Logging Practices in Software Engineering: A Systematic Mapping Study

Shenghui Gu, Guoping Rong, He Zhang, and Haifeng Shen

Abstract—*Background:* Logging practices provide the ability to record valuable runtime information of software systems to support operations tasks such as service monitoring and troubleshooting. However, current logging practices face common challenges. On the one hand, although the importance of logging practices has been broadly recognized, most of them are still conducted in an arbitrary or ad-hoc manner, ending up with questionable or inadequate support to perform these tasks. On the other hand, considerable research effort has been carried out on logging practices, however, few of the proposed techniques or methods have been widely adopted in industry. *Objective:* This study aims to establish a comprehensive understanding of the research state of logging practices, with a focus on unveiling possible problems and gaps which further shed light on the potential future research directions. *Method:* We carried out a systematic mapping study on logging practices with 56 primary studies. *Results:* This study provides a holistic report of the existing research on logging practices by systematically synthesizing and analyzing the focus and inter-relationship of the existing research in terms of issues, research topics and solution approaches. Using *3W1H*—***W**hy to log*, ***W**here to log*, ***W**hat to log* and ***H**ow well is the logging*—as the categorization standard, we find that: (1) the best known issues in logging practices have been repeatedly investigated; (2) the issues are often studied separately without considering their intricate relationships; (3) the *Where and What* questions have attracted the majority of research attention while little research effort has been made on the *Why* and *How well* questions; and (4) the relationships between issues, research topics, and approaches regarding logging practices appear many-to-many, which indicates a lack of profound understanding of the issues in practice and how they should be appropriately tackled. *Conclusions:* This study indicates a need to advance the state of research on logging practices. For example, more research effort should be invested on *why to log* to set the anchor of logging practices as well as on *how well is the logging* to close the loop. In addition, a holistic process perspective should be taken into account in both the research and the adoption related to logging practices.

Index Terms—Logging practices, Log, Systematic mapping, Empirical study

1 Introduction

Recently, along with the popularity of DevOps, continuity has been widely recognized as an inherent requirement for many on-line software services [1]. This situation makes it more prominent than ever to monitor software services and address anomalies in a timely manner. Meanwhile, modern software systems are getting larger and more distributed, making it more challenging to detect anomalies and locate the corresponding root causes particularly in intricate software systems. Information such as runtime behavior of software systems is thus believed critical by many researchers and practitioners in detecting and addressing anomalies. Logging is commonly used to capture such runtime behavior information. According to Zhao et al.'s [Zhao 17] and Yuan et al.'s [Yuan 12b] studies, the information contained in logs is often the only source available for software engineers to troubleshoot and diagnose software failures in a production environment.

As an intensively investigated research topic, there are many similar terms describing key concepts around logging. To avoid confusion, we summarize common concepts in Table 1, which will be consistently used throughout this paper. Apparently, the information contained in logs is determined by the log statements that developers insert in source code, which will generate logs when being triggered. Figure 1 depicts relevant logging entities. The upper part of this figure is an example of a log statement, which normally occurs in the source code as necessary. After it has been executed under certain conditions, the log statement will generate a 'Log', as shown by the lower part of Fig. 1. A 'Log' usually contains a large number of entries, aka 'Log messages'. In general, 'Log level', 'Log content', and a suitable place to put a log statement in are usually decided by developers during software development. One focus of the current research is the *2-W* questions, i.e. the context of a logging-prone code snippet (*where to log?*) and the content or level of a log statement (*what to log?*) [2], [Chen 20], [Chen 17a].

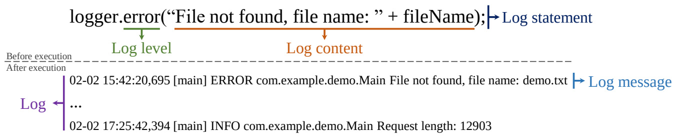
```
logger.error("File not found, file name: " + fileName);
```
```
02-02 15:42:20,695 [main] ERROR com.example.demo.Main File not found, file name: demo.txt
...
02-02 17:25:42,394 [main] INFO com.example.demo.Main Request length: 12903
```

Fig. 1. Logging related terms in an example.

• Shenghui Gu, Guoping Rong, and He Zhang are with the State Key Laboratory for Novel Software Technology, Nanjing University, China.
E-mail: shenghui.gu@smail.nju.edu.cn, {ronggp, hezhang}@nju.edu.cn
• Haifeng Shen is with the HilstLab, Peter Faber Business School, Australian Catholic University, Sydney, Australia.
E-mail: haifeng.shen@acu.edu.au

1.1 Problem description and research motivation

Although the importance of logging practices has been broadly recognized in industry [Kabi 16b], [Pecc 15], [Yuan 12b], [3], such practices are still far from satisfac-

TABLE 1
Logging related terms.

Term	Description
Log statement	A log statement is a statement placed in the source code that outputs a record of the behavior of the specified program during execution.
Log message	A log message is the information generated by the program at runtime based on a log statement.
Log	A log is a collection of the execution outputs of log statements, usually stored in a text file or a database.
Log level	Log levels reflect the verbosity or severity of the log messages, and log messages are usually clustered for analysis based on these log levels.
Log content	Log content refers to the information carried by the log statement, including static text and variables.
Log location	A log location is the place where a log statement is inserted.
Log placement	Log placement is a strategic design and implementation of the *2-W* questions for all log statements in the source code of a system or component.
Logging	Logging is a developer's action of inserting a log statement to a certain code snippet. Apparently, the developer needs to decide the *2-W* when conducting logging.
Logging practice	A logging practice refers to any practice pertinent to the insertion or revision of log statements for a system or component, which is also the research object in this study.

tory [Liu 20]. As a matter of fact, current logging practices are basically carried out manually and heavily rely on the expertise, experience and preference of software engineers [Pecc 15], [Yuan 12b], [Yuan 12c]. Typically, software engineers need to consider the *2-W* questions during coding, easily leading to either insufficient or excessive logging. For example, studies carried out both in industry and academia revealed several typical issues in current logging practices, including low and divergent density of log statements [Rong 18], [4], misuse of log levels [Rong 18], [Yuan 12b], and lack of necessary information-capturing variables [Yuan 12a], [4]. These issues may easily lead to questionable realization of the primary purpose to carry out logging practices [Li 20a], [Rong 20], i.e. capturing the runtime behavior of software systems as intended. Meanwhile, although relatively extensive research has been done to improve logging practices, it seems that the industry has not benefited from these research outcomes. For instance, optimizing log location and enhancing log content have been well proposed in the literature to improve logging practices, however, they are rarely adopted by the most widely used logging tools in daily development tasks [Fu 14], [Li 18b], [Yuan 12b], [Zhu 15].

To uncover possible reasons behind this phenomenon, it is important to establish a holistic view on the research status of logging practices, including the challenges in logging practices and the proposed solutions, and more importantly the possible problems and gaps in the existing research which may further shed light on the potential future research directions. Only a handful of studies have been conducted [2], [5], [6], [7] to review the challenges and the solutions available. However, the issues in the existing research are rarely analyzed and discussed. To this end, this work is motivated by the need to depict a landscape of logging practices, identify gaps between challenges and solutions, and suggest potential remedies and future research directions to close the loop through a systematic mapping study (SMS). To be specific, we describe the goal of this study using a Goal-Question-Metric (GQM) [8] style as follows:

By aggregating and synthesizing *the existing issues, research topics, and proposed approaches related to logging practices*

For the purpose of *systematically establishing an insight to the state-of-the-art in the area of logging practices, revealing omissions/problems and hence indicating potential research directions*

From the perspective of *software engineering researchers and practitioners*

In the context of *logging practices in the development and maintenance of regular business software systems.*

The description of the research goal implies the focus of this study which is elaborated in detail in Section 2.2. In short, we are only concerned with software engineers' logging practices when instrumenting log statements in regular business software systems. As a result, log analysis or building logging framework/infrastructure is beyond the scope of this work.

1.2 Research approach and contribution

A systematic mapping study (SMS) provides an overview of a research area through classification and counting contributions in relation to the categories of that classification [9]. It is a suitable approach to addressing the research goal presented in the above subsection. By following the guidelines [9], [10], [11], [12], we examined 56 primary studies published in the mainstream Software Engineering (SE) venues with a focus on logging practices. As the result, we are able to provide a holistic view of the existing research, clarify the issues in logging practices and summarize relevant solutions or approaches. More importantly, through cross analysis, we are able to identify gaps and derive potential follow-up research directions that may bring significant benefits to the whole community that heavily relies on logging in their software development and maintenance tasks. The main contributions of this paper can be summarized as follows:

- It presents a comprehensive understanding on the state-of-the-art research on logging practices using *3W1H*—***W****hy to log*, ***W****here to log*, ***W****hat to log* and ***H****ow well is the logging*—as the categorization standard.
- It discovers that the current research focus regarding logging practices has been on the *2-W* questions. In other words, the majority of issues, research topics and solution approaches fall in the categories of ***W****here to log* and ***W****hat to log*, in contrast, little research effort has been made on ***W****hy to log* and ***H****ow well is the logging*.
- It unveils possible problems in the existing research through cross analysis, e.g., "lack of research to address critical issues", "unrealistic expectation of general yet adaptable solutions" and "separated research within logging practices and between logging practices and log analysis".
- It suggests several potential future research directions, for example, considering a process perspective to logging practices and recognizing the anchor value of logging intentions and concerns (*I&C*).

1.3 Organization

The rest of this paper is organized as follows. Section 2 elaborates our research method. The metadata results are presented in Section 3. In Section 4, we explain the *3W1H* categorization scheme that is applied throughout the whole study. The results and findings to each research question are presented in Section 5. In Section 6, we discuss possible reasons behind the current status quo, and several considerations for the next-step research on logging practices. The threats to validity are discussed in Section 7. To position this work among all relevant empirical studies, we describe the related work in Section 8. Finally, we conclude this paper in Section 9.

2 Research Method

In this section, we elaborate the research questions, the research scope, the roles and responsibilities of participating researchers, and the research tasks, followed by the detailed process for literature search as well as the processes for data extraction, synthesis and analysis.

2.1 Research questions

To address the research goal and shed proper light on the research state of logging practices, four research questions (RQs) are promoted as follows.

RQ1: What major issues regarding logging practices have been identified by existing research?

Researchers dedicated to logging practices are usually at a better position to understand the issues that are targeted in relevant studies. In this sense, this research question aims to aggregate these issues to better understand the major challenges and necessary contextual information regarding logging practices.

RQ2: What major research topics regarding logging practices have been investigated by existing research?

This research question aims to reveal the hot spots in logging practices that researchers have been investigating. It attempts to portray the landscape of major research topics around logging practices.

RQ3: What solutions/approaches are proposed in existing research?

This research question aims to summarize and classify the solutions or approaches proposed by existing studies, through which we may be able to establish a fair understanding of the research progress towards addressing the issues identified in RQ1 and the topics identified in RQ2.

RQ4: What gaps exist between the identified issues regarding logging practices and the research efforts in tackling these issues?

This research question aims to reveal the extent to which existing research is able to address the issues in logging practices and pinpoint possible gaps between the issues/topics identified in RQ1/RQ2 and the solutions/approaches proposed in RQ3.

2.2 Research scope

It is important to clearly define the scope of a study so that only relevant primary studies are reviewed to answer the research questions.

First of all, the scope of this study is limited to the *starting point* of logging, i.e. the practices used to generate logs during software development and maintenance, which is similar to that of Chen's survey paper [2], i.e. the *Log Instrumentation Phase*. Therefore, research on log analysis, which is the *ending point* of logging and concentrates on utilizing the information in logs to achieve development or management objectives such as detecting bugs, optimizing system performance or performing recommendations, is out of the scope of this study. That said, it is worth noting that from a process perspective, knowing the information needs in log analysis may have a positive impact on improving the practices for log generation. However, the process perspective has not been established in existing research and we have found that the work reporting the impact of log analysis on logging practices is generally scarce. Therefore, it makes sense to exclude log analysis in the scope of this study.

Second, we only focus on the logging practices that assist in generating logs used to diagnose errors or failures contained in the software systems therein. Therefore, those logging practices generating logs of end user behavior analysis for business needs, e.g., audit logs [13] and security logs [14], are beyond the scope of this study.

In a nutshell, the scope of this study is thus constrained to the studies that investigate how an appropriate log statement is placed in a suitable place in source code to effectively generate logs for capturing runtime behavior so as to support defect detection in the software system. See Appendix B for more detail and examples.

2.3 Roles and responsibilities

In addition to the listed authors, this study also involved eight master students from a seminar series over a span of 5 months. All the participants met weekly or as needed for publication screening, data extraction, consolidation, and discussion.

In detail, one author (the supervisor) took charge of the discussion on research questions, research methodology, and data extraction schema at the time of designing the SMS protocol.

When it came to literature screening and data extraction, at least two students handled each paper independently and the supervisor cross-checked all the work accomplished by the students. In the process of data synthesis, one doctoral student of the authors led the whole process with the participation of the eight master students, and the supervisor also cross-checked their outcomes.

To reach consensuses and avoid potential biases, we established several mechanisms as follows.

1) We carried out several trial runs in the seminar series to train these students for mastering data extraction and synthesis methods. The students are required to present and discuss their findings at the end of each trial run.
2) All participants needed to vote for each finding with the options of 'agree', 'object', and 'not object'. The option of 'not object' is a neutral opinion, meaning one does not fully agree on a finding but can accept it. A consensus requires at least one 'agree' and no 'object' at the same time.

3) If a consensus is not reached on a finding, a discussion meeting is organized to sort out the discrepancies.

2.4 Research tasks

To answer the four research questions, three research tasks were conducted: the 'setup' task is to set up the literature review; the 'aggregation&synthesis' task is dedicated to collecting evidence to answer research questions RQ1, RQ2 and RQ3; the 'cross analysis' task is conducted to synthesize the information to answer RQ4. The tasks were conducted in a sequential manner and interconnected through a number of artifacts generated by their subtasks. The overall research process is illustrated in Fig. 2 and detailed as below.

The 'setup' task includes definition of the research questions and the review protocol, selection of search venues and databases, definition and revision of the search string, and identification of the primary studies. The detailed steps in this task is presented in Section 2.5. The 'aggregation&synthesis' task was conducted to extract the data from the primary studies in terms of the research questions, and then to map the data into four categories, i.e. *Why to log? Where to log? What to log? and How well is the logging?* (cf. Section 4). Further, we used an iterative coding process to identify the main categories of the extracted data for each research question, which is detailed in Section 2.7. In the 'cross analysis' task, we synthesized the review results and performed cross analysis so as to further reveal several implications to answer RQ4, which is presented in Section 5.4.

2.5 Search process

A strict and repeatable search process is one important characteristic of an SMS. We applied several strategies to boost this characteristic in the search process of our study. In general, an explicitly defined search process is required, as illustrated in the 'setup' task in Fig. 2. First, we conducted a manual search from the premier SE conferences and journals, and retrieved 11 papers. Based on these papers, a search string was formed, revised and used for automated search in the selected digital libraries. At that point, 7,669 papers were retrieved. By applying the predefined inclusion/exclusion criteria (as shown in Table 2), these papers were then quickly screened manually, reducing to 109 papers. After reading the full-text of these 109 papers for further screening, we finally ended up with 54 papers. As the last step, a complementary snowballing search process was performed to obtain potentially relevant studies so as to pursue a comprehensive set of relevant studies. From this step, we obtained two more papers, making a final set of 56 papers selected as the primary studies for this work.

2.5.1 Manual search

The manual search was carried out on the following major software engineering conferences and journals:

- International Conference on Software Engineering (ICSE)
- IEEE Transactions on Software Engineering (TSE)
- Empirical Software Engineering (EMSE)
- IEEE International Conference on Software Maintenance and Evolution (ICSME)
- IEEE Working Conference on Mining Software Repositories (MSR)
- Journal of Systems and Software (JSS)

The purpose of manual search was not to cover all potential sources, but only a portion that contains the most high-quality studies relevant to our RQs according to our initial knowledge. The publication time span was constrained from 2000 to 2020 for the reason that the number of papers related to logging practices is quite small before 2000, according to our previous study [5]. As the result, we gathered 11 papers on logging practices from the manual search.

2.5.2 Search string

We developed the search string using the following steps in the light of the guidelines proposed in [15], [16].

Step 1: We first derived major search terms from our RQs and the keywords in the relevant publications we obtained from manual search.

Step 2: We then combined the identified search terms into a set of candidate search strings.

Step 3: At last, we performed several pilot searches to improve and determine the most suitable search string according to the Quasi-Gold Standard (QGS) based systematic search approach [17]. With reference to our previous experience [5], this time we keep 90.9% and 0.13% for *sensitivity* and *precision* respectively, so as to include as much literature as possible.

As the result, we have the search string as below.

(logging OR log OR logs) AND (practice OR practices OR statement OR statements OR construct OR constructs OR format OR formats OR code)

Note that the search string needs to be coded according to different digital libraries' search syntax.

2.5.3 Automated search

The automated search was performed in the following digital libraries.

- ACM Digital Library[1]
- IEEE Xplore[2]
- ScienceDirect[3]
- Scopus[4]
- Wiley Online Library[5]

With reference to many existing systematic mapping and review studies, as well as the statistics of the literature search engines [17], these selected digital libraries are believed to provide a wide coverage of relevant primary studies.

2.5.4 Study identification

Three steps are performed to identify and determine the set of primary studies for final analysis, including *quick scanning* to filter out irrelevant studies, *full text reading* to assure quality of the relevant studies, and *snowballing* to ensure that the chance of missing important relevant studies is minimized as much as possible. We detail these steps in the following paragraphs.

1. https://dl.acm.org/
2. https://ieeexplore.ieee.org/
3. https://www.sciencedirect.com/
4. https://www.scopus.com/
5. https://onlinelibrary.wiley.com/

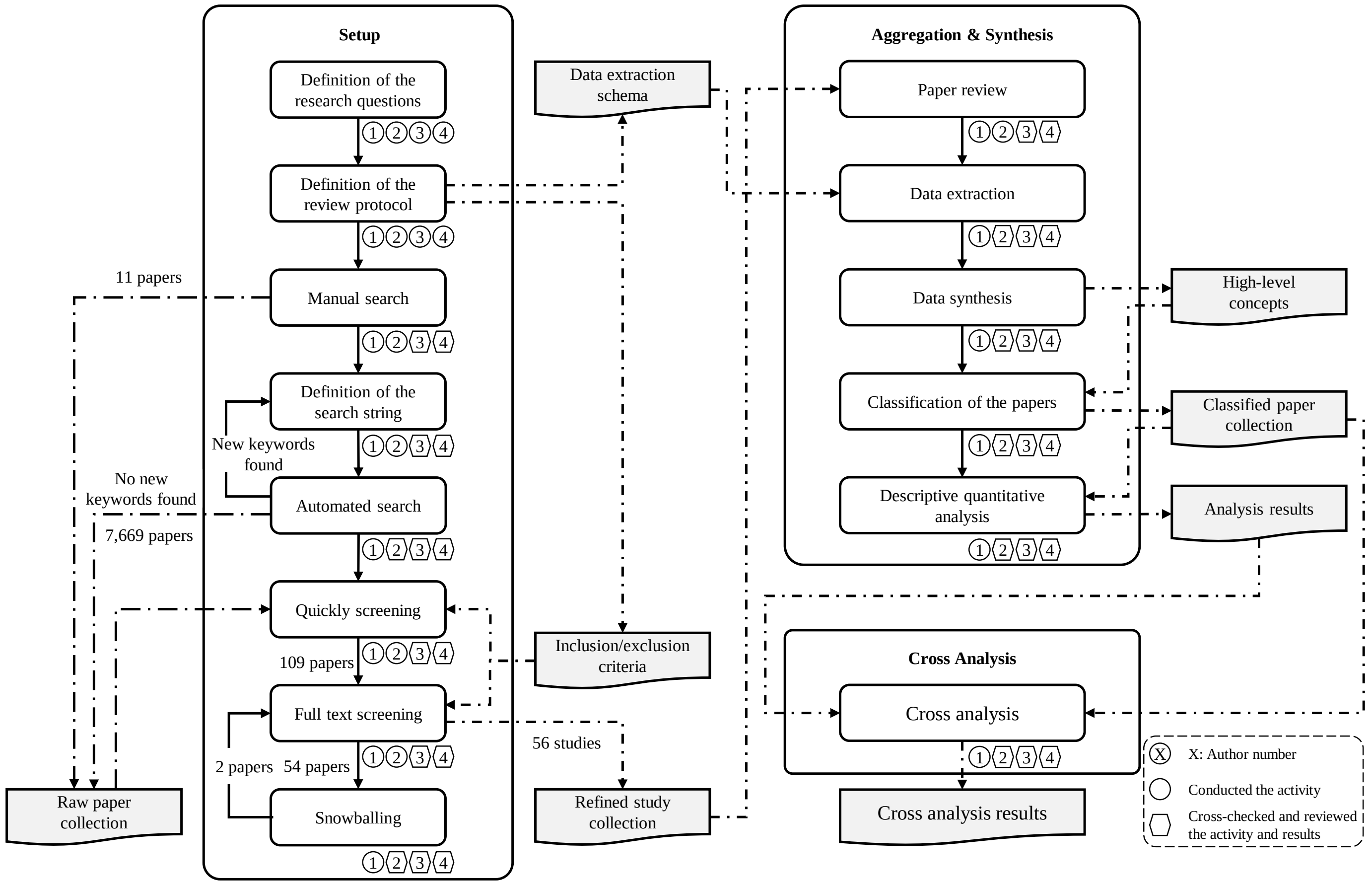

Fig. 2. Research process.

TABLE 2
Criteria to include/exclude a study.

	Inclusion criteria
I1	Studies must be published by conferences, journals, or symposiums[*].
I2	Studies must be written in English.
I3	Studies must be published after 2000.
I4	Studies must be primary research.
I5	Studies must focus on logging practices.
	Exclusion criteria
E1	Studies investigate log analysis or usage of log messages.
E2	Studies investigate techniques for logging user behaviors.
E3	Studies do not explicitly discuss logging practices.
E4	Studies are published by workshops.
E5	Studies are published as position papers or work in progress.
E6	Studies replicate work in previous publications.

[*] Only for those well-known symposiums, e.g., ISSRE, ESEM.

Quick scanning: During this step, a quick screening of the titles and abstracts of the potential studies was performed by following the inclusion/exclusion criteria detailed in Table 2. For those papers that are unable to be excluded by checking title and abstract, we postponed the decision to the following steps. As the result, we had 109 papers after this step.

Full text reading: With the 109 papers, we read their full-text to make further exclusion and then eliminate the irrelevant ones from the previous step. It is worth noting that the same study may have been published in multiple papers, and in this case we only keep the most complete version by reading and comparing the content of all versions of the work in detail. As the result, we retained 54 papers after this step.

Snowballing: We performed a *snowballing* search to further retrieve potential relevant studies, which consists of *backward snowballing*, *forward snowballing*, and *author snowballing*. Detailed steps are listed as the following.

Backward snowballing: Based on the reference lists of the 54 papers derived from the previous steps, we retrieved papers using the criteria from Table 2.

Forward snowballing: Use Google Scholar to identify papers citing the current 54 papers. Meanwhile, the selection criteria were also applied to identify and include an extra relevant paper.

Author snowballing: All publications of each author of the 54 identified papers were further checked against the selection criteria for final possible inclusion.

As the result, two new papers were identified in the *forward snowballing* step.

2.6 Data Extraction

The relevant data was extracted from the 56 selected studies according to a predefined extraction schema, as depicted in Table 3, that covers two major aspects of information, i.e. metadata and specific information, respectively.

The metadata includes information such as author, year, title, and publishing venue, which provides a big picture

TABLE 3
Data extraction schema.

Attribute	Description	RQ
Author	The authors of the publication.	Meta
Title	The title of the publication.	
Year	The published year of the publication.	
Venue type	The venue type of the publication (conference, journal, etc).	
Venue name	The venue name of the publication.	
Logging issue	The issues and problems in current logging practices.	RQ1, RQ4
Research motivation	The motivation of the selected study.	RQ2, RQ4
Research question	The research questions of the selected study.	
Research subject	The subject of the selected research.	
Proposed approach	The approach, method, tool or algorithm proposed by the selected research, as well as their description, pros and cons.	RQ3, RQ4
Research contribution	The main contribution of the selected study.	
Research conclusion	The conclusion of the selected study.	

of the overall research state, for instance, the number of studies each year, the potential trend, and the distribution of publication venues.

Meanwhile, the specific information is derived from the research questions of this study. The relationship between the specific information and research questions is listed in Table 3. Note that answering RQ4 requires a cross analysis of the information used to answer RQ1, RQ2 and RQ3.

2.7 Data synthesis

We applied both quantitative and qualitative methods to synthesize the evidence to answer all the research questions. This subsection elaborates the data synthesis methods we adopted in this research.

Descriptive statistics is the major quantitative method used to describe and summarize data characteristics. To be specific, it was adopted to synthesize the quantitative data to present trend (Fig. 3), distribution (Fig. 4), etc.

Thematic analysis is the major qualitative method applied in our study to identify common themes within data [18]. It was used to understand commonly raised issues, research topics, proposed approaches regarding logging practices. To assist thematic analysis, coding, the process of labeling and organizing qualitative data, was applied to identify and distinguish themes and the relationships between them [19]. Basically, we used *open coding* and *axial coding* to attain high-level concepts from the extracted data for each research question. *Open coding* is conducted to identify the conceptual categories from the original data, where the generated 'code' is designed to retain the exact words as much as possible. *Axial coding* is used to generate a set of new 'codes' by comparing and merging the 'codes' identified in open coding as appropriate. For example, *"useless log messages"* and *"arbitrarily placed log statements likely generate a lot of trivial logs that may be redundant or useless"* were further coded as *"redundant or useless log messages"*.

TABLE 4
Study distribution over electronic libraries.

Electronic library	# Retrieved studies	# Relevant studies
ACM Digital Library	1,757	26
IEEE Xplore	2,472	28
ScienceDirect	983	1
Scopus	548	37
Wiley Online Library	1,909	2

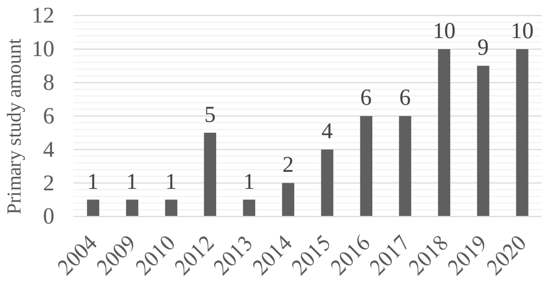

Fig. 3. Study distribution over years.

3 Results

We first summarize the results of metadata in this section to provide a bird's eye view of the studies included in our study.

A full list of the 56 studies is presented in Appendix A. Table 4 describes the distribution of the 56 studies across different electronic libraries. Note that one study may be indexed by multiple digital libraries. Apparently, ACM Digital Library, IEEE Xplore and Scopus contribute the vast majority of studies to our SMS.

The distribution of the publishing time can be illustrated by grouping these studies into years, as shown in Fig. 3. In general, a roughly ascending trend can be observed regarding the number of studies over years, especially in the recent years, indicating an increasing attention among researchers to logging practice.

The publication venues of these 56 studies tend to be diverse. However, we could observe that more than half of

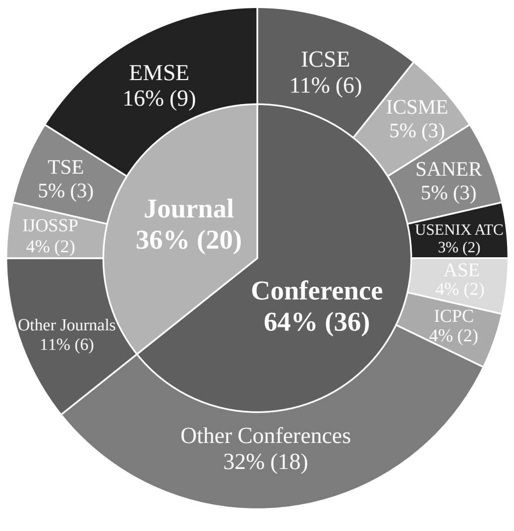

Fig. 4. Study distribution over publication venues.

the selected studies in our SMS are published at premier venues such as TSE, EMSE, and ICSE, which to some extent implies the high quality of the research as well as the intensive interest in the community.

4 3W1H—Essentials about logging

To provide a basis for data synthesis, a taxonomy scheme is necessary to be defined in the beginning. Therefore, we applied the *thematic analysis* elaborated in Section 2.7 together with our previous experience [Rong 20], [5] and current knowledge on logging practice research (e.g., [2], [Li 20a], [Fu 14], [Liu 19], [Kabi 16b]) to categorize the relevant studies. As the scope of this study is limited to investigating developers (***W**ho*) instrumenting log statements during software development (***W**hen*), the ***W**ho* and ***W**hen* categories are fixed and hence removed from the final category set. As elaborated in Section 2.2, the 56 primary studies are categorized into the four themes of *3W1H*: ***W**hy to log?* ***W**here to log?* ***W**hat to log?* and ***H**ow well is the logging?*

- *Why to log?*

This question is related to the original intention to carry out logging practices. Through log statements, most developers' **intention** is to capture and record information relevant to dynamic system behaviors. However, this practice incurs cost/overhead, which triggers **concerns** to most developers as well. To a certain degree, the intrinsic Intentions & Concerns (*I&Cs*) play the role of initiating logging practices. Several researchers have raised the awareness of the importance of *I&Cs* for logging practices. For example, Jia et al. present a model to describe the logging intentions, which is a typical reason to answer *why to log* [Jia 18]. Nevertheless, our previous work on this topic revealed major gaps between the developers' *I&Cs* and the actual log statements in the source code of real-world projects [Rong 20].

- *Where to log?*

This question considers one of the critical aspects regarding logging practices, i.e. to determine the location of log statements in source code. Usually the answer to this question only provides a general guidance on where to place a log statement [Li 20b], [Fu 14]. As one of the most concerned questions by researchers, many studies have been conducted to determine and improve the location of log statements in source code. For example, a high-level strategy on logging practices is achieved by importing information entropy theory into logging practices to optimize the location of log statements [Zhao 17]. Lal et al. conducted a series of studies adopting machine learning to predict log statements in different code snippets [Lal 17], [Lal 16a], [Lal 16c], [Lal 16b], [Lal 19].

- *What to log?*

This question involves with the subtle consideration about concrete content (e.g., what variables should be recorded? what is the suitable verbosity level?) of a log statement. Various methods have been proposed to develop the concrete content of a log statement. For example, some tools have been designed to assist developers in deciding verbosity level of log statements [Anu 19], [Li 17a], while some tools have been designed to determine which variables to log [Liu 19].

- ***H**ow well is the logging?*

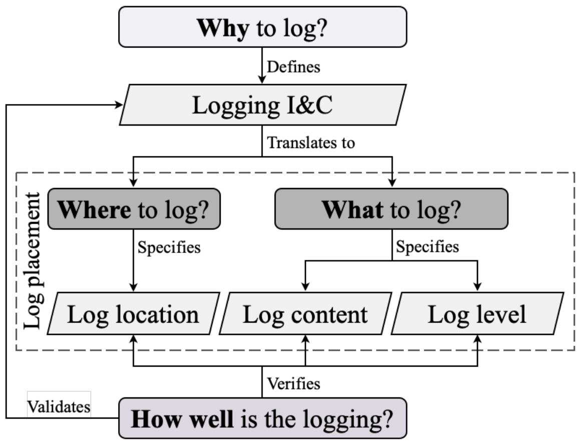

Fig. 5. The relationships between the questions in the *3W1H* categorization.

In general, this question is concerned about the degree to which developers' *I&Cs* can be fulfilled in the implemented source code and evolve over time. In this sense, the *how well* question is only valid on existing log statements and can be considered from two aspects. One is the deviation between the implemented source code and some generally accepted rule-of-thumb guidelines/rules regarding logging practices [20], [21], [22]. Although these guidelines/rules may be followed to convey developers' *I&Cs*, they are not always necessarily able to reflect the true *I&Cs*. The other aspect is the deviation between the implemented source code and the developers' original *I&Cs*. In this sense, the importance of relevant practices is self-evident. Some research in this category focuses on evaluating the effectiveness of existing logging mechanisms in the context of real-world case studies [Cinq 10]. Other research focuses on the revision of log statements after the delivery/release of a software product, for example, to repair faults or to improve performance or other quality attributes. A typical example is the study carried out by Chen et al. [Chen 19]. In this study, researchers extracted the historical issues in log statements and revealed that the log statements had not been well revised when the corresponding business code evolved to meet new business requirements.

The relationships between the *3W1H* questions are illustrated in Fig. 5. In general, the question of *why to log* seeks the developer's *I&Cs*, both implicitly and explicitly, which translate to the *2-W* questions (i.e. *where to log* and *what to log*) in log placement. The question of *where to log* specifies the log location, whereas the *what to log* question specifies both log content and log level. To close the loop, the *how well* question not only verifies whether log placement is properly reflected in actual source code, but more importantly, it also validates that the *I&Cs* are adequately satisfied. With the *3W1H* categorization scheme, we are not only able to properly classify the issues, research topics and solution approaches in current studies but also lay a solid foundation for synthesis according to the inherent relationships across the four questions.

To achieve a consistent understanding, a team discussion is conducted to assign a proper category to a study according to the specific research question and the corresponding

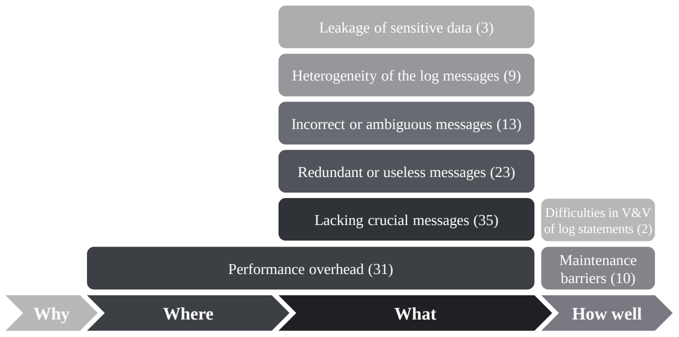
Leakage of sensitive data (3)
Heterogeneity of the log messages (9)
Incorrect or ambiguous messages (13)
Redundant or useless messages (23)
Lacking crucial messages (35)
Difficulties in V&V of log statements (2)
Performance overhead (31)
Maintenance barriers (10)
Why
Where
What
How well

Fig. 6. Relationship between issues and the *3W1H* categorization.

relevant evidence. It is noteworthy that one study may occur in multiple categories simultaneously. For example, if the issue (RQ1) is about the 'performance overhead', then it may cover both *where to log* and *what to log*. Besides, one study may also appear in different categories providing evidence to answer different research questions. For example, a study with the concern on 'performance overhead' may be categorized as *where to log* in RQ1. If the relevant solution pertains to the *I&C*s, it may also appear in the category of *why to log* in RQ3.

5 Synthesis and Findings

In this section, we answer the four RQs using the evidence we collected and synthesized from the review. By adopting the *3W1H* questions (cf. Section 4), we classified all the 56 studies into different categories according to the specific research question and the corresponding relevant evidence. The rest of this section elaborates our major observations and findings.

5.1 Issues in logging practices (RQ1)

We first categorize the issues raised or discovered in the primary studies through a manual coding approach (cf. Section 2.7). Although the issues discovered in these primary studies may contain higher credibility since concrete evidence is provided, we also include the issues directly claimed by researchers with the consideration that researchers focusing on the exact area may better understand the various pains of logging practice. Then from a different perspective, we further categorize the issues into different *3W1H* questions. With these two orthogonal perspectives, we aim to present a big picture about the various pains of logging practice.

In general, we identified 8 types of issues covering different *3W1H* questions except the question of *why to log*. The detailed distribution of issues in logging practices is shown in Table 5 and the relationship between these issues and *3W1H* questions is illustrated in Fig. 6. It is clear that 3/4 of the issues regarding logging practices fall into the *2-W* questions. In contrast, only 1/4 issues are related to the question of *how well is logging*, while no issue has been identified to be relevant to the *why to log* question. This finding implies that certain aspects of logging practice have been neglected in the community. Note that the percentage of each issue denotes the proportion of the primary studies in relation to this issue. Since one study may raise different issues and occur in different categories, the sum of all the percentage may exceed 100%. In the following subsections, we elaborate these issues in detail for each *3W1H* question.

5.1.1 *"What to log" issues*

This category has the most issues related to logging practices.

Lacking crucial messages in log statements turns out to be the most discussed issue, which is reported in 35 (62.5%) studies. In practice, key decisions about logging are normally left to the programming stage, which easily ends up with insufficient logging [Cinq 09]. Several studies [Zhi 19], [Yuan 12b], [Cinq 12], [Rong 18], [Li 18b], [Tova 13], [Hass 18], [Kubo 20], [Cinq 20] directly pointed out or implied this issue or the like. In fact, insufficient logging may be derived from either less than expected log statements [Fu 14], [Ghol 20], [Jia 18], [Lal 16b], [Lal 17], [Lal 15], [Lal 16a], [Lal 16c], [Lal 19], [Li 20b], [Li 17a], [Li 17b], [Luo 18], [Mizo 19], [Sain 16], [Yao 18], [Zeng 19], [Zhao 17] or missing key variables in log statements [Liu 19]. Both create obstacles for log analysis [Yuan 12c], [Li 18a], i.e. lacking crucial messages. In [Yuan 12a], it is confirmed that the majority of failures do not have failure-related log messages. Moreover, Li et al. conducted a qualitative study and confirmed that missing log messages may lead to extra effort in subsequent analysis and even confuse users [Li 20a], which is also confirmed in [Cinq 10].

Redundant or useless messages in log statements become an issue on the opposite end of the spectrum, which has also been pointed by many researchers, 23 (41.1%) studies to be specific. Excessive logging may be one of the reasons for redundant or useless messages [Fu 14]. Meanwhile, with the increasing complexity of software systems and the way to construct and deploy them (e.g., microservice architecture), software systems or services are producing more log messages than before [23], [24], [25]. A large portion of log messages are redundant or useless as mentioned by many studies [Cinq 09], [Zhi 19], [Cinq 12], [Marr 18], [Ding 15], [Li 18b], [Tova 13], [Hass 18], [Fu 14], [Li 17a], in which valuable log messages may be obscured by these 'garbage' noises [Li 18a], [Anu 19], [Lal 17], [Lal 16a], [Lal 16c], [Li 20b], [Liu 20], [Liu 19], [Sain 16], [Shan 14], [Zeng 19], [Zhu 15], [Li 20a], creating big challenges to store and analyze the logs to satisfy the original intention of logging practices, e.g., failure detection, diagnosis and recovery.

Incorrect or ambiguous messages in log statements are another notable issue, which has been mentioned by 13 (23.2%) studies. Several studies report that some events and information captured by the logs may be incorrect or misleading and unable to support further operations [Cinq 09], [Zhi 19], [Cinq 10], [Kim 19], [Pecc 12], [Pecc 15], [Tova 13], [Hass 18], [Chen 17a], [Cinq 12], [He 18], [Li 19]. For example, Cinque et al. stated the presence of misleading log messages may cause inaccurate log analysis, compromising the ability of discriminating events related to actual failures from presumed ones [Cinq 09]. Li et al. also claimed that incorrect log content or levels may confuse developers in debugging [Li 20a].

TABLE 5
Distribution of issues in logging practices.

Category	Issue	Primary studies	Percentage
Where & What	Performance overhead	[Chen 17a], [Chow 18], [Ding 15], [Fu 14], [Ghol 20], [Jia 18], [Kabi 16a], [Lal 16b], [Lal 17], [Lal 15], [Lal 16a], [Lal 16c], [Lal 19], [Li 20b], [Li 17a], [Li 17b], [Li 18a], [Li 20a], [Liu 20], [Liu 19], [Luo 18], [Marr 18], [Mizo 19], [Sain 16], [Shan 14], [Yao 18], [Yuan 12b], [Yuan 12a], [Zeng 19], [Zhao 17], [Zhu 15]	55.4%
What	Lacking crucial messages	[Cinq 09], [Cinq 10], [Cinq 20], [Cinq 12], [Fu 14], [Ghol 20], [Hass 18], [Jia 18], [Kubo 20], [Lal 16b], [Lal 17], [Lal 15], [Lal 16a], [Lal 16c], [Lal 19], [Li 20b], [Li 17a], [Li 17b], [Li 18a], [Li 18b], [Li 20a], [Liu 20], [Liu 19], [Luo 18], [Mizo 19], [Rong 18], [Sain 16], [Tova 13], [Yao 18], [Yuan 12b], [Yuan 12a], [Yuan 12c], [Zeng 19], [Zhao 17], [Zhi 19]	62.5%
	Redundant or useless messages	[Anu 19], [Cinq 09], [Cinq 12], [Ding 15], [Fu 14], [Hass 18], [Lal 17], [Lal 16a], [Lal 16c], [Li 20b], [Li 17a], [Li 18a], [Li 18b], [Li 20a], [Liu 20], [Liu 19], [Marr 18], [Sain 16], [Shan 14], [Tova 13], [Zeng 19], [Zhi 19], [Zhu 15]	41.1%
	Incorrect or ambiguous messages	[Chen 17a], [Cinq 09], [Cinq 10], [Cinq 12], [Hass 18], [He 18], [Kim 19], [Li 19], [Li 20a], [Pecc 15], [Pecc 12], [Tova 13], [Zhi 19]	23.2%
	Heterogeneity of the log messages	[Chen 20], [Cinq 09], [Ghol 20], [He 18], [Liu 20], [Marr 18], [Pecc 15], [Salf 04], [Tova 13]	16.1%
	Leakage of sensitive data	[Li 20a], [Zhi 20], [Zhou 20]	5.4%
How well	Maintenance barriers	[Chen 17b], [Chen 19], [Chen 17a], [Ghol 20], [Li 18b], [Pecc 15], [Yuan 12b], [Li 20a], [Kabi 16b], [Shan 14]	17.9%
	Difficulties in V&V of log statements	[Chen 19], [Rong 20]	3.6%

Heterogeneity of log messages is also an issue attracting researcher's attention, which has been mentioned in 9 (16.1%) studies. Due to the unstructured nature of log content itself and the arbitrariness of logging practices, the format and content of log messages is generally heterogeneous [Liu 20], [He 18], [Pecc 15], [Marr 18], [Ghol 20], [Tova 13], [Salf 04], especially with the increase of system complexity, which naturally involves more developers [Cinq 09]. The format of log messages is usually determined by the logging tools or libraries in use, while the content of log messages is normally determined by developers' intentions or information needs. Both aspects can vary with different developers or projects, leading to heterogeneity of log messages and further influencing the effectiveness of log analysis [Cinq 09].

Leakage of sensitive data is an issue pertaining to information security. Log files may contain sensitive information due to security breaches in logging practices. Zhi et al. identified several common root causes of this issue [Zhi 20], e.g., insecure whole-object logging in which developers make logging calls with direct reference to composite objects and incorrect permission assignment that leads to leak of sensitive information in log files. Although it is only mentioned in three studies [Zhou 20], [Li 20a], [Zhi 20] (5.4% of the overall studies), this type of privacy and security vulnerabilities can not be neglected.

5.1.2 "How well is logging" issues

We identified two issues related to *how well is logging*.

Maintenance barriers is a major issue in this category, which has been reported in 10 (17.9%) studies. As mentioned in [Kabi 16b], logs are often unstable, e.g., the log statements often change without considering other stakeholders, influencing the subsequent analysis and increasing the maintenance cost. Some of the modifications may even introduce errors into the target systems [Shan 14] and approximately one third of modifications of log statements are after-thoughts when there is no proper log statement in the first place [Yuan 12b]. Moreover, log statements often co-evolve with bug fixes or feature updates, making it even more challenging to maintain them in frequently evolving systems [Li 18b], [Chen 17a], [Chen 17b], [Chen 19], [Ghol 20], [Yuan 12b], [Pecc 15]. A notable point is that too many log statements inside the business code may also decrease code readability and quality [Li 20a], [Chen 17a], thereby further hindering the maintainability of both log statements and business code, which needs extra efforts [Li 20a].

Difficulties in V&V of log statements are one of the two issues identified in this category, which has been mentioned in 2 (3.6%) studies. Unlike feature code or some code types of cross-cutting concerns (e.g., exception handling or configuration), whose correctness easily attracts attention and can also be verified and validated via immediate practices such as software testing, it is quite challenging to verify and validate log statements [Chen 19]. As a result, log statements in source code may thus incorrectly reflect developer's *I&C*s, as confirmed in [Rong 20].

5.1.3 Issues across multiple categories

While the issues discussed above belong to a single category, there is one issue covering a relatively wider range of categories.

Performance overhead caused by log statements is a major issue in the *2-W* questions, which has been mentioned in 31 (55.4%) studies. Both the location and content of log statements can cause performance overhead. On the one hand, adequate logging is important to understand the system behavior comprehensively. The more log statements, the more information to be captured in the log files. On the other hand, excessive log statements may lead to unexpected side effects such as performance slowdown or high cost to disk I/O bandwidth [Shan 14], [Chow 18], [Liu 20], [Chen 17a], [Zeng 19], [Li 17b], [Li 17a], [Liu 19], [Li 20a]. In this sense, finding an appropriate trade-off between benefits and costs of log placement is thus both crucial and challenging. Although this issue in most cases is regarded as the issue in programming, researchers began to realize the importance of design in solving this issue [Lal 17], [Lal 16a], [Zhu 15], [Marr 18], [Ding 15], [Yao 18], [Sain 16], [Ghol 20], [Lal 16c], [Lal 16b], [Mizo 19], [Jia 18], [Lal 19], [Luo 18], [Li 20b], [Kabi 16a], [Yuan 12b], [Yuan 12a].

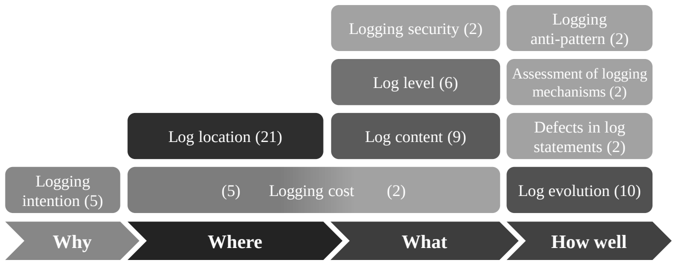

Fig. 7. Relationship between research topics and *3W1H* categorization.

Finding 1: Most studies focus on the issues related to log content (*what to log*), which directly impacts the information captured in log files. Among them, *lacking crucial messages* and *performance overhead* have attracted the majority of researchers' attention. The latter issue is also related to log location (*where to log*). In sum, the *2-W* questions (*where to log* and *what to log*) are the top concerns to researchers. A noteworthy phenomenon is that there are no specific issues associated to the category of *why to log*.

5.2 Research topics on logging practices (RQ2)

In this section, we categorized the research topics through manual coding with the *3W1H* questions. Generally, we identified 10 research topics and mapped them into the *3W1H* questions/categories. The detailed distribution of research topics is listed in Table 6 and the relationship between the research topics and the *3W1H* questions is shown in Fig. 7. It is obvious that most research topics pertain to the *2-W* questions, which is consistent with the findings we discussed in RQ1, i.e. most issues falling into the *2-W* questions. In contrast, there are relatively few studies focusing on research topics in the categories such as *why to log* and *how well is logging*.

The identified research topics in each category of *3W1H* are elaborated as follows.

5.2.1 "Why to log" topics

There is only one research topic in this category.

Logging intention is the only one we identified in this category, which has been raised by 5 (8.9%) studies. Li et al. performed a qualitative study to understand the benefits and costs towards logging practices from developers' perspectives [Li 20a]. In addition, approaches to balancing both aspects were also summarized in this study. Both benefits and costs to a certain degree were able to reflect the *I&C*s that developers may have when making logging decisions. Rong et al. carried out a case study at a real-world company to understand developers' *I&C*s on logging and found that the *I&C*s always had not been reflected well in the actual source code [Rong 20]. Pecchia et al. investigated the reasons for logging practices and found three major purposes of logging, i.e. state dump, execution tracing and event reporting [Pecc 15]. While these studies directly investigate logging intentions, several others explore the adoption of logging intentions in their research. For example, Li et al. studied the relationship between the topics of a code snippet and the likelihood of a code snippet being logged, where such topics can be regarded as intentions [Li 18a]. Jia et al. proposed two models to describe logging intentions, and further designed and implemented an automatic log placement tool based on the intention models [Jia 18]. One reason for investigating logging intentions is that developers may not be fully aware of them, leading to uncertainty about whether developers' intentions are properly reflected by the actual log statements in source code. In addition, automated approaches to logging improvement might not be convincing enough to developers without a clear understanding of developers' logging intentions [Li 20a].

5.2.2 "Where to log" topics

We identified one research topic in this category.

Log location, i.e. *where to log*, is a major research topic that has been investigated by 21 (37.2%) studies. As one part of the *2-W* questions, location of log statements affects performance, storage overheads, and many other aspects of the target system. Hence researchers attempted to establish an understanding of log location [Zhu 15], [Rong 18], [Pecc 15], [Fu 14], [Li 20b] and optimize log location [Cinq 09], [Cinq 10], [Yuan 12a], [Cinq 12], [Yao 18], [Sain 16], [Ghol 20], [Jia 18], [Kubo 20] so as to make logging practices more useful. In general, most studies attempted to predict log location and provided convenient tools to support developers in making such decisions. For instance, Lal et al. [Lal 17], [Lal 16a], [Lal 16c], [Lal 16b], [Lal 19], [Lal 15] conducted a set of studies that used machine learning to predict log statements in different code snippets. Several other studies tried to optimize log location through better log design. For example, Zhao et al. proposed an algorithm that can automate the placement of log statements in source code based on 'entropy' theory [Zhao 17].

5.2.3 "What to log" topics

We identified 3 research topics in this category.

Log content and log level are two research topics that appear at the same time in many cases. These two topics constitute the major part of *what to log*, which has been investigated by 9 (16.1%) studies and 6 (10.7%) studies, respectively. Researchers have identified the characteristics of log level and log content [Rong 18], [He 18], [Liu 20], [Marr 18], [Zhi 19] and also proposed several approaches to effectively capturing the information the log statements carry with from multiple perspectives, e.g., suitable log level [Anu 19], [Kim 19], [Mizo 19], [Li 17a], static text [Salf 04], [Yuan 12c], [Luo 18], [Tova 13], and necessary variables [Liu 19]. Similar to the research on log location, studies on log level/content also mainly focus on the prediction of appropriate level/content and the approaches to designing proper log level/content.

Logging security is another research topic in this category which contains two (3.6%) studies. In [Zhou 20], they pointed out the risk of data leakage of sensitive information through logging practices and studied its impact. Zhi et al. investigated the vulnerabilities that expose sensitive information through logging, and found the top root cause for the vulnerabilities is insecure whole-object logging,

TABLE 6
Distribution of research topics.

Category	Topic	Primary studies	Percentage
Why	Logging intention	[Jia 18], [Li 18a], [Li 20a], [Pecc 15], [Rong 20]	8.9%
Where	Log location	[Cinq 09], [Cinq 10], [Cinq 12], [Fu 14], [Ghol 20], [Jia 18], [Kubo 20], [Lal 16b], [Lal 17], [Lal 15], [Lal 16a], [Lal 16c], [Lal 19], [Li 20b], [Pecc 15], [Rong 18], [Sain 16], [Yao 18], [Yuan 12a], [Zhao 17], [Zhu 15]	37.2%
Where & What	Logging cost	[Chow 18], [Ding 15], [Li 20a], [Marr 18], [Zeng 19]	8.9%
What	Log content	[He 18], [Liu 20], [Liu 19], [Luo 18], [Marr 18], [Salf 04], [Tova 13], [Yuan 12c], [Zhi 19]	16.1%
	Log level	[Anu 19], [Kim 19], [Li 17a], [Marr 18], [Mizo 19], [Rong 18]	10.7%
	Logging security	[Zhi 20], [Zhou 20]	3.6%
How well	Log evolution	[Chen 17b], [Kabi 16b], [Li 17b], [Li 18b], [Pecc 15], [Shan 14], [Yuan 12b], [Chen 20], [Kabi 16a], [Zhi 19]	17.9%
	Assessment of logging mechanisms	[Cinq 10], [Cinq 20], [Pecc 12]	5.4%
	Logging anti-pattern	[Chen 17a], [Li 19]	3.6%
	Defects in log statements	[Chen 19], [Hass 18]	3.6%

i.e. developers make logging calls with direct reference to composite objects [Zhi 20].

5.2.4 "How well is logging" topics

Three research topics have been identified in the category of *how well is logging*.

Log evolution is a major research topic in this category involving 10 (17.9%) studies. These studies investigated the factors that influence the evolution of log statements and how they evolve. The research results of these studies can, to some extent, guide the implementation of log statements to reduce the overhead required for subsequent log maintenance. For example, several studies [Yuan 12b], [Chen 17b], [Shan 14], [Pecc 15] answered the question "*how do developers change log statements?*" from several different aspects, e.g., frequency and content of modification. Kabinna et al. examined the changes to log statements in four open source projects in order to reduce the maintenance effort [Kabi 16b]. Li et al. attempted to explore the reasons for changes made to log statements and thus provided suggestions on log evolution [Li 17b]. Similarly, Li et al. aimed to learn log evolution proactively from software evolution and provided a tool to guide log evolution [Li 18b]. Besides these studies, some researchers focused on the utilities supporting logging practices in terms of evolution. For example, Kabinna et al. studied logging library migrations within open source projects to help developers mitigate the migration effort [Kabi 16a]. The research result suggested that logging library migration is not a trivial task which requires significant design so as to balance the benefits and costs. Zhi et al. explored configurations of logging utilities in both open source projects and industrial projects, and found that about 12.9% of changes to logging configurations were to enhance usability and strengthen maintainability of log statements [Zhi 19]. To maintain high quality log statements, Chen et al. conducted a large-scale empirical study on the use of logging utilities in real-world projects and found that as the software systems evolve and grow bigger, the number of logging utilities also increased [Chen 20].

Assessment of logging mechanisms is a further research topic in this category, which has been studied by 3 (5.4%) studies. For example, Cinque et al. [Cinq 10] evaluated the effectiveness of current logging mechanisms in the context of three real-world case studies and found that logs were not able to provide useful information about failures in most cases, which was further confirmed in their subsequent work [Cinq 20]. Similarly, Pecchia et al. pointed out that it is crucial to understand the reasons behind the issues in logging mechanisms towards failure detection and the resultant low efficiencies so as to increase the accuracy of logs produced at runtime [Pecc 12]. Therefore, they proposed an experimental study on the factors determining accurate detection of software failures through logs and found that the effectiveness of the log location was related to the chance of error propagation paths to be exercised at runtime.

Logging anti-pattern is also a research topic belonging to the *how well* category, which is reported by 2 (3.6%) studies. The anti-patterns in log statements are recurrent mistakes which may hinder both the understandability and the maintainability of the resulted log files. Chen et al. [Chen 17a] conducted a comprehensive study to characterize and detect anti-patterns in log statements so as to develop and maintain high-quality log statements. Six main anti-patterns were identified by the authors, e.g., containing nullable objects, wrong verbosity level, and malformed output of variables without a human readable format. Similarly, as one of code smells, duplication of log statements has been studied by Li et al. [Li 19] to help developers improve their logging practices.

Defects in log statements is another research topic reported by 2 (3.6%) studies. The defects contained in log statements have become one of the major concerns due to the vast usage of logs in practice. Moreover, as claimed in [Hass 18], it often takes longer time for issues regarding log statement to be reported. Chen et al. extracted and studied the historical issues in log statements and their fixes based on several open source projects with the attempt to support effective maintenance of log statements [Chen 19].

5.2.5 Topics across multiple categories

We also identified one research topic that is across the categories of *where to log* and *what to log*.

Logging cost attracts concerns in 5 (8.9%) studies. As mentioned in Section 5.1.3, excessive logging may lead to performance issues. Therefore, empirical studies have been conducted to investigate this dilemma [Li 20a], [Chow 18], [Zeng 19] and several cost-aware logging mechanisms have been proposed. For example, Marron proposed a logging

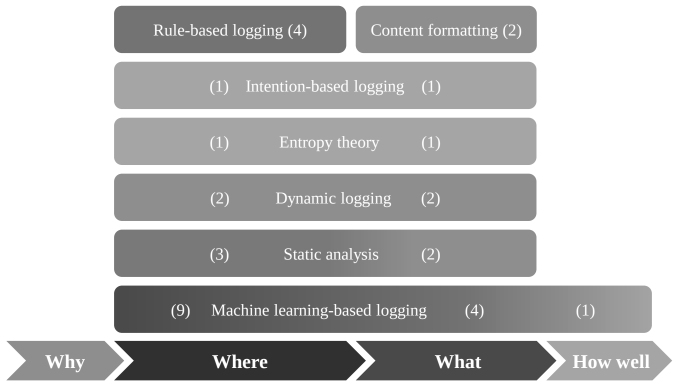

Fig. 8. Relationship between approaches and *3W1H* categorization.

system based on a set of design principles to make logging more efficient [Marr 18]. Ding et al. proposed a cost-aware logging mechanism that helps achieve a balance between logging overhead and effectiveness [Ding 15].

Finding 2: The distribution of research topics across multiple categories is generally similar to that of research issues. One noticeable exception is 'logging intention', one topic in the category of *why to log*, which received certain attention from the researchers. However, effective methods to obtain and satisfy developers' actual logging intentions or the related obstacles and issues to do so have never been essentially explored.

5.3 Proposed approaches and findings (RQ3)

Similarly, we also reviewed the approaches to logging practices through a manual coding process so as to understand the research progress and trend. Using the *3W1H* questions elaborated in Section 4, we identified 7 types of approaches covering three categories except *why to log*. The detailed distribution of the approaches is shown in Table 7. Note that all the approaches are categorized based on their inherent characteristics instead of the issues they are addressing. Besides, the relatively small percentages on the most right column imply that approximately half of the studies did not propose a concrete approach or tool. Apparently, most approaches address the *2-W* questions, which reasonably confirms the finding that the *2-W* questions are most concerned by the researchers in this community. In the following subsections, we elaborate the approaches proposed in the primary studies in each category.

5.3.1 "Where to log" approaches

One approach has been identified in this category.

Rule-based logging approach instruments log statements according to several manually or automatically defined rules. There are 4 (7.1%) studies discussing this approach. For example, Cinque et al. defined a minimal set of rules to be followed during programming log statements in order to effectively pinpoint failure locations [Cinq 09]. Their another work introduces a novel rule-based logging approach that leverages design artifacts to support effective log placement into source code [Cinq 12]. The rule-based logging approach was also applied in their follow-up work [Cinq 20] to increase the amount of useful information carried by log statements. In [Kubo 20], they developed a tool for identifying typical data structures and applying coding conventions (rules) to log statements so as to track inter-thread data dependencies. The advantage of rule-based logging is that key decisions can be designed in advance, thus enabling the standardization of the log location and content during the programming.

5.3.2 "What to log" approaches

We identified one approach in this category.

Content formatting is an approach to enforcing the output format of logs by designing a mechanism for formatting log statements, which has been proposed by 2 (3.6%) studies. For example, a logging mechanism proposed by Tovarňák et al. [Tova 13] is able to produce logs in a unified and extensible format allowing for efficient and automated processing. The approach proposed by Marron provides a suite of innovative log format and level management techniques that enable a consistent and unified log content [Marr 18]. As the result, the log statements can be well formatted logs for easy management and further processing.

5.3.3 Approaches across multiple categories

As a matter of fact, most proposed approaches can be classified into multiple categories, which implies that these approaches can be adopted to address issues across different categories.

Static analysis is an approach to addressing the *2-W* questions, which has been proposed by 5 (8.9%) studies. This approach analyzes the source code and thus builds a model for specific tasks such as failure diagnosis. For example, Yuan et al. [Yuan 12c], [Yuan 12a] developed a tool that uses Saturn static analysis framework [26] to identify potential unlogged exceptions. Similarly, Yao et al. [Yao 18] and Fu et al. [Fu 14] applied static analysis frameworks and tools to analyze source code in order to suggest suitable log location. Li et al. used static analysis to identify code smells of duplicate log statements to provide data basis for developers to obtain a clearer understanding of the system behavior and thus improve the logging practices [Li 19].

Dynamic logging is an approach which allows the output of log statements to be dynamically determined by the logging strategy or mechanism at runtime. The method has been advocated in 4 (7.1%) studies. For instance, a tool proposed by Mizouchi et al. is able to dynamically adjust the log level of a running system to record detailed logs for the abnormal events while limiting the amount/size of logs for normal events [Mizo 19]. Ding et al. presented a cost-aware logging system that can automatically determine whether to record runtime information according to predefined resource budget [Ding 15]. Similarly, Zhao et al. introduced an algorithm that can automate the placement of log statements within a specified threshold of performance overhead [Zhao 17]. Luo et al. presented a logging system for troubleshooting transiently-recurrent problems, with which logging information generated by a method over a period of time is proportional to how often it is reported for various misbehaviors [Luo 18]. In a nutshell, dynamic logging is a strategy that is able to adjust log

TABLE 7
Distribution of proposed approaches.

Category	Approach	Primary studies	Percentage
Where	Rule-based logging	[Cinq 09], [Cinq 20], [Cinq 12], [Kubo 20]	7.1%
Where & What	Static analysis	[Fu 14], [Li 19], [Yao 18], [Yuan 12a], [Yuan 12c]	8.9%
	Dynamic logging	[Ding 15], [Luo 18], [Mizo 19], [Zhao 17]	7.1%
	Entropy theory	[Hass 18], [Zhao 17]	3.6%
	Intention-based logging	[Anu 19], [Jia 18]	3.6%
What	Content formatting	[Marr 18], [Tova 13]	3.6%
Where & What & How well	Machine learning-based logging	[Anu 19], [Fu 14], [Ghol 20], [Kim 19], [Lal 16b], [Lal 17], [Lal 16a], [Lal 16c], [Lal 19], [Li 17a], [Li 17b], [Liu 19], [Sain 16], [Zhu 15]	25.0%

placement at runtime according to the system status, predefined rules, or some other metrics. Generally speaking, dynamic logging is mainly used in scenarios where resource or additional overhead is limited for logging.

Entropy theory in log design is another approach, which was adopted by 2 (3.6%) studies. Such approach calculates the entropy of the information carried by log statements or their context, and thus optimizes the location or enhances the content or level of these log statements. For example, Zhao et al. applied entropy to optimizing the location of log statements [Zhao 17]. Hassani et al. proposed a log level checker that used entropy to calculate the proper level of log statements based on the probability of appearance of phrases in the log content [Hass 18]. Based on the information contained in the content or context of the log statement, information entropy is believed to be able to assist the design of log placement.

Intention-based logging is an approach specifically to the *2-W* questions, which we identified in 2 (3.6%) studies. Jia et al. regarded the semantics of log context (usually the natural interpretation of the code comments) as logging intention and suggested that a better log placement strategy should take the logging intention into consideration [Jia 18]. As the authors pointed out, log placement was far beyond certain rules. Therefore, the authors proposed an Intention Description Model to describe the intention of log statements for a better log placement strategy. Similarly, Anu et al. also adopted the concept of logging intention and then proposed an automatic approach to assist decisions to appropriate log level [Anu 19]. Nevertheless, as intangible as the concept carries, intention is usually difficult to capture, communicate, implement and verify. As a result, there are usually many types of inconsistencies between the intention and the actual log placement in practice [Rong 20].

Machine learning-based logging is an approach that covers all the categories except *why to log*, proposed by 14 (25.0%) studies. These studies typically rely on various machine learning techniques for predicting the location, content, level of log statements or the necessity to make revisions. In essence, the prediction is a binary or multi-label classification problem, e.g., whether a log statement is needed [Li 17b] or which level should be applied [Anu 19] given the classification derived from various factors such as the contexts of code snippet [Zhu 15] or the characteristics (e.g., the number of variables, the number of existing log statements, etc.) of the source file containing log statements [Li 17a]. For example, several studies [Zhu 15], [Fu 14], [Kim 19] applied decision tree based algorithms as the learning model/classifier. Random forest was also one of the most commonly used classifiers for logging prediction [Anu 19], [Li 17b], [Sain 16], [Kim 19]. Several studies used multiple machine learning algorithms at the same time, and further compared their performance to find the best one. For example, Lal et al. adopted five different machine learning algorithms (i.e. Adaboost, Decision Trees, Random Forest, Gaussian Naive Bayesian, K-Nearest Neighbor) in their approaches [Lal 16c], [Lal 16b], [Lal 19]. Moreover, in their further studies [Lal 17], [Lal 16a], they proposed ensemble-based approaches to capturing the strength of multiple base classifiers. Two studies [Sain 16], [Kim 19] used support vector machine to recommend log statements. In addition, deep learning [Ghol 20] and neural networks [Liu 19] were also applied to logging prediction for complicated multi-label classification tasks. Ordinal regression model [Li 17a] was used when the number of labels to the classification is small yet the relative ordering among these labels is critical. One notable challenge of the machine learning based approach is that the prediction performance highly relies on the quality of the dataset used for model training. Nevertheless, as implied in [Rong 18], [4], the quality of logging practices is far from satisfactory in real-world software projects. Therefore the quality of training dataset (normally based on real-world software projects) is often questionable.

Finding 3: Only about half of the studies proposed concrete solution approaches to specific issues regarding logging practices, mainly focusing on the categories of *where to log* and *what to log*. Among them *machine learning-based logging* has received the most attention.

5.4 Cross analysis (RQ4)

In this section, we aim to answer RQ4 by extending the analysis of the extracted data to across different research questions. Apparently, the current research work on logging practices is anchored in issues. To this end, we detail RQ4 into three concrete Cross-Analysis Questions (CAQs) as follows:

CAQ1: Are there trends or patterns in the investigation of issues over the years?

CAQ2: Which research topics have received the most attention in relation to different issues?

CAQ3: Which approaches have addressed the most number of issues?

In the following, we present the findings from the cross analysis.

5.4.1 *Investigated issues over the years (CAQ1)*

To portray a general status of the issues under investigation over years, we list all the issues identified in Section 5.1 according to the year of publication. The result is shown in Table 8, from which two points seem to be noteworthy.

Firstly, some of the hot issues in the recent years did not attract enough attention in the early years. For example, **PO** (Performance Overhead) and **LCM** (Lacking Crucial Messages) are two issues attracting significant and continuous attention recently. One possible reason for this may lie in that these two issues are the most direct issues in carrying out logging practices, compared to other issues. With more attention to logging practices, these two issues turn to be increasingly prominent.

Secondly and more importantly, except for the **LSD** (Leakage of Sensitive Data) and **DVLS** (Difficulties in V&V of Log Statements) issues, all other issues raised in the early years have been re-investigated several times in the following years, which implies that they might not be well addressed yet. Taking 2015 as a watershed, the most studied issues before 2015 remain as the most investigated ones after 2015.

Finding 4: The major issues regarding logging practices have been continuously studied, implying these issues are still yet to be completely solved. Findings 5–6 from further cross-analysis have nailed down the major causes leading to this outcome. Besides, *leakage of sensitive data* and *difficulties in V&V of log statements* have been attracting researchers' attention since 2019.

5.4.2 *Issues and research topics (CAQ2)*

The result from the cross analysis between the issues (cf. Section 5.1) and the research topics (cf. Section 5.2) is presented in Table 9. It is worth noting that the numbers outside the parentheses denote the number of studies in which an issue was discussed or addressed under a certain research topic, while the percentage in the parentheses indicates the corresponding proportion. We elaborate several interesting observations as follows.

Firstly, from the issue's perspective (i.e. horizontal view), almost all issues appear across multiple research topics. Among these issues, **PO** (Performance Overhead), **LCM** (Lacking Crucial Messages), **RUM** (Redundant or Useless Messages), **IAM** (Incorrect or Ambiguous Messages), **HLM** (Heterogeneity of the Log Messages) and **MB** (Maintenance Barriers) are covered by relatively more research topics. Note that these are also the most investigated issues according to the findings for RQ1.

Secondly, from the angle of research topic (i.e. vertical view), we observe a similar pattern, i.e. nearly all research topics cover multiple issues. For example, **LIT** (Logging Intention), **LLC** (Log Location), **LCS** (Logging Cost), **LCT** (Log Content), **LLV** (Log Level) and **LEV** (Log Evolution) all investigate most of these issues.

The findings presented in Table 9 show that there are many-to-many relationships between the issues and research topics, which further imply that issues regarding logging practices may be naturally entangled with each other. Therefore, it is unlikely to rely solely on a single research topic to address one issue at a time. A systematic strategy which contains multiple aspects regarding logging practices may be required to deal with the major issues.

Finding 5: There are a number of many-to-many relationships between the issues and the research topics, i.e. an issue appears in multiple research topics, and a research topic attempts to address multiple issues. Some of the issues even cover almost all topics, and vice versa. This observation suggests a lack of profound understanding of the issues and their intricate relationships and how they should be appropriately tackled.

5.4.3 *Issues and proposed approaches (CAQ3)*

Similar to CAQ2, the evidence presented in Table 10 reveals the relationship between issues (cf. Section 5.1) and approaches (cf. Section 5.3). Apparently, the observation that one issue has multiple approaches and one approach solves multiple issues also exists in Table 10. To be specific, from the angle of issues, **PO** (Performance Overhead), **LCM** (Lacking Crucial Messages), **RUM** (Redundant or Useless Messages), **IAM** (Incorrect or Ambiguous Messages) and **HLM** (Heterogeneity of the Log Messages) have all attracted multiple studies with various approaches. On the other hand, all the approaches are claimed to address multiple issues, among which, **MLBL** (Machine Learning-Based Logging) is claimed to solve more issues than other approaches do. However, no specific approach has been proposed to address the issues of **LSD** (Leakage of Sensitive Data) and **DVLS** (Difficulties in V&V of Log Statements), which to a certain degree implies the difficulty of solving these two issues. The result listed in Table 10 confirms our conjecture that there are no commonly accepted solutions to the major issues in logging practices, given the fact that these issues have been continuously studied in recent years. This observation partially explains the low adoption of research solutions by the industry.

Finding 6: Similar to Finding 5, the relationships between the issues and the proposed approaches are also mostly many-to-many, i.e. one approach is often claimed to solve multiple issues and vice versa. This again to a fair degree reflects an inadequate analysis of the depth of an issue to be addressed, which inevitably hinders tackling the issue at an appropriate granularity and level. As a result, such a solution is generally far from solving the intended issue and the existence of multiple inadequate solutions makes it even harder to adopt any one in practice.

TABLE 8
Issues over years.

Issue and corresponding category		Year											
		2004	2009	2010	2012	2013	2014	2015	2016	2017	2018	2019	2020
Where & What	**PO***				2		2	3	5	5	6	4	4
What	**LCM***		1	1	4	1	1	1	4	4	7	5	6
	RUM*		1		1	1	2	2	3	2	4	4	3
	IAM*		1	1	2	1		1		1	2	3	1
	HLM*	1	1			1		1			2		3
	LSD*												3
How well	**MB***				1		1	1	1	2	1	1	2
	DVLS*											1	1

* **PO** (Performance Overhead); **LCM** (Lacking Crucial Messages); **RUM** (Redundant or Useless Messages); **IAM** (Incorrect or Ambiguous Messages); **HLM** (Heterogeneity of the Log Messages); **LSD** (Leakage of Sensitive Data); **MB** (Maintenance Barriers); **DVLS** (Difficulties in V&V of Log Statements).

TABLE 9
Issues versus research topics.

Issue and corresponding category		Topic and corresponding category										
		Why	Where	Where &What	What			How				Total
		LIT†	LLC†	LCS†	LCT†	LLV†	LSC†	LEV†	ALM†	LAP†	DLS†	
Where & What	**PO***	3 (9.7%)	15 (48.4%)	5 (16.1%)	4 (12.9%)	3 (9.7%)		4 (12.9%)		1 (3.2%)		31 (100.0%)
What	**LCM***	3 (8.6%)	19 (54.3%)	2 (5.7%)	6 (17.1%)	3 (8.6%)		4 (11.4%)	2 (5.7%)		1 (2.9%)	35 (100.0%)
	RUM*	2 (8.7%)	9 (39.1%)	4 (17.4%)	5 (21.7%)	3 (13.0%)		3 (13.0%)			1 (4.3%)	23 (100.0%)
	IAM*	2 (15.4%)	4 (30.8%)	1 (7.7%)	3 (23.1%)	1 (7.7%)		2 (15.4%)	2 (15.4%)	2 (15.4%)	1 (7.7%)	13 (100.0%)
	HLM*	1 (11.1%)	3 (33.3%)	1 (11.1%)	5 (55.6%)	1 (11.1%)		2 (22.2%)				9 (100.0%)
	LSD*	1 (33.3%)		1 (33.3%)			2 (66.7%)					3 (100.0%)
How well	**MB***	2 (20.0%)	2 (20.0%)	1 (10.0%)				6 (60.0%)		1 (10.0%)	1 (10.0%)	10 (100.0%)
	DVLS*	1 (50.0%)									1 (50.0%)	2 (100.0%)

* The same as Table 8.
† **LIT** (Logging Intention); **LLC** (Log Location); **LCS** (Logging Cost); **LCT** (Log Content); **LLV** (Log Level); **LSC** (Logging Security); **LEV** (Log Evolution); **ALM** (Assessment of Logging Mechanisms); **LAP** (Logging Anti-Pattern); **DLS** (Defects in Log Statements).

TABLE 10
Issues versus approaches.

Issue and corresponding category		Approach and corresponding category							
		Where	Where&What				What	Where& What&How	Total
		RBL†	SA†	DL†	ET†	IBL†	CF†	MLBL†	
Where & What	**PO***		3 (16.7%)	4 (22.2%)	1 (5.6%)	1 (5.6%)	1 (5.6%)	10 (55.6%)	18 (100.0%)
What	**LCM***	4 (18.2%)	4 (18.2%)	3 (13.6%)	2 (9.1%)	1 (4.5%)	1 (4.5%)	9 (40.9%)	22 (100.0%)
	RUM*	2 (15.4%)	1 (7.7%)	1 (7.7%)	1 (7.7%)	1 (7.7%)	2 (15.4%)	6 (46.2%)	13 (100.0%)
	IAM*	2 (40.0%)	1 (20.0%)		1 (20.0%)		1 (20.0%)		5 (100.0%)
	HLM*	1 (25.0%)					2 (50.0%)	1 (25.0%)	4 (100.0%)
	LSD*								0 (0.0%)
How well	**MB***							1 (100.0%)	1 (100.0%)
	DVLS*								0 (0.0%)

* The same as Table 8.
† **RBL** (Rule-Based Logging); **SA** (Static Analysis); **DL** (Dynamic Logging); **ET** (Entropy Theory); **IBL** (Intention-Based Logging); **CF** (Content Formatting); **MLBL** (Machine Learning-Based Logging).

6 Discussions

In this section, we discuss possible reasons behind the current status of the research and the adoption of logging practices as well as several next-step considerations for logging practices.

6.1 Research status and reason analysis

In essence, a log statement is quite similar to any business statement in source code—they are both executable program instructions. Both types of statements are supposed to implement certain 'requirements', which will produce a certain 'result' for their intended users later on. In a typical scenario, the log statement usually generates logs for internal users such as developers and maintainers while regular business statements produce 'results' for external end-users. Unlike coping with conventional coding for features, which is significantly supported by the state-of-the-art of SE, however, logging practices currently still lack a systematic methodological support for practitioners based on our observation derived from this SMS. We reckon there are multiple reasons leading to the current status of logging practices, among which "lack of research to address critical issues", "unrealistic expectation of general yet adaptable solutions", and "separated research within logging practices and between logging practices and log analysis" may have played an important role.

6.1.1 Lack of research to address critical issues

We noticed that most research effort has been invested on the issues relating to the 2-*W* questions, essentially (if not completely) neglecting two equally critical categories, i.e. *why to log* and *how well is logging*.

According to the elaboration in Section 4, *why to log* answers the *I&C*s of logging practices, which should serve as the starting point of logging practices. Unfortunately, very little effort has been put onto identifying and addressing issues in this important category. Just like regular software development, lack of requirement analysis will inevitably lead to deviations in software design and implementation. Logging practice without proper clarification of *I&C*s will also lead to major gaps between the *I&C*s and the actual log statements in source code, which has been noticed and reported in [Rong 20], [Li 20a]. More importantly, *I&C*s are the conceptual abstraction of *why to log* and usually used to support communication among different participants regarding logging practices. An *I&C* is implemented via log placement by translating to log location, content, and level. Clearly, without properly understanding and analyzing the true *I&C*, it is difficult to implement an appropriate log placement that matches the context in source code. For this reason, practitioners may encounter barriers to adopting a logging approach or tool for log placement.

Similarly, the issues relating to the category of *how well is logging* have not been well addressed by the current research on logging practices. Like the importance of V&V in regular software development, without proper V&V in logging practices, it is usually uncertain that log statements actually reflect developer's original *I&C*s and capture the system information needed for further analysis. As a result, logging practices are inevitably subject to many questionable log statements in actual source code. In fact, several studies have emphasized the importance of fulfilling logging *I&C*s [Yao 18], [Anu 19], [Jia 18], but none of them provided concrete V&V mechanisms to ensure proper implementation of logging *I&C*s.

Given the current status elaborated above, a very interesting analogy is that the state-of-the-art logging practices are quite similar to the famous 'Code and Fix' software development model in some sense. While it may work well with software systems of small size, it will encounter huge challenges to deal with large and complex software systems. Similarly, typical scenarios of logging practices nowadays always involve with large-scale software systems with complex business logic. In this sense, it is not surprising that logging practices are far from satisfactory in these software systems [Rong 18], [4].

6.1.2 Unrealistic expectation for general yet adaptable solutions

An ultimate goal of research on logging practices is to provide guidance or assistance for software developers to perform better logging practices. To achieve this goal, several approaches and tools have been proposed. However, in practice, it is extremely challenging or even unrealistic to expect solutions that are generally applicable to all logging practices yet they are adaptable enough to perform the best in various specific contexts.

One example is the expectation of general-purpose guidelines. Many studies have mentioned the importance to have practical guidelines for logging practices [Chen 17a], [Zhu 15], [He 18], [Liu 19], [Anu 19], [Liu 20], [Li 18a]. Not only is the academia committed to putting forward such guidelines, but there are also similar requests in industry. For example, some popular blogs have discussed the best or worst logging practices, e.g., [21], [22], which could be taken as reference guidelines. Some world-leading software companies have also introduced internal guidelines for logging practices, e.g., Alibaba [20]. Most of these guidelines are so-called general-purpose guidelines which guide logging practices without an explicit and specific *I&C*. However, there is less chance to promote general-purpose logging guidelines due to the vast variety of *I&C*s. Take two typical logging intentions as an example. Failure diagnosis and performance analysis may require completely different log placement. The logical branches in source code may provide useful information for the former logging intention, while the timestamps at the entries and exits of complicated methods may support the latter logging intention better. In this example, designing general-purpose logging guidelines to satisfy both intentions might never be feasible.

Another example is the prevailed adoption of machine learning techniques to support logging practices, which involves more number studies than others (cf. Table 7). The main idea is to learn the contexts/features of code snippets and recommend log statements at code snippets with similar contexts/features. The challenge is, however, the efficacy of machine learning techniques largely depends on the quality of the dataset used for training. If only general-purpose log statements are included in the training dataset, the trained models cannot support specific *I&C*, i.e. there may be totally different log statements at the exactly same location in the source code to meet different *I&C*s. It is worth noting that it is non-trivial to ensure the quality of log placement in existing software systems in the first place. We noticed that several studies proposed methods for extracting *I&C*s from source code [Anu 19], [Jia 18]. This may bring some opportunities for better use of machine learning techniques to support logging practices as long as the *I&C*s can be adequately learned and included in the resulting models.

Although a certain log placement is commonly related to specific and concrete contexts (e.g., *I&C*s, structure/logic of the code nearby), the specific contexts normally limit the generality of a certain log placement in return. This is a dilemma. On the one hand, we need a solution to logging practices (e.g., general purpose guidelines) that is generally applicable to as many contexts as possible. Otherwise, the usage of the solution is inevitably constrained by the contexts. On the other hand, the more contexts involved, the more unlikely to find a suitable log placement to address various *I&C*s for various specific contexts.

6.1.3 Separated research within logging practices and between logging practices and log analysis

It seems that the related research around logs, log placement and subsequent log analysis are in a state of separation. This has resulted in a situation in which research on how to optimize log placement is rarely considered from the perspective of what information is needed for log analysis. On the contrary, in the related research on log analysis, most of the research efforts have been spent on analysis and processing algorithms using existing log data, researchers rarely express concerns about the quality of the data source—log placement. In short, there is a lack of positive interaction between log placement and log analysis, which may be one of the reasons for the current research status of logging practice. Take one of the most studied issue (i.e. performance overhead) as an example, without information requirement derived from the corresponding log analysis need, reasonable trade-offs are inherently impossible to achieve.

Within logging practices, while the research topics and the proposed approaches are concentrated on the *2-W* questions, some key aspects regarding logging practices specially around *why* and *how well* questions are essentially neglected. This separation of research leads to the consequence that some proposed approaches cannot be verified and validated from the perspective of logging *I&C*s. It is very desirable to consider a holistic research approach that takes into account all *3W1H* aspects and the interaction between logging practices and logging analysis.

6.2 Next steps

The importance of logging practices in modern software development and operations is undeniable. Therefore, we discuss several promising next-step research in this subsection.

6.2.1 A process perspective for logging practices

Apparently, the chance is slim to solve the critical issues regarding logging practices (as shown in Table 5) by solely considering one or two categories. For example, *performance overhead* is obviously a compromise between the benefit and cost of logging, which can not be completely addressed by merely answering the *2-W* questions. Without knowing specific information needs (perhaps derived from log analysis), the issue of *lacking crucial messages* is unlikely to be addressed, and further all the issues relating to 'messages' may be subject to this observation. Therefore, we advocate a process perspective for logging practices in which not only is logging systematically considered from log generation to log utilization but can logging issues be also holistically investigated. For example, we may borrow the idea from regular software development processes, given the essentially similar paradigm to develop and maintain both log statements and regular business statements. The concept of Software Development Life Cycle (*SDLC*) [27] has already taken shape in existing studies regarding logging practices.

For example, Jia et al. proposed an 'intention' description model that is able to represent developer's intentions of log statements [Jia 18]. Based on the 'intention' description model, they implemented an intention-aware log automation tool to insert log statements at proper places. An 'intention' is similar to the concept of a requirement in *SDLC*. Cinque et al. proposed a rule-based logging approach based on the artifacts generated at the design stage [Cinq 12]. Zhao et al. proposed an algorithm to optimize the location of log statements in source code based on information theory [Zhao 17]. Kabinna et al. examined changes to log statements in order to help developers instrument more stable log statements [Kabi 16b]. The maintenance of log statements was brought into the limelight. Nevertheless, to close the loop, the value of **V&V** (i.e. *how well is logging*) should be taken seriously in logging practices.

Moreover, apart from the process perspective, some common good practices in SE can also be useful to logging practices due to the fact that log statements are source code in essence. For example, several studies have realized that conducting logging practices in a 'Code and Fix' fashion is problematic, and thus summarized a set of anti-patterns [Chen 17a]. Along this thread, adapting the well-known best practices in regular SE to logging practices may also be a valuable research direction.

6.2.2 Recognizing the anchor value of I&C

The *I&C* of a log placement and its underlying context should be taken as the starting point and also the anchor of all logging practices. Without a clear answer to the *why to log* question, it is hard to design and implement an appropriate log placement, and all the downstream activities around logging practices would become unrooted trees. Moreover, as mentioned in [Li 20a], without a clear understanding of developers' logging *I&C*s, automated approaches to logging improvement may not be convincing to developers. In fact, several studies have already raised the importance of logging intention [Anu 19], [Jia 18] to support log placement. However, a systematic approach to extracting, defining, and representing multiple logging *I&C*s, especially the real *I&C*s from relevant stakeholders, is still yet to be developed. One good example is the pervasive DevOps approach in which staff in charge of operations are encouraged to contribute to the development of requirements from their perspectives [28]. In the same spirit, consumers of the information contained in logs by means of log analysis should also play a part in defining *I&C*s for generating appropriate logs.

With the *I&C*s clearly defined, the next question is how to implement them through log placement. Compared to current logging practices in which developers usually consider log placement during coding [Cinq 09], [Pecc 15], [Shan 14], we advocate a shift-left strategy for key logging decisions. As suggested in [Shan 14], logging should be thoroughly designed first rather than just deferred to the implementation stage. However, although several approaches have been proposed (cf. Fig. 8) to implement *I&C*s in log placement, the next-step research needs to explore how to effectively reflect multiple *I&C*s in log placement and address *I&C*s in multiple artifacts at different development stages in a timely manner.

7 Threats to validity

This paper attempts to provide a systematic and comprehensive overview of the state-of-the-art logging practices in SE, which is based on our review of 56 primary studies. In this section, we discuss potential threats to the validity of this study, the approaches by which we strived to mitigate them, and other aspects that need to be taken into consideration in order to generalize the results of this study. The threats to validity are organized into four categories (i.e. the Construct, Conclusion, Internal and External) as proposed in [29].

7.1 Construct validity

Construct validity is concerned with the issues that to what extent the object of study truly represents theory behind the study [29]. In this study, the main treats related to this validity are the suitability of research questions and the schema for data extraction.

The suitability of research questions determines whether the research objective can be addressed. To minimize the threat derived from this factor, all the research questions are designed based on the consensus through team discussion. Meanwhile, using an iterative way, some trials have been conducted for justification towards data extraction, evidence answering research questions as well as research questions addressing research objective. To this end, the threats related to research questions could be minimized.

The rationality of research scope determines whether the selected studies are able to provide appropriate information to answer the research questions. We excluded the studies focusing on log analysis in this systematic mapping study. However, since useful information may also be extracted from studies on log analysis to support logging practices (as elaborated in Section 6.2.1, we acknowledge the importance of the information needs derived from log analysis), the research scope in our study may inevitably leave out some relevant studies. Nevertheless, as we have found that the work reporting the impact of log analysis on logging practices is generally scarce and further the setting of the research scope is based on our previous experience [5] and drawing on similar research [Yuan 12b], [He 18], [Kabi 16b], the threat could be minimized.

The data extraction schema determines the quality of the evidence we may obtain to answer research questions. To mitigate this threat, we have applied a standard classification [10] and finalized the schema through several optimization iterations.

7.2 Conclusion validity

Conclusion validity is a measure of the reasonable degree to which a research conclusion could be trusted. For the sake of reaching reasonable conclusions, we adopted a manual coding approach described in Section 2.7 to assist data synthesis. Nevertheless, this method inherently carries with validity threats.

The categorization results guide the whole aggregation and synthesis process and draw the major conclusions. To mitigate the threat, each the primary study was peer reviewed by at least two researchers and each finding was derived from open discussion. Once a disagreement emerges, a consensus has to be reached before further work. In this way, the threat derived from categorization can be controlled.

7.3 Internal validity

Internal validity is the extent to which a study establishes a trustworthy cause-and-effect relationship between a treatment and an outcome. To make sure that this SMS is repeatable, the search string, search engines, inclusion/exclusion criteria and data extraction schema were cautiously designed, carefully tuned and explicitly presented. In our study, the main threats to the internal validity arise from the limitation of the search string and search engines, as well as personal bias in applying inclusion/exclusion criteria and performing data extraction.

The limitation of the search string and search engines may lead to an incomplete set of primary studies. Different authors may use different terms to refer to a similar concept. In order to mitigate the risk of incomplete retrieval of the relevant studies, a formal search process has been designed and followed, combining manual search, automated search and snowballing, in an iterative manner. To control threats due to search engines, we have included digital libraries that are believed to be suitable depending on the existing protocols [17]. Therefore, we consider our retrieval to be nearly complete, and if any primary studies were missed, that percentage would be negligible.

The personal bias may lead to subjective decisions occurred during paper selection and data extraction. In order to control the impacts, in the paper selection and data extraction process, an iterative strategy has been applied in the selection process in which the data extraction was performed collaboratively by multiple reviewers with cross-checking. Therefore, the threats derived from personal bias can be mitigated.

7.4 External validity

External validity is concerned with to what extent the SMS results can be generalized. One possible threat is related to the degree to which the primary studies are representative for the review topic.

Representativeness of the included primary studies is critical in any SMS study. In order to mitigate this external threats, the search process presented in Section 2.5 was defined after several pilot searches and validated with open discussion with all the researchers in this work. We argue that the relevant primary studies in our final pool contain sufficient information to represent the research topic discussed in this paper.

8 Related empirical studies

Logging practices have attracted increasing attention recently. Therefore, some empirical research on logging practices has been conducted from various perspectives.

For instance, Yuan et al. investigated logging practices using four pieces of large open source software, quantifying the pervasiveness and the benefit of logging [Yuan 12b]. They identified several particular aspects in logging choices where developers spend most efforts in getting them right, as well as many opportunities for tools to improve the logging practices. A large-scale replication study on similar topics was reported in [Chen 17b] later to confirm these findings.

In [Rong 18], Rong et al. focused on the quality of logging practices in 28 popular open source projects on GitHub. The researchers mined evidence from these projects, which implied major issues in the logging practices in these 28 projects (e.g., very low and divergent density of log statements, arbitrary location to put log statements), easily leading to questionable implementation of the logging purpose—to capture the intended information of system behaviors. Similarly, the empirical study conducted by OverOps [4] also indicated these issues in logging practices. Besides, the researchers also found a large portion of log statements tended to contain insufficient variables to record as much information as intended, indicating the low quality of logging practices in the industry. Chen et al. characterized six anti-patterns in log statements by carefully studying the development history of three open source software systems from different application domains [Chen 17a].

Besides these empirical studies, there are also a handful of secondary studies attempting to portray the research and adoption status of logging practices with a primary focus on current logging practices, common challenges and proposed solutions (cf. Table 11), which are elaborated below.

• As a pilot of this work, we have carried out a review on logging practices [5] in 2017. However, due to page limits, we did not discuss evidence and relevant implication in detail.

• Sambasivan et al. carried out a survey on the tracing infrastructures for distributed systems and distilled the design space of workflow-centric tracing and described key design choices [6]. However, this study focuses on the building of logging infrastructure, which according to the discussion in Section 2.2 falls outside the scope of this SMS.

• Cândido et al. also conducted an SMS on log-based software monitoring [7]. However, the topic of 'logging practice' is only one out of four research focuses in this study, as a result, very limited insights into logging practices are provided by this study.

• A most recent survey study conducted by Chen et al. [2] also reviewed software log instrumentation, a concept similar to logging practices in our work. The major differences between Chen's work and ours are three-fold. First and foremost, the two studies carry with different research objectives. While Chen's work is focused on identifying the challenges and the proposed solutions used in log instrumentation, our focus is on unveiling possible problems and gaps which further shed light on the potential future research directions by establishing a comprehensive understanding of the research status of logging practices. Second, due to different research objectives, the two studies have also adopted different research methods. While both work use thematic analysis and qualitative methods to identify challenges and solutions, we additionally apply a quantitative method to depict a research landscape of current logging practices and more importantly a cross-analysis method to unveil the gaps between challenges and solutions. The findings (Findings 1–6) with these methods extend our understanding of the state-of-the-art research of logging practices. Last but not least, although the two studies use a similar dataset (i.e. selected literature) for final synthesis, our work identified nine additional high quality studies not included in Chen's work, most of which have been published on premier venues in their respective fields, e.g., [Li 20a], [Anu 19], [Li 20b], [Luo 18], [Pecc 12], etc. As a result, our work identified several critical issues/challenges regarding logging practices not in Chen's paper. For example, "Redundant or useless messages", reported by 9 primary studies, were not discussed in Chen's work, as this issue may inevitably impact developer's capability to perform failure diagnosis or performance analysis, which is listed in Chen's work as one major challenge category.

In summary, existing studies have exposed the challenges/solutions of logging practices from different perspectives, which means the challenges with which developers are confronted when conducting logging practices are well known in the community. However, a holistic understanding of the research state of logging practices is yet to be established, which is the major motive for our work. It is worth highlighting the major contributions of our work as compared to the other secondary studies, which are: (1) we portray a profound research state landscape of logging practices in terms of issues, research focuses and solution approaches using *3W1H* categorization scheme through a *systematic mapping study*; (2) we pinpoint the gaps between challenges and solutions and suggest potential remedies and future research directions to close the loop through *cross-analysis*.

9 Final remarks

Logging is an important task of software development and in recent years research on logging practices has been steadily increasing. However, not quite aligned with this trend, adoption of the proposed approaches and solutions by the software industry remains low. In this paper, we have presented the results of an SMS on logging practices from 56 primary studies in order to paint a landscape of the state-of-the-art of logging practices for a holistic understanding of this research area.

The main observations from this study can be summarized as follows:

TABLE 11
Comparison against other reviews related to logging practices.

Authors	**Focus**	**Year**	**PC**	**OC**	**A**	**B**	**C**	**D**	**E**
Sambasivan et al. [6]	To build workflow-centric tracing infrastructures for distributed systems	2016	N/A	N/A			☑		
Rong et al. [5]	Occasions where logging practices are needed, challenges, current state-of-the-art, and future research directions of logging practices	2017	41	10	☑	🗹	☑	○	
Cândido et al. [7] (previous work [30])	To present the research focus, opportunities, and directions on log-based software monitoring, in which the logging practice is only one of the three topics beside log infrastructure and log analysis.	2021	108	19		☑		○	
Chen et al. [2]	Techniques, challenges, and possible solutions of conventional logging, rule-based logging, and distributed tracing	2021	69	34	☑		☑		
This study	Issues, research topics, proposed approaches, and gap areas regarding the research on logging practices	2021	56	N/A	☑	☑	☑	●	☑

[*] PC: Peer-reviewed literature count; OC: Overlapping peer-reviewed literature count with other reviews;
A: Did the study include issues regarding logging practices? B: Did the study include research topics around logging practices? C: Did the study include proposed approaches for logging practices? D: Did the study quantitatively analyze the research focus on issues, research topics, proposed approaches and the relationships between them? E: Did the study identify gaps in current research on logging practices? [N/A]: Not applicable;
☑ Issues, research topics, proposed approaches and gap areas are addressed directly through research questions in the study;
🗹 Issues, research topics, proposed approaches and gap areas are addressed partially through other research questions in the study (i.e. by addressing a different research question, partial information is provided).
● Quantitative analysis is applied to every aspects of issues, research topics, proposed approaches and gap areas.
○ Quantitative analysis is applied to some aspects of issues, research topics, proposed approaches and gap areas.

Firstly, there is some consensus on the major issues in logging practices. However, even the issues that have received the most attention are still being discussed and explored repeatedly, implying that the proposed solutions are yet workable as expected.

Secondly, based on the distribution of issues, research topics, and approaches, the *2-W* questions are still the focus of common concern. However, only addressing the *2-W* questions without considering other questions may has clear limitations. Therefore, research to identify and address the issues relating to the categories of *why to log* and *how well is the logging* should be encouraged in the community.

Last but not least, the many-to-many relationships between issues, research topics and approaches indicate a lack of profound understanding of the real issues and how they should be appropriately tackled.

The value of our SMS is not only limited to showing possible solutions to the issues/challenges of logging practices, which are also discussed in several existing secondary studies to some extent. A more significant contribution of this SMS lies in that it reveals the current problems and omissions in the research related to logging practices. If these problems are not adequately concerned and addressed, the status quo of research on logging practices will likely remain unchanged.

Based on the results of this study, we make the following recommendations:

- As the starting point of logging practices and the anchor point of downstream logging practices, logging *I&C*s should be given with full attention. The source (e.g., information needs derived from log analysis), performance overhead limits (e.g., derived from regular system requirements) and other contextual factors should be clarified before carrying out logging practices. For this purpose, research effort is demanded to explore pragmatic practices, methods and tools.
- A process perspective and a holistic approach should be considered to propose more effective solutions to logging practices, which means that the current research direction and focus need to be adjusted. For example, the *2-W* questions should not be investigated and addressed separately and also without considering other questions. Proposed solutions should not only implement *I&C*s but also verify and validate them by downstream practices. In short, the practices, methods and tools for planning, designing, producing, analyzing and consuming logs should be studied in a systematic manner.
- With DevOps becoming the mainstream method of software development, operations and maintenance [31], the use of tools to improve the level of automation becomes an apparent need that has to be considered, but how to design tools to meet the basic requirements of the above two points calls for further research efforts.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No.62072227, No.61802173), the National Key Research and Development Program of China (No.2019YFE0105500) jointly with the Research Council of Norway (No.309494), the Key Research and Development Program of Jiangsu Province (No.BE2021002-2), as well as the Intergovernmental Bilateral Innovation Project of Jiangsu Province (No.BZ2020017).

References

[1] B. Fitzgerald and K.-J. Stol, "Continuous software engineering: A roadmap and agenda," *Journal of Systems and Software*, vol. 123, pp. 176–189, Jan. 2017.
[2] B. Chen and Z. M. J. Jiang, "A survey of software log instrumentation," *ACM Computing Surveys (CSUR)*, vol. 54, no. 4, pp. 1—34, Jul. 2021. [Online]. Available: http://dx.doi.org/10.1145/3448976
[3] B. W. Kernighan and R. Pike, *The practice of programming*. Addison-Wesley Professional, 1999.
[4] OverOps, "The complete guide to Java logging in production," 2017. [Online]. Available: https://land.overops.com/java-logging-in-production-ebook/

[5] G. Rong, Q. Zhang, X. Liu, and S. Gu, "A systematic review of logging practice in software engineering," in *2017 24th Asia-Pacific Software Engineering Conference (APSEC '17)*. IEEE, Dec. 2017, pp. 534–539. [Online]. Available: http://dx.doi.org/10.1109/APSEC.2017.61

[6] R. R. Sambasivan, I. Shafer, J. Mace, B. H. Sigelman, R. Fonseca, and G. R. Ganger, "Principled workflow-centric tracing of distributed systems," in *Proceedings of the Seventh ACM Symposium on Cloud Computing*. ACM, Oct. 2016, pp. 401–414. [Online]. Available: http://dx.doi.org/10.1145/2987550.2987568

[7] J. Cândido, M. Aniche, and A. van Deursen, "Log-based software monitoring: A systematic mapping study," *PeerJ Computer Science*, vol. 7, p. e489, 2021. [Online]. Available: https://doi.org/10.7717/peerj-cs.489

[8] V. R. Basili, "Goal question metric paradigm," *Encyclopedia of software engineering*, pp. 528–532, 1994.

[9] K. Petersen, S. Vakkalanka, and L. Kuzniarz, "Guidelines for conducting systematic mapping studies in software engineering: An update," *Information and Software Technology*, vol. 64, pp. 1–18, Aug. 2015. [Online]. Available: http://dx.doi.org/10.1016/j.infsof.2015.03.007

[10] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson, "Systematic mapping studies in software engineering," in *12th International Conference on Evaluation and Assessment in Software Engineering (EASE '08)*. BCS Learning & Development, Jun. 2008, pp. 1–10. [Online]. Available: https://doi.org/10.14236/ewic/ease2008.8

[11] B. Kitchenham, "What's up with software metrics? - a preliminary mapping study," *Journal of Systems and Software*, vol. 83, no. 1, pp. 37–51, Jan. 2010. [Online]. Available: http://dx.doi.org/10.1016/j.jss.2009.06.041

[12] B. Kitchenham, R. Pretorius, D. Budgen, O. P. Brereton, M. Turner, M. Niazi, and S. Linkman, "Systematic literature reviews in software engineering - a tertiary study," *Information and Software Technology*, vol. 52, no. 8, pp. 792–805, Aug. 2010. [Online]. Available: http://dx.doi.org/10.1016/j.infsof.2010.03.006

[13] A. A. Yavuz and P. Ning, "BAF: An efficient publicly verifiable secure audit logging scheme for distributed systems," in *2009 Annual Computer Security Applications Conference*. IEEE, 2009, pp. 219–228.

[14] D. Kim, E. Hwang, and S. Rho, "Multi-camera-based security log management scheme for smart surveillance," *Security and Communication Networks*, vol. 7, no. 10, pp. 1517–1527, 2014.

[15] B. A. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," *Technical Report EBSE 2007-001. Keele University and Durham University Joint Report*, pp. 1–57, Jul. 2007.

[16] B. A. Kitchenham, "Procedures for performing systematic reviews," *Keele, UK, Keele University*, vol. 33, no. 2004, pp. 1–26, Jul. 2004.

[17] H. Zhang, M. A. Babar, and P. Tell, "Identifying relevant studies in software engineering," *Information and Software Technology*, vol. 53, no. 6, pp. 625–637, Jun. 2011. [Online]. Available: https://doi.org/10.1016/j.infsof.2010.12.010

[18] V. Braun and V. Clarke, "Using thematic analysis in psychology," *Qualitative Research in Psychology*, vol. 3, no. 2, pp. 77–101, 2006. [Online]. Available: https://doi.org/10.1191/1478088706qp063oa

[19] J. Saldaña, *The Coding Manual for Qualitative Researchers*. Sage, 2021.

[20] Alibaba. (2017) Alibaba Java coding guidelines. [Online]. Available: https://alibaba.github.io/Alibaba-Java-Coding-Guidelines/

[21] J. Skowronski. (2017, Jan.) 30 best practices for logging at scale. [Online]. Available: https://www.loggly.com/blog/30-best-practices-logging-scale/

[22] L. Tal. (2017, Jan.) 9 logging best practices based on hands-on experience. [Online]. Available: https://www.loomsystems.com/blog/single-post/2017/01/26/9-logging-best-practices-based-on-hands-on-experience

[23] J. Boulon, A. Konwinski, R. Qi, A. Rabkin, E. Yang, and M. Yang, "Chukwa: A large-scale monitoring system," in *Proceedings of CCA*, vol. 8, 2008, pp. 1–5.

[24] G. F. Creţu-Ciocârlie, M. Budiu, and M. Goldszmidt, "Hunting for problems with Artemis," in *Proceedings of the First USENIX conference on Analysis of system logs*. USENIX Association, 2008, pp. 2–2.

[25] H. Mi, H. Wang, Y. Zhou, M. R.-T. Lyu, and H. Cai, "Toward fine-grained, unsupervised, scalable performance diagnosis for production cloud computing systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 6, pp. 1245–1255, Jun. 2013.

[26] A. Aiken, S. Bugrara, I. Dillig, T. Dillig, B. Hackett, and P. Hawkins, "An overview of the Saturn project," in *Proceedings of the 7th ACM SIGPLAN-SIGSOFT workshop on Program analysis for software tools and engineering (PASTE '07)*, Jun. 2007, pp. 43–48. [Online]. Available: https://doi.org/10.1145/1251535.1251543

[27] W. W. Royce, "Managing the development of large software systems: Concepts and techniques," in *Proceedings of the 9th international conference on Software Engineering*, Mar. 1987, pp. 328–338.

[28] C. A. Cois, J. Yankel, and A. Connell, "Modern DevOps: Optimizing software development through effective system interactions," in *2014 IEEE International Professional Communication Conference (IPCC)*. IEEE, oct 2014, pp. 1–7. [Online]. Available: https://doi.org/10.1109%2Fipcc.2014.7020388

[29] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering*. Springer Science & Business Media, Jun. 2012.

[30] J. Cândido, M. Aniche, and A. van Deursen, "Contemporary software monitoring: A systematic literature review," *arXiv preprint arXiv:1912.05878*, Dec. 2019.

[31] Puppet, "The 2021 state of DevOps report," 2021. [Online]. Available: https://puppet.com/resources/report/2021-state-of-devops-report/

[32] W. Xu, L. Huang, A. Fox, D. Patterson, and M. I. Jordan, "Detecting large-scale system problems by mining console logs," in *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles (SOSP' 09)*. New York, NY, USA: ACM, Oct. 2009, pp. 117–132. [Online]. Available: https://doi.org/10.1145/1629575.1629587

[33] D. Yuan, H. Mai, W. Xiong, L. Tan, Y. Zhou, and S. Pasupathy, "SherLog: Error diagnosis by connecting clues from run-time logs," in *Proceedings of the fifteenth edition of ASPLOS on Architectural support for programming languages and operating systems (ASPLOS '10)*. ACM, Mar. 2010, pp. 143–154. [Online]. Available: https://doi.org/10.1145/1736020.1736038

[34] W. Shang, "Bridging the divide between software developers and operators using logs," in *2012 34th International Conference on Software Engineering (ICSE '12)*. IEEE, Jun. 2012, pp. 1583–1586. [Online]. Available: https://doi.org/10.1109/icse.2012.6227031

[35] T. Barik, R. DeLine, S. Drucker, and D. Fisher, "The bones of the system: A case study of logging and telemetry at Microsoft," in *Proceedings of the 38th International Conference on Software Engineering Companion (ICSE-C '16)*. IEEE, May 2016, pp. 92–101. [Online]. Available: https://doi.org/10.1145/2889160.2889231

[36] W. Shang, M. Nagappan, A. E. Hassan, and Z. M. Jiang, "Understanding log lines using development knowledge," in *2014 IEEE International Conference on Software Maintenance and Evolution (ICSME '14)*. IEEE, Sep. 2014, pp. 21–30. [Online]. Available: https://doi.org/10.1109/icsme.2014.24

[37] M. I. H. Sukmana, K. A. Torkura, F. Cheng, C. Meinel, and H. Graupner, "Unified logging system for monitoring multiple cloud storage providers in cloud storage broker," in *2018 International Conference on Information Networking (ICOIN '18)*. IEEE, Jan. 2018, pp. 44–49. [Online]. Available: https://doi.org/10.1109/icoin.2018.8343081

[38] A. Pi, W. Chen, W. Zeller, and X. Zhou, "It can understand the logs, literally," in *2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW '19)*. IEEE, May 2019, pp. 446–451.

[39] D. Schipper, M. Aniche, and A. van Deursen, "Tracing back log data to its log statement: From research to practice," in *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR '19)*. IEEE, May 2019, pp. 545–549.

[40] M. Bartsch and R. Harrison, "An exploratory study of the effect of aspect-oriented programming on maintainability," *Software Quality Journal*, vol. 16, no. 1, pp. 23–44, May 2007. [Online]. Available: https://doi.org/10.1007/s11219-007-9022-7

[41] G. W. Dunlap, S. T. King, S. Cinar, M. A. Basrai, and P. M. Chen, "Revirt: Enabling intrusion analysis through virtual-machine logging and replay," *ACM SIGOPS Operating Systems Review*, vol. 36, no. SI, pp. 211–224, 2002.

[42] C. N. Chong, Z. Peng, and P. H. Hartel, "Secure audit logging with tamper-resistant hardware," in *IFIP International Information Security Conference*. Springer, May 2003, pp. 73–84.

[43] P. Massonet, S. Naqvi, C. Ponsard, J. Latanicki, B. Rochwerger, and M. Villari, "A monitoring and audit logging architecture for data

location compliance in federated cloud infrastructures," in *2011 IEEE International Symposium on Parallel and Distributed Processing Workshops and Phd Forum*. IEEE, May 2011, pp. 1510–1517.

[44] J. King, J. Stallings, M. Riaz, and L. Williams, "To log, or not to log: using heuristics to identify mandatory log events–a controlled experiment," *Empirical Software Engineering*, vol. 22, no. 5, pp. 2684–2717, Oct. 2017.

[45] G. S. Hartman and L. Bass, "Logging events crossing architectural boundaries," in *IFIP Conference on Human-Computer Interaction*. Springer, Sep. 2005, pp. 823–834.

[46] G. Lee, J. Lin, C. Liu, A. Lorek, and D. Ryaboy, "The unified logging infrastructure for data analytics at Twitter," *Proceedings of the VLDB Endowment*, vol. 5, no. 12, pp. 1771–1780, Aug. 2012.

[47] A. Rabkin, W. Xu, A. Wildani, A. Fox, D. Patterson, and R. Katz, "A graphical representation for identifier structure in logs," in *Proceedings of the 2010 workshop on Managing systems via log analysis and machine learning techniques (SLAML '10)*. ACM, Oct. 2010, pp. 3–3.

[48] 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, pp. 1–27, Feb. 2015. [Online]. Available: https://doi.org/10.1007/s10664-013-9274-8

[49] A. Kadav, M. J. Renzelmann, and M. M. Swift, "Tolerating hardware device failures in software," in *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*. ACM, 2009, pp. 59–72.

[50] D. Subhraveti and J. Nieh, "Record and transplay: Partial checkpointing for replay debugging across heterogeneous systems," in *Proceedings of the ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems*. ACM, 2011, pp. 109–120.

[51] K. Veeraraghavan, D. Lee, B. Wester, J. Ouyang, P. M. Chen, J. Flinn, and S. Narayanasamy, "Doubleplay: Parallelizing sequential logging and replay," *ACM Transactions on Computer Systems (TOCS)*, vol. 30, no. 1, p. 3, 2012.

[52] M. Cinque, R. Natella, A. Pecchia, S. Russo, C.-I. Laboratorio, C. Savy, and C. U. M. Sant'Angelo, "Improving ffda of web servers through a rule-based logging approach," in *Proceedings of the 1st International Workshop on Field Failure Data Analysis, Niagara Falls, NY, USA*, 2008.

[53] T. Jia, Y. Li, C. Zhang, W. Xia, J. Jiang, and Y. Liu, "Machine deserves better logging: A log enhancement approach for automatic fault diagnosis," in *2018 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW '18)*, Oct. 2018, pp. 106–111. [Online]. Available: https://doi.org/10.1109/issrew.2018.00-22

[54] C. Zhang, Z. Guo, M. Wu, L. Lu, Y. Fan, J. Zhao, and Z. Zhang, "AutoLog: Facing log redundancy and insufficiency," in *Proceedings of the Second Asia-Pacific Workshop on Systems (APSys '11)*, Jul. 2011, pp. 1–5. [Online]. Available: https://doi.org/10.1145/2103799.2103811

[55] X. Zhao, K. Rodrigues, Y. Luo, M. Stumm, D. Yuan, and Y. Zhou, "The game of twenty questions: Do you know where to log?" in *Proceedings of the 16th Workshop on Hot Topics in Operating Systems (HotOS '17)*. ACM, May 2017, pp. 125–131. [Online]. Available: https://doi.org/10.1145/3102980.3103001

[56] F. Baccanico, G. Carrozza, M. Cinque, D. Cotroneo, A. Pecchia, and A. Savignano, "Event logging in an industrial development process: Practices and reengineering challenges," in *2014 IEEE International Symposium on Software Reliability Engineering Workshops (ISSRE Workshops)*. IEEE, Nov. 2014, pp. 10–13. [Online]. Available: https://doi.org/10.1109/issrew.2014.69

[57] B. Chen, "Improving the software logging practices in DevOps," in *2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*. IEEE, May 2019, pp. 194–197. [Online]. Available: https://doi.org/10.1109/icse-companion.2019.00080

APPENDIX A SELECTED PRIMARY STUDIES

[Anu 19] H. Anu, J. Chen, W. Shi, J. Hou, B. Liang, and B. Qin. "An Approach to Recommendation of Verbosity Log Levels Based on Logging Intention". In: *2019 IEEE International Conference on Software Maintenance and Evolution (ICSME '19)*, pp. 125–134, IEEE, Sep. 2019.

[Chen 17a] B. Chen and Z. M. Jiang. "Characterizing and Detecting Anti-Patterns in the Logging Code". In: *2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE '17)*, pp. 71–81, IEEE, May 2017.

[Chen 17b] 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, pp. 330–374, Feb. 2017.

[Chen 19] B. Chen and Z. M. Jiang. "Extracting and Studying the Logging-Code-Issue-Introducing Changes in Java-Based Large-Scale Open Source Software Systems". *Empirical Software Engineering*, Vol. 24, No. 4, pp. 2285–2322, Aug. 2019.

[Chen 20] B. Chen and Z. M. J. Jiang. "Studying the Use of Java Logging Utilities in the Wild". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering (ICSE '20)*, pp. 397–408, Oct. 2020.

[Chow 18] S. Chowdhury, S. D. Nardo, A. Hindle, and Z. M. Jiang. "An Exploratory Study on Assessing the Energy Impact of Logging on Android Applications". *Empirical Software Engineering*, Vol. 23, No. 3, pp. 1422–1456, June 2018.

[Cinq 09] M. Cinque, D. Cotroneo, and A. Pecchia. "A Logging Approach for Effective Dependability Evaluation of Complex Systems". In: *2009 Second International Conference on Dependability (DEPEND '09)*, pp. 105–110, IEEE, June 2009.

[Cinq 10] M. Cinque, D. Cotroneo, R. Natella, and A. Pecchia. "Assessing and Improving the Effectiveness of Logs for the Analysis of Software Faults". In: *2010 IEEE/IFIP International Conference on Dependable Systems & Networks (DSN '10)*, pp. 457–466, IEEE, June 2010.

[Cinq 12] M. Cinque, D. Cotroneo, and A. Pecchia. "Event Logs for the Analysis of Software Failures: A Rule-Based Approach". *IEEE Transactions on Software Engineering*, Vol. 39, No. 6, pp. 806–821, Oct. 2012.

[Cinq 20] M. Cinque, R. D. Corte, and A. Pecchia. "An empirical analysis of error propagation in critical software systems". *Empirical Software Engineering*, Vol. 25, No. 4, pp. 2450–2484, March 2020.

[Ding 15] R. Ding, H. Zhou, J.-G. Lou, H. Zhang, Q. Lin, Q. Fu, D. Zhang, and T. Xie. "Log2: A Cost-Aware Logging Mechanism for Performance Diagnosis". In: *2015 USENIX Annual Technical Conference (USENIX ATC '15)*, pp. 139–150, USENIX Association, Santa Clara, CA, July 2015.

[Fu 14] 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 '14)*, pp. 24–33, ACM, New York, NY, USA, May 2014.

[Ghol 20] S. Gholamian and P. A. S. Ward. "Logging Statements' Prediction Based on Source Code Clones". In: *Proceedings of the 35th Annual ACM Symposium on Applied Computing (SAC '20)*, pp. 82–91, ACM, March 2020.

[Hass 18] M. Hassani, W. Shang, E. Shihab, and N. Tsantalis. "Studying and Detecting Log-Related Issues". *Empirical Software Engineering*, Vol. 23, No. 6, pp. 3248–3280, March 2018.

[He 18] P. He, Z. Chen, S. He, and M. R. Lyu. "Characterizing the Natural Language Descriptions in Software Logging Statements". In: *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering (ASE '18)*, pp. 178–189, ACM, Sep. 2018.

[Jia 18] Z. Jia, S. Li, X. Liu, X. Liao, and Y. Liu. "SMARTLOG: Place Error Log Statement by Deep Understanding of Log Intention". In: *2018 IEEE 25th International Conference on Software Analysis, Evolution and Reengineering (SANER '18)*, pp. 61–71, IEEE, March 2018.

[Kabi 16a] 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 Workshop on Mining Software Repositories (MSR '16)*, pp. 154–164, ACM, May 2016.

[Kabi 16b] S. Kabinna, W. Shang, C.-P. Bezemer, and A. E. Hassan. "Examining the Stability of Logging Statements". In: *2016*

IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER '16), pp. 326–337, IEEE, March 2016.

[Kim 19] T. Kim, S. Kim, S. Park, and Y. Park. "Automatic Recommendation to Appropriate Log Levels". *Software: Practice and Experience*, Vol. 50, No. 3, pp. 189–209, Nov. 2019.

[Kubo 20] T. Kubota, N. Aota, and K. Kono. "Logging Inter-Thread Data Dependencies in Linux Kernel". *IEICE Transactions on Information and Systems*, Vol. E103.D, No. 7, pp. 1633–1646, July 2020.

[Lal 15] 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 '15)*, Vol. 6, No. 1, pp. 49–73, Jan. 2015.

[Lal 16a] 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 '16)*, Vol. 7, No. 2, pp. 43–71, Apr. 2016.

[Lal 16b] S. Lal, N. Sardana, and A. Sureka. "LogOptPlus: Learning to Optimize Logging in Catch and If Programming Constructs". In: *2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC '16)*, pp. 215–220, IEEE, June 2016.

[Lal 16c] 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 '16)*, pp. 151–155, ACM, Feb. 2016.

[Lal 17] 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, pp. 7–38, 2017.

[Lal 19] S. Lal, N. Sardana, and A. Sureka. "Three-Level Learning for Improving Cross-Project Logging Prediction for If-Blocks". *Journal of King Saud University - Computer and Information Sciences*, Vol. 31, No. 4, pp. 481–496, Oct. 2019.

[Li 17a] 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, pp. 1684–1716, Aug. 2017.

[Li 17b] 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, pp. 1831–1865, Aug. 2017.

[Li 18a] H. Li, T.-H. P. Chen, W. Shang, and A. E. Hassan. "Studying Software Logging Using Topic Models". *Empirical Software Engineering*, Vol. 23, No. 5, pp. 2655–2694, Oct. 2018.

[Li 18b] S. Li, X. Niu, Z. Jia, J. Wang, H. He, and T. Wang. "LogTracker: Learning Log Revision Behaviors Proactively from Software Evolution History". In: *Proceedings of the 26th Conference on Program Comprehension (ICPC '18)*, pp. 178–188, ACM, May 2018.

[Li 19] Z. Li, T.-H. Chen, J. Yang, and W. Shang. "DLFinder: Characterizing and Detecting Duplicate Logging Code Smells". In: *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE '19)*, pp. 152–163, IEEE, May 2019.

[Li 20a] H. Li, W. Shang, B. Adams, M. Sayagh, and A. E. Hassan. "A Qualitative Study of the Benefits and Costs of Logging From Developers' Perspectives". *IEEE Transactions on Software Engineering*, pp. 1–1, Jan. 2020.

[Li 20b] Z. Li, T.-H. Chen, and W. Shang. "Where Shall We Log? Studying and Suggesting Logging Locations in Code Blocks". In: *Proceedings of the 35th International Conference on Automated Software Engineering (ASE '20)*, pp. 361–372, IEEE, Sep. 2020.

[Liu 19] Z. Liu, X. Xia, D. Lo, Z. Xing, A. E. Hassan, and S. Li. "Which Variables Should I Log?". *IEEE Transactions on Software Engineering*, Sep. 2019.

[Liu 20] X. Liu, T. Jia, Y. Li, H. Yu, Y. Yue, and C. Hou. "Automatically Generating Descriptive Texts in Logging Statements: How Far Are We?". In: *Asian Symposium on Programming Languages and Systems*, pp. 251–269, Springer, Nov. 2020.

[Luo 18] L. Luo, S. Nath, L. R. Sivalingam, M. Musuvathi, and L. Ceze. "Troubleshooting Transiently-Recurring Errors in Production Systems with Blame-Proportional Logging". In: *2018 USENIX Annual Technical Conference (USENIX ATC '18)*, pp. 321–334, USENIX Association, July 2018.

[Marr 18] M. Marron. "Log++ Logging for a Cloud-Native World". In: *Proceedings of the 14th ACM SIGPLAN International Symposium on Dynamic Languages (DLS '18)*, pp. 25–36, ACM, Oct. 2018.

[Mizo 19] T. Mizouchi, K. Shimari, T. Ishio, and K. Inoue. "PADLA: A Dynamic Log Level Adapter Using Online Phase Detection". In: *2019 IEEE/ACM 27th International Conference on Program Comprehension (ICPC '19)*, pp. 135–138, IEEE, May 2019.

[Pecc 12] A. Pecchia and S. Russo. "Detection of Software Failures through Event Logs: An Experimental Study". In: *2012 IEEE 23rd International Symposium on Software Reliability Engineering (ISSRE '12)*, pp. 31–40, IEEE, Nov. 2012.

[Pecc 15] A. Pecchia, M. Cinque, G. Carrozza, and D. Cotroneo. "Industry Practices and Event Logging: Assessment of a Critical Software Development Process". In: *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering (ICSE '15)*, pp. 169–178, IEEE, May 2015.

[Rong 18] G. Rong, S. Gu, H. Zhang, D. Shao, and WanggenLiu. "How Is Logging Practice Implemented in Open Source Software Projects? A Preliminary Exploration". In: *2018 25th Australasian Software Engineering Conference (ASWEC '18)*, pp. 171–180, IEEE, Nov. 2018.

[Rong 20] G. Rong, Y. Xu, S. Gu, H. Zhang, and D. Shao. "Can You Capture Information As You Intend To? A Case Study on Logging Practice in Industry". In: *2020 IEEE International Conference on Software Maintenance and Evolution (ICSME '20)*, pp. 12–22, Sep. 2020.

[Sain 16] S. Saini, N. Sardana, and S. Lal. "Logger4u: Predicting Debugging Statements in the Source Code". In: *2016 Ninth International Conference on Contemporary Computing (IC3 '16)*, pp. 1–7, IEEE, Aug. 2016.

[Salf 04] F. Salfner, S. Tschirpke, and M. Malek. "Comprehensive Logfiles for Autonomic Systems". In: *Proceedings of the 18th International Parallel and Distributed Processing Symposium (IPDPS '04)*, pp. 211–219, IEEE, Apr. 2004.

[Shan 14] W. Shang, Z. M. Jiang, B. Adams, A. E. Hassan, M. W. Godfrey, M. Nasser, and P. Flora. "An Exploratory Study of the Evolution of Communicated Information About the Execution of Large Software Systems". *Journal of Software: Evolution and Process*, Vol. 26, No. 1, pp. 3–26, Feb. 2014.

[Tova 13] D. Tovarňák, A. Vašeková, S. Novák, and T. Pitner. "Structured and Interoperable Logging for the Cloud Computing Era: The Pitfalls and Benefits". In: *2013 IEEE/ACM 6th International Conference on Utility and Cloud Computing (UCC '13)*, pp. 91–98, IEEE, Dec. 2013.

[Yao 18] K. Yao, G. B. de Pádua, W. Shang, S. Sporea, A. Toma, and S. Sajedi. "Log4Perf: Suggesting Logging Locations for Web-Based Systems' Performance Monitoring". In: *Proceedings of the 2018 ACM/SPEC International Conference on Performance Engineering (ICPE '18)*, pp. 127–138, ACM, March 2018.

[Yuan 12a] D. Yuan, S. Park, P. Huang, Y. Liu, M. M.-J. Lee, X. Tang, Y. Zhou, and S. Savage. "Be Conservative: Enhancing Failure Diagnosis with Proactive Logging". In: *Presented as part of the 10th USENIX Symposium on Operating Systems Design and Implementation (OSDI '12)*, pp. 293–306, Oct. 2012.

[Yuan 12b] D. Yuan, S. Park, and Y. Zhou. "Characterizing Logging Practices in Open-Source Software". In: *2012 34th International Conference on Software Engineering (ICSE '12)*, pp. 102–112, IEEE, June 2012.

[Yuan 12c] D. Yuan, J. Zheng, S. Park, Y. Zhou, and S. Savage. "Improving Software Diagnosability via Log Enhancement". *ACM Transactions on Computer Systems*, Vol. 30, No. 1, p. 4, Feb. 2012.

[Zeng 19] Y. Zeng, J. Chen, W. Shang, and T.-H. P. Chen. "Studying the Characteristics of Logging Practices in Mobile Apps: A Case Study on F-Droid". *Empirical Software Engineering*, Vol. 24, No. 6, pp. 3394–3434, Feb. 2019.

[Zhao 17] X. Zhao, K. Rodrigues, Y. Luo, M. Stumm, D. Yuan, and Y. Zhou. "Log20: Fully Automated Optimal Placement of Log Printing Statements Under Specified Overhead Threshold". In: *Proceedings of the 26th Symposium on Operating Systems Principles (SOSP '17)*, pp. 565–581, ACM, Oct. 2017.

[Zhi 19] C. Zhi, J. Yin, S. Deng, M. Ye, M. Fu, and T. Xie. "An Exploratory Study of Logging Configuration Practice in

Java”. In: *2019 IEEE International Conference on Software Maintenance and Evolution (ICSME ’19)*, pp. 459–469, IEEE, Sep. 2019.

[Zhi 20] C. Zhi, J. Yin, J. Han, and S. Deng. “A Preliminary Study on Sensitive Information Exposure Through Logging”. In: *2020 27th Asia-Pacific Software Engineering Conference (APSEC ’20)*, pp. 470–474, IEEE, Dec. 2020.

[Zhou 20] R. Zhou, M. Hamdaqa, H. Cai, and A. Hamou-Lhadj. “MobiLogLeak: A Preliminary Study on Data Leakage Caused by Poor Logging Practices”. In: *2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering (SANER ’20)*, pp. 577–581, Feb. 2020.

[Zhu 15] J. Zhu, P. He, Q. Fu, H. Zhang, M. R. Lyu, and D. Zhang. “Learning to Log: Helping Developers Make Informed Logging Decisions”. In: *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering (ICSE ’15)*, pp. 415–425, IEEE, May 2015.

Appendix B
Explanation of the typically excluded papers

First, we did not include studies that focus on log analysis or the usage of log messages, which have been excluded according to E1 listed in Table 2.

For example, Xu et al. [32] proposed a general methodology to mine and analyze console logs to automatically detect system runtime problems. Yuan et al. [33] designed and implemented an effective diagnosis technique, which can analyze logs from a failed production run and source code to automatically generate useful information that assist engineers in diagnosing errors. Similarly, other studies like [34], [35], [36], [37], [38], [39], [40] focusing on this topic have been excluded from our study.

Second, we did not include studies that focus on the technologies that log and analyze the behaviors of users rather than software systems, which meet the exclusion criterion E2 listed in Table 2. For example, many studies on audit logs are outside the scope of our research because audit logs typically record the behavior of users. The purpose of such research is usually to ensure the security of the system [41], [42], [43]. King et al. [44] evaluated the use of a heuristics-driven method for identifying mandatory log events to support security analysts in performing forensic analysis. Similarly, some other studies have focused on the recording of user behaviors [45], [46] have been removed from the final study list.

Third, we did not include studies that did not explicitly discuss logging practices, which have been excluded according to exclusion criterion E3 listed in Table 2. For example, Rabkin et al. [47] described an abstract graphical representation of console logs called the identifier graph and a visualization based on this representation. Shang et al. [48] studied the relationship between logs and code quality, and the main focus of this paper is on the code quality rather than on logging practice. Some other studies [49], [50], [51] are excluded for similar reasons.

Last but not least, we did not include studies that are workshop papers, position papers, or ongoing work, which have been excluded according to exclusion criterion E3–5 in Table 2. The typical examples for workshop papers are [52], [53], [54]. The typical examples for position papers or ongoing work are [55], [56], [57].

Shenghui Gu received the BSc degree from Nanjing University, China. He is currently working toward the PhD degree in the Software Institute, Nanjing University, China. His research interests are in software engineering, particularly in AIOps, software log analytics, DevOps, as well as empirical and evidence-based software engineering.

Guoping Rong received the BSc degree in computer science and technology, MSc degree in software theory and PhD degree in applied software engineering, all from Nanjing University. He now is a faculty member with the Software Institute, Nanjing University and the director of the joint laboratory of Nanjing University and Transwarp on data technology. His research area includes software process, DevOps, AIOps and empirical methodology, etc.

He Zhang is a Full Professor of Software Engineering and the Director of DevOps+ Research Laboratory at the Nanjing University, China, also a Principal Scientist with CSIRO, Australia. He undertakes research in software engineering, in particular software & systems process, software architecture, DevOps, software security, blockchain-oriented software engineering, empirical and evidence-based software engineering. He has published over 160 peer-reviewed papers in high quality international conferences and journals, and won 11 Best/Distinguished Paper Awards from several prestigious international conferences and journals in software engineering community.

Haifeng Shen is an associate professor, head of discipline of information technology, and director of the HilstLab in the Faculty of Law and Business, Australian Catholic University. His research expertise is interdisciplinary and revolves around human-centred artificial intelligence and software technologies, which is uniquely positioned at the intersection of human computer interaction, software engineering, and artificial intelligence with a unique focus on ‘interaction’ and ‘integration’: human-AI interaction, human-software interaction, and integration of AI and software.