audit trail that can be employed to understand the activity of the system and to diagnose problems. They forms the basis of understand activities of complex systems such as server applications. Event logging is observed in SU and SO Highest number of transaction-log tags and binary log tags are observed in DB. This may be attributed to the fact that in order to allow the database to recover from crashes or other errors and to basically maintain consistent state, most of the databases maintain a transaction log [45]. Binary log is similar to transaction log, it records all changes in the databases including both data and structure [46].
Logging Tools and Libraries: Many logging tools and libraries are used by programmers to manage and process logs. Some of the popular logging tools detected in our analysis are Graylog, Graylog2, NXLog, Logparser, Logwatch, Logstash, Hugo, Lynx, Log4j and mysqlbinlog. Graylog and NXLog tag is frequently observed in both SF and SO. Usage of Graylog may be attributed to the fact that Graylog is a server that collects log messages along with that provides an interface for analysis and monitoring [47]. This tools is frequently used by Server Administrators while NXLog's high-performance I/O layer make

it capable of handling thousands of parallel client connections in order to process huge log volumes [48]. Thus making it suitable for use in Server Environments. Logwatch provides feature to deliver a unified report of all activity on a server through command line or email to the administrator [49,50]. Logparser is designed to automate tests for IIS logging. IIS is an extensible web server created by Microsoft used by database and server administrators. It aims to provide query axis to text-based data for e.g. log files, XML, CSV etc. Logparser is frequently observed on SF and SO while Logwatch is chiefly present on SF only. mysqlbinlog is used for processing binary log files and usage of this tag is observed in SO and DB [51]. Among all the above mention tools, frequency of usage of Logstash tag is maximum on SF while frequency of Log4j is maximum on SO. Logstash is one of the components of ELT-stack. This ELT-stack combination is widely used by Wikimedia Foundation. Logstash collects all the log-events sent by Wikimedia applications and stores them in an Elasticsearch cluster followed by use of font-end client Kibana in order to filter and display messages [52].

RQ7 conclusions: Syslog-based model Rsyslog is more popular on SO and SF than Syslogd-based model syslogng. System programmers tends to use transaction and binary logging more than error and event logging. This may be attributed to the fact that error and event logging is not observed in SF while transaction and binary logging is observed. DB Administrators tends to use transaction logging and binary logging frequently in order to maintain consistency of their database. Among all the websites observed tag frequency of transaction and binary logging is highest for DB. Among all the logging tools and libraries namely Logparser, Logwatch, NXLog, GrayLog, Logstash, Hugo, Log4j, the popularity of Logstash is maximum on SF while popularity of Log4j is maximum on SO in terms of tag-frequency.

4. Threats to validity

In this section, we discuss various threats to validity related to the results presented in this work.

Threats to external validity: In our study, we conduct experiments on 82K logging questions from six different Q & A websites of the stack exchange network. These websites are subject to a general audience (e.g. SO) and specific audiences (e.g. AE, SF, DB, SU, and SE). Hence, we have depicted the results of source code logging analysis separately for each domain. Logging-related results can be generalized within a domain but may not be generalized across domains. However, results of SO provides a basic level of generalization considering its vast audience across multiple domains.

Threats to internal validity: For topic generation using LDA, we have used $K = 50$ for all six websites. However, this is done irrespective of the size of the corpus of each website. Further, all the code snippets were removed from the analysis using regular expressions along with HTML tags. We notice that previous studies analyzing content from StackExchange websites have also removed source code present in the description of posts [16]. We further minimize the threats of internal validity by using built-in Python libraries (for example, Sklearn, NLTK) for doing data processing.

Threats to construct validity: It is concerned with the identification of logging-related tags that formed the basis of our study and further interpretation of topics. Threats to construct validity is categorized into 3 main parts: **the construction of programming language-related tags**, **construction of general logging-related tags** and **Interpretation of LDA-topics**. First, There are several programming languages. However, we selected 6 programming language through ostensible randomization. Additionally, we use term programming languages for programming languages as well as scripting languages (JS). In order to determine the number of logging-related questions corresponding to programming languages, we have used various logging libraries, tools, APIs etc. specific to that programming languages. Table 7 depicts these logging-related libraries, tools, APIs etc. corresponding to six-programming languages. These libraries are selected after rigorous manual exploration of the internet and existing research by best of our knowledge but due to several programming-language related libraries in the field, there may exist some libraries left unexplored. Hence, C++ and Java have more logging-related questions than Python and C may be because some of Python or C related logging libraries could be left unexplored. Moreover, Boost is a set of libraries written in C++ and aims to provide support for a multitude of tasks, for example, algebra, unit testing, multithreading etc. Thus, Boost tag consists of a set of logging as well as non-logging libraries for C++ which can affect the results of the actual number of logging questions concerning C++. Second, Logging-related Tags used in our analysis are depicted in Table 7. Mostly, these tags are the comprehensive collection of logging-related terms (transaction-log, log-files etc.) and logging-related libraries, Tools, and APIs (SL4J, Logstash etc.). Some of these tags are used by developers in more than one context, for example, Observed results of Lynx and Hugo can be inconsistent as Lynx is a logging library as well as a text-mode web browser while Hugo is also a logging library but also a static site generator written in Go. Third, Interpretation of topics generated from LDA is not an easy task [53] and can be subjective. Thus, first, second author and corresponding author understood the topics and derived the topic labels and other authors verified them. In the cases where topics were hard to interpret, we further studied the questions related to them in order to drive a label.

5. Conclusions and future work

Logging is an important software development practice. Log statements present in the source code are used to record important runtime information. Software practitioners can use this information at the time of debugging. In past, several research studies have been conducted that propose solutions to help software developers in source code logging. These solutions are helpful but at present there is no study that analyzes the issues that software developers face while logging. In this paper, we perform a three dimensional, empirical study of logging questions asked on the six popular Q & A websites. We perform statistical, programming language and content analysis of logging questions. Our analysis helped us to gain insight about the logging discussion happening in six different domains of the Stack-Exchange websites.

Our analysis provides an insight about the logging needs of software developers. Results of our in-depth empirical study show that logging questions are pervasive in all the Q & A websites. The mean time to get accepted answer for logging questions on SU and SF websites are much higher as compared to other websites. It also shows that a large number of logging question invite a great amount of discussion in the SoftwareEngineering Q & A website. We have found that software developers face most of the logging issues in C++ and Java. It shows that the trend of number of logging questions is increasing for Java, Python, and JavaScript, whereas, it is decreasing or constant for C, C++, C#. Researchers can use these results to fine tune the automated logging tools proposed by them. Companies can use these results to fine-tune their tools and to decide which technique to support.

Our analysis also shows that different websites have different dominant programming language. For the SO website C++ and Java are dominant language whereas for the SF and SU website, ‘C’ is the dominant programming language. Researchers can use this information, for example, if they are providing automated logging tool for server they can target it with ‘C’ language, whereas, if they are making general purpose logging tool, they can target it with ‘C++’ or ‘Java’.

Since, this is the first study of on logging issues of Q & A websites, in this we explored different dimensions of logging question in future, we will explore more specific logging problems faced by software practitioners. We plan to extend this work in several dimensions. First, at present we have performed topic analysis using question title and description only. In future, we plan to perform topic analysis of *answers* as well. Second, we plan to perform sentiment analysis of comments associated with logging questions. To find the overall sentiment of users about logging. Third, we plan to perform topic analysis for small time intervals like 1–3 months in order to to find how topic related to logging are changing over the period of time. Fourth, we will perform analysis of most popular logging questions irrespective of the website on which they are asked.

6. Acknowledgment

We thank our advisor and mentor Dr. Ashish Sureka for providing guidance and inputs in this project.

References

[1] Q. Fu, J.G. Lou, Y. Wang, and J. Li, "Execution anomaly detection in distributed systems through unstructured log analysis," in *Proceedings of the Ninth IEEE International Conference on Data Mining*, ICDM '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 149–158.

[2] K. Nagaraj, C. Killian, and J. Neville, "Structured comparative analysis of systems logs to diagnose performance problems," in *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation*, NSDI'12, 2012, pp. 26–26.

[3] S. Lal and A. Sureka, "LogOpt: Static feature extraction from source code for automated catch block logging prediction," in *Proceedings of the 9th India Software Engineering Conference (ISEC)*, 2016, pp. 151–155.

[4] S. Lal, N. Sardana, and A. Sureka, "LogOptPlus: Learning to optimize logging in catch and if programming constructs," in *Proceedings of the IEEE 40th Annual Computer Software and Applications Conference (COMPSAC)*, Vol. 1, June 2016, pp. 215–220.

[5] H. Li, W. Shang, and A.E. Hassan, "Which log level should developers choose for a new logging statement?" *Empirical Software Engineering*, Vol. 22, No. 4, 2017, pp. 1684–1716.

[6] S. Kabinna, C.P. Bezemer, W. Shang, and A.E. Hassan, "Logging library migrations: A case study for the Apache Software Foundation projects," in *Proceedings of the 13th International Conference on Mining Software Repositories*, MSR '16. New York, NY, USA: ACM, 2016, pp. 154–164.

[7] S. Lal, N. Sardana, and A. Sureka, "Improving logging prediction on imbalanced datasets: A case study on open source java projects," *International Journal of Open Source Software and Processes (IJOSSP)*, Vol. 7, No. 2, 2016, pp. 43–71.

[8] StackExchange Community, *StackOverflow home page.* [Online]. https://stackoverflow.com/ [accessed: 26.12.2017].

[9] StackExchange Community, *Serverfualt stack exchange home.* [Online]. https://serverfault.com/ [accessed: 26.12.2017].

[10] StackExchange Community, *Superuser Stack Exchange home page.* [Online]. https://superuser.com/ [accessed: 26.12.2017].

[11] StackExchange Community, *Database Administrators Stack Exchange home page.* [Online]. https://dba.stackexchange.com/ [accessed: 26.12.2017].

[12] StackExchange Community, *Android Enthusiasts home page.* [Online]. https://android.stackexchange.com/ [accessed: 26.12.2017].

[13] StackExchange Community, *SoftwareEngineering home page.* [Online]. https://softwareengineering.stackexchange.com/ [accessed: 26.12.2017].

[14] G. Pinto, F. Castor, and Y.D. Liu, "Mining questions about software energy consumption," in *Proceedings of the 11th Working Conference on Mining Software Repositories.* ACM, 2014, pp. 22–31.

[15] M. Linares-Vásquez, B. Dit, and D. Poshyvanyk, "An exploratory analysis of mobile development issues using Stack Overflow," in *Proceedings of the 10th Working Conference on Mining Software Repositories.* IEEE Press, 2013, pp. 93–96.

[16] A. Barua, S.W. Thomas, and A.E. Hassan, "What are developers talking about? an analysis of topics and trends in Stack Overflow," *Empirical Software Engineering*, Vol. 19, No. 3, 2014, pp. 619–654.

[17] B. Chen and Z.M.J. Jiang, "Characterizing logging practices in Java-based open source software projects – A replication study in Apache Software Foundation," *Empirical Software Engineering*, Vol. 22, No. 1, 2017, pp. 330–374.

[18] Q. Fu, J. Zhu, W. Hu, J.G. Lou, R. Ding, Q. Lin, D. Zhang, and T. Xie, "Where do developers log? An empirical study on logging practices in industry," in *Companion Proceedings of the 36th International Conference on Software Engineering*, ICSE Companion, 2014, pp. 24–33.

[19] S. Lal, N. Sardana, and A. Sureka, "Two level empirical study of logging statements in open

source Java projects," *International Journal of Open Source Software and Processes (IJOSSP)*, Vol. 6, No. 1, 2015, pp. 49–73.

[20] W. Shang, M. Nagappan, and A.E. Hassan, "Studying the relationship between logging characteristics and the code quality of platform software," *Empirical Software Engineering*, Vol. 20, No. 1, 2015, pp. 1–27.

[21] D. Yuan, S. Park, and Y. Zhou, "Characterizing logging practices in open-source software," in *Proceedings of the 34th International Conference on Software Engineering*, (ICSE), 2012, pp. 102–112.

[22] H. Li, W. Shang, Y. Zou, and A.E. Hassan, "Towards just-in-time suggestions for log changes," *Empirical Software Engineering*, Vol. 22, No. 4, 2017, pp. 1831–1865.

[23] D. Yuan, S. Park, P. Huang, Y. Liu, M.M. Lee, X. Tang, Y. Zhou, and S. Savage, "Be conservative: Enhancing failure diagnosis with proactive logging," in *Proceedings of the 10th USENIX Conference on Operating Systems Design and Implementation (OSDI)*, 2012, pp. 293–306. [Online]. http://dl.acm.org/citation.cfm?id=2387880.2387909

[24] J. Zhu, P. He, Q. Fu, H. Zhang, M. Lyu, and D. Zhang, "Learning to log: Helping developers make informed logging decisions," in *Proceedings of the IEEE/ACM 37th IEEE International Conference on Software Engineering (ICSE)*, Vol. 1, May 2015, pp. 415–425.

[25] S. Kabinna, C.P. Bezemer, W. Shang, and A.E. Hassan, "Examining the stability of logging statements," in *Proceedings of the 23rd IEEE International Conference on Software Analysis, Evolution, and Reengineering (SANER)*, 2016, pp. 326–337.

[26] S. Lal, N. Sardana, and A. Sureka, "ECLogger: Cross-project catch-block logging prediction using ensemble of classifiers," *e-Informatica Software Engineering Journal*, Vol. 11, No. 1, 2017, pp. 9–40.

[27] S. Beyer and M. Pinzger, "A manual categorization of android app development issues on Stack Overflow," in *Software Maintenance and Evolution (ICSME), 2014 IEEE International Conference on*. IEEE, 2014, pp. 531–535.

[28] X.L. Yang, D. Lo, X. Xia, Z.Y. Wan, and J.L. Sun, "What security questions do developers ask? A large-scale study of Stack Overflow posts," *Journal of Computer Science and Technology*, Vol. 31, No. 5, 2016, pp. 910–924.

[29] H. Malik, P. Zhao, and M. Godfrey, "Going green: An exploratory analysis of energy-related questions," in *Proceedings of the 12th Working Conference on Mining Software Repositories*. IEEE Press, 2015, pp. 418–421.

[30] C. Nagy and A. Cleve, "Mining Stack Overflow for discovering error patterns in SQL queries," in *Software Maintenance and Evolution (ICSME)*. IEEE, 2015, pp. 516–520.

[31] StackExchange Community, *StackExchange*. [Online]. https://stackexchange.com/ [accessed: 26.12.2017].

[32] Quora Community, *Quora Home Page*. [Online]. https://www.quora.com/ [accessed: 26.12.2017].

[33] StackExchange Community, *What does it mean when an answer is "accepted"*. [Online]. https://stackoverflow.com/help/accepted-answer [accessed: 26.12.2017].

[34] Python Community, *Latent Dirichlet Allocation (LDA) in Python*. [Online]. https://radimrehurek.com/gensim/models/ldamodel.html [accessed: 9.04.2018].

[35] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*, 3rd ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011.

[36] Neurobs, *Neurobs*. [Online]. https://www.neurobs.com/pres_docs/html/03_presentation/07_data_reporting/01_logfiles/index.html [accessed: 12.03.2018].

[37] Wikipedia, *4th Dimension (software)*. [Online]. https://en.wikipedia.org/wiki/4th_Dimension_(software) [accessed: 12.03.2018].

[38] PostgreSQL, *Warm Standby Servers for High Availability*. [Online]. http://www.postgresql.org/docs/8.2/static/warm-standby.html [accessed: 12.03.2018].

[39] MySQL Community, *Reference Manual on Configuring Replication*. [Online]. https://dev.mysql.com/doc/refman/5.7/en/replication-configuration.html [accessed: 12.03.2018].

[40] Network Working Group, *The Syslog Protocol*. [Online]. https://tools.ietf.org/html/rfc5424 [accessed: 12.03.2018].

[41] Rsyslog Community, *Rsyslog*. [Online]. https://www.rsyslog.com/ [accessed: 12.03.2018].

[42] Syslog-ng Community, *Reliable, scalable, secure central log management*. [Online]. https://syslog-ng.com/ [accessed: 12.03.2018].

[43] Python Community, *syslogd – Linux man page*. [Online]. https://linux.die.net/man/8/syslogd [accessed: 12.03.2018].

[44] Techopedia, *Error Log*. [Online]. https://www.techopedia.com/definition/26306/error-log [accessed: 12.03.2018].

[45] T.A. Peters, "The history and development of transaction log analysis," *Library Hi Tech*, Vol. 11, No. 2, 1993, pp. 41–66.

[46] MariaDB Community, *Binary Log.* [Online]. https://mariadb.com/kb/en/library/binary-log/ [accessed: 12.03.2018].

[47] StackOverflow Community, *Graylog.* [Online]. https://stackoverflow.com/tags/graylog/info [accessed: 3.05.2018].

[48] StackOverflow Community, *NXLOG.* [Online]. https://stackoverflow.com/tags/nxlog/info [accessed: 3.05.2018].

[49] archlinux, *Logwatch.* [Online]. https://wiki.archlinux.org/index.php/Logwatch [accessed: 3.05.2018].

[50] Bjorn, F. Crawford, J. Pyeron, J. Soref, K. Bauer, M. Tremaine, O. Poplawski, and S. Jakobs, *Logwatch.* [Online]. https://sourceforge.net/p/logwatch/wiki/Home/ [accessed: 12.03.2018].

[51] MySQL Community, *mysqlbinlog – Utliity for Processing Binary Log Files.* [Online]. https://logging.apache.org/log4j/2.x/ [accessed: 12.03.2018].

[52] Wikitech, *Logstash – Wikitech.* [Online]. https://wikitech.wikimedia.org/wiki/Logstash [accessed: 12.03.2018].

[53] A. Hindle, C. Bird, T. Zimmermann, and N. Nagappan, "Do topics make sense to managers and developers?" *Empirical Software Engineering*, Vol. 20, No. 2, 2015, pp. 479–515.