Empirical Software Engineering (2023) 28:35
https://doi.org/10.1007/s10664-022-10269-5

What is asked about technical debt (TD) on Stack Exchange question-and-answer (Q&A) websites? An observational study

Reem Alfayez[1] · Yunyan Ding[2] · Robert Winn[2] · Ghaida Alfayez[1] · Christopher Harman[2] · Barry Boehm[2]

Accepted: 5 December 2022 / Published online: 28 January 2023

Abstract
Technical debt (TD) is a term coined by agile software pioneer Ward Cunningham to account for the added software system effort or cost resulting from taking early software project shortcuts. Previous research on TD has extensively outlined and discussed the various consequences derived from accumulating TD and the difficulty in managing it. A review of the software engineering literature revealed that Stack Exchange question-and-answer (Q&A) websites can provide valuable, real world perspectives on a number of software engineering topics. Therefore, this study aims to observe how the TD term is utilized on Stack Exchange Q&A websites. Specifically, this study utilizes a dataset derived from three Stack Exchange Q&A websites, which are Stack Overflow (SO), Software Engineering (SE), and Project Management (PM), to retrieve and analyze 578 TD-related questions. The results unveiled that TD-related questions can be categorized into 14 different categories, a total of 636 unique tags are utilized in the acquired set of TD-related questions, and a few TD-related categories both lack accepted answers and have a longer median time to receive an accepted answer than other categories. This study's findings highlight the TD-related challenges that are addressed by Stack Exchange Q&A website users, which may prove beneficial in steering future TD-related efforts.

Keywords Technical debt · Technical debt management · Software engineering · Question-and-answer (Q&A) websites · Stackoverflow

Communicated by: Marco Torchiano

✉ Reem Alfayez
reealfayez@ksu.edu.sa

[1] College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia

[2] Center for Systems and Software Engineering, University of Southern California, Los Angeles, CA USA

1 Introduction

TD is a metaphor first introduced by Ward Cunningham to reflect the added software system cost or effort accrued from taking early software project shortcuts (Kruchten et al. 2012; Cunningham 1992). The metaphor embodies that debt accumulates interest: the later it is paid, the more it costs (Guo and Seaman 2011; Kruchten et al. 2012; Alves et al. 2016). Taking on TD can yield short-term benefits at a consequence of a sub-optimal design or system changes that are difficult to maintain long term (Kruchten et al. 2012; Lenarduzzi et al. 2019; Rios et al. 2020; Cunningham 1992; Suryanarayana et al. 2014). In a select few situations, adding TD can actually be beneficial in the long term and have little to no lasting effects, such as throwaway prototypes. However, individuals may fall into the trap of amassing large amounts of TD, without any plan to manage it, to meet deadlines or push out new software (Kruchten et al. 2012; Rios et al. 2020). One should proceed cautiously when accumulating TD, as large amounts can lead to a significant drop in system quality, increased maintenance costs, and much more (Parnas 1994; Izurieta et al. 2012; Morgenthaler et al. 2012; Codabux and Williams 2013; Spínola et al. 2013; Zazworka et al. 2011b; Rios et al. 2020; Ahmed et al. 2017; de Mello et al. 2017; Suryanarayana et al. 2014; Guo and Seaman 2011; Martini et al. 2014; Nord et al. 2012; Rios et al. 2020).

One of the dangers of TD lies within the fact of how common it is in modern society software systems. Billions of dollars are spent each year to pay off or keep up with the demands of TD (Curtis et al. 2012). Researchers at CAST software conducted a study in which 700 different software systems, submitted by 158 organizations in the US, Europe, and India, were measured for their respective levels of TD and found that, out of a cumulative total of 550 million lines of code (LOC), each LOC, on average, had a TD amount of $3.61 (Curtis et al. 2012). Furthermore, it can be difficult at times to recognize the amount present and cost of TD, as it typically does not have a direct impact on the external behavior of a software system (Fowler 2018; Tom et al. 2013). It is imperative to not delay the management of TD for too long, as it can result in an inflexible and uncontrollable system (Martini et al. 2014; Nord et al. 2012; Rios et al. 2020). Moreover, neglecting the repayment of TD can double or triple the initial cost of said TD (Guo et al. 2011). Ignoring TD in a system not only significantly increases the repayment cost over time, but also can lead to eventual system decay and the need for extremely specialized developers for system maintenance (Izurieta et al. 2012). Unfortunately, the negative consequences of TD extend past the system itself and on to the software practitioners as well. InsighTD, a global industrial surveys project on the causes and effects of TD in which 107 software practitioners participated, revealed that team demotivation was one of the most frequently incurred consequences of TD. The project also found that TD can result in high team turnover, mounting pressures, decreased productivity, and undesirable, though necessary, activities (Rios et al. 2020). Additionally, a jointly published report, which included over 1,000 developers and more than 1,000 C-level executives, by Stripe research, Evans Data Corp, and CIA Factbook found that the average developer spends slightly over 13 hours a week on repaying TD. Over half of the participants believe that their productivity is inhibited by TD, and nearly 80% stated that repaying TD negatively impacts their morale.[1] Additionally, a survey conducted by Stack Overflow (SO) with 56,033 developers in 173 countries found nearly 30% of the subjects consider a fragile code base as their primary challenge in the workplace.[2] Studies like the

[1]https://stripe.com/files/reports/the-developer-coefficient.pdf
[2]https://insights.stackoverflow.com/survey/2016/

ones included above illustrate the dangers of accumulating TD in a software system and why repaying TD is a difficult, though necessary, endeavor.

Though it is essential to avoid the negative consequences of TD, technical debt management (TDM), which can be defined as the process of identifying, acknowledging, measuring, maintaining, and reducing TD, is not a simple or easy feat (Li et al. 2015). Given the typical abundance of TD in a software system, it can be challenging at times to manage accumulated TD present in a given software system (Guo et al. 2016; Fernández-Sánchez et al. 2017; Erdogmus 1999; Biffl et al. 2006; Ampatzoglou et al. 2015; Zazworka et al. 2011a; Suryanarayana et al. 2014). A naive approach to TDM would be attempting to repay all TD present in a system. Unfortunately, the repayment of all TD present in a system is typically deemed infeasible, due to resource constraints present during TD repayment activities (Fernández-Sánchez et al. 2017; Erdogmus 1999; Biffl et al. 2006; Ampatzoglou et al. 2015; Zazworka et al. 2011a; M Bomfim and A Santos 2017). Resource constraints during TD repayment activities stem from the innate nature of TD. Repaying TD does not necessarily have a visible impact on the external behavior of a software system, nor is it guaranteed to have immediate, positive results (Fowler 2018; Tom et al. 2013). Moreover, repaying TD is occasionally a waste of resources, as the given repayment activity may have little to no benefits (Allman 2012). As a result, business and software practitioners alike prefer to allocate available resources on activities that will have a direct, visible impact on the external behavior of a software system, such as adding new features (Tom et al. 2013; Fernández-Sánchez et al. 2017; Zazworka et al. 2011a; M Bomfim and A Santos 2017). This indicates that TD repayment activities will always have a resource constraint in place and be challenging to conduct, which is why TDM is crucial (Fernández-Sánchez et al. 2017; Erdogmus 1999; Biffl et al. 2006; Ampatzoglou et al. 2015; Zazworka et al. 2011a).

The software engineering community has gone to great lengths to understand the current status of TD and TDM (Li et al. 2015; Ampatzoglou et al. 2015; Fernández-Sánchez et al. 2017; Alves et al. 2016; Rios et al. 2018). Fortunately, reviewing the software engineering literature revealed that Stack Exchange Q&A websites can provide valuable, real world perspectives on a number of software engineering topics (Silva et al. 2021; Barua et al. 2014). Specifically, online Q&A websites allow users to ask and answer questions to their peers and others. Examining popular online Q&A websites can provide valuable insights on what problems are faced by users and which questions remain unanswered relating to a given subject matter (Barua et al. 2014; Silva et al. 2021).

This paper presents an observational study to review how the TD term is utilized on Stack Exchange Q&A websites (Yin 2009; Ralph et al. 2020). Acquiring such knowledge may aid the software engineering community in pursuing research on or solutions to the TD-related challenges that Q&A website users are facing. This study began with the collection of 578 TD-related questions from Stack Overflow (SO), Software Engineering (SE), and Project Management (PM) Stack Exchange Q&A websites. By analyzing the compiled set of questions, we found that TD-related questions could be categorized into one or more of the following 14 categories: TD tools, TD repayment, TD prevention, TD communication, TD representation/documentation, TD consequences, TD measurement, TD incurring, TD prioritization, TD management, TD identification, TD definitions, TD monitoring, or Other. Additionally, the tags of each question and category were analyzed; the five most utilized tags were Sonarqube, technical-debt, java, agile, and scrum. The technical-debt tag was also found to be within the top 10 most frequently utilized tag of each of the previously identified TD-related categories. Finally, the remaining findings of the study revolve around the fact that a few TD-related categories lack accepted answers and have higher median wait times to receive an accepted answer when viewed in comparison to the other categories. In particular, the results revealed that questions categorized under TD consequences, TD incurring, and TD tools were the most difficult questions for Stack Exchange Q&A website users, as they were all within the top four of categories when reviewing the category statistics of the percentage of questions without an accepted answer and average highest median time to receive an accepted answer.

The remaining parts of this paper are organized as follows: Section 2 provides background information for the study. Section 3 presents the research settings and setup. Section 4 details the study's results. Section 5 discusses the results and Section 6 discloses the implications of the results. Section 7 addresses potential threats to the study's validity. Section 8 provides an overview on related studies. Lastly, Section 9 concludes the study.

2 Background

This section provides background information for this study. Specifically, technical debt management (TDM) is summarized below with an aim to make this study self-contained.

TDM is the process of acknowledging, identifying, measuring, and reducing TD, which includes processes, techniques, and tools. It is essential to the evolution of a software system and should be conducted collaboratively between technical and business teams to ensure the addressing and balancing of both of their goals (Sterling 2010; Rubin 2012). Li et al. (2015) conducted an extensive study on TDM, where the authors mapped a total of 49 primary studies and concluded that TDM can be described into eight distinctive activities. The said activities are presented in Fig. 1 and defined below, based on the definitions of Li et al. (2015).

The identification of TD entails the detection of TD, which may be caused by intentional or unintentional decisions during the software life cycle through the means of specific techniques, such as human or automated approaches. Measurement of TD revolves around the quantification of the cost and benefit of known TD items in a given system. Representation/documentation involves providing a set of procedures to represent and codify TD in a uniform manner. TD communication is making all identified TD items and their respective consequences visible to all stakeholders, where the TD items can be discussed and managed further. TD prioritization is the process of deciding which TD items are to be repaid first in a given repayment activity and which are to be delayed. Repayment of TD is the resolution or mitigation of TD in a given software system by the use of specific techniques, such as reengineering and refactoring. The monitoring of TD encompasses the observations of variations in cost and benefit of unresolved TD over time. Lastly, prevention of TD focuses on steps that software practitioners can take to prevent potential and future TD from being incurred (Li et al. 2015).

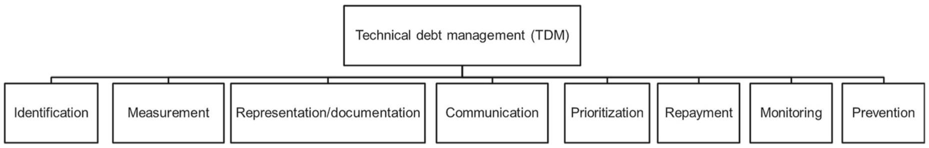


Fig. 1 TDM activities

3 Research Settings

This section presents the setup of the study: selected research questions, sources of data, utilized data collection procedure, and a data preliminary descriptive analysis.

3.1 Research Questions

This study addresses three research questions that we formulated to achieve the goal of the study: to observe how the TD term is utilized on Stack Exchange Q&A websites.[3]

3.1.1 Identified TD-related Categories

As stated previously, not only can TD substantially, negatively impact a given software system, but the consequences of TD also extend towards being a detriment towards entire organizations, software team morale and productivity, customer satisfaction, and more (Behutiye et al. 2017; Li et al. 2015; Rios et al. 2020). The impact of TD, that of which can be both positive and negative, extend towards the entire software development life cycle (Li et al. 2015; Behutiye et al. 2017). Furthermore, TDM can be a challenging task that has a number of distinct activities that each has many aspects to take into account (Li et al. 2015). With the previous information in mind and in conjunction with this study's primary aim being to observe how the term TD is utilized on Stack Exchange Q&A websites, it may prove beneficial to accrue more knowledge on what specific aspects of TD are currently being addressed on Stack Exchange Q&A websites, which can provide a real world view of the TD topics that users are discussing (Silva et al. 2021; Barua et al. 2014; Rosen and Shihab 2016). This leads the study's first research question being the following:
RQ1: What TD-related categories are addressed on Stack Exchange Q&A websites?

3.1.2 Associated Tags

When one posts a question on a Stack Exchange Q&A website, at least one tag and up to five tags must be utilized for each asked question. A tag is considered a label or keyword that aims to categorize similar questions based on their relevance to a topic. Tags can be utilized to facilitate and ease the process of searching for questions related to a given topic or browsing similar questions (Rosen and Shihab 2016). Given that TD can encompass a broad set of topics and that this study's primary objective is to observe how the TD term is utilized on Stack Exchange Q&A websites (Avgeriou et al. 2016; Rios et al. 2018; Li et al. 2015), this study aims to identify the tags that users utilize when asking a TD-related question on Stack Exchange Q&A websites and the tags that were utilized within each of the TD-related categories. Thus, this study's second research question is: ***RQ2: What tags are used in TD-related questions and specific TD-related categories on Stack Exchange Q&A websites?***

3.1.3 Difficult TD-related Categories

The first research question aims to identify the TD-related categories that are addressed on the selected Q&A websites and pinpoints the most frequently addressed ones. However, it

[3]stackexchange.com

fails to note the level of difficulty of each identified category and how challenging the questions in the said categories are. Having such knowledge can aid in gaining a more holistic understanding, with regards to the state of the identified TD-related categories (Rosen and Shihab 2016; Haque et al. 2020). Therefore, this study seeks to better understand whether there exists more difficult TD-related categories than others and, if so, identify such categories, which leads the third research question being: ***RQ3: What TD-related categories are the most difficult?***

3.2 Research Data

This section summarizes the study's utilized data sources, data collection procedure, and a data preliminary descriptive analysis.

3.2.1 Data Sources

We derived the dataset of this study from three Stack Exchange network Q&A communities websites: Stack Overflow (SO),[4] Software Engineering (SE),[5] and Project Management (PM).[6] An overview of the Stack Exchange network and each included website is included below. One should note that the included overviews are obtained from each individual website's meta.

The Stack Exchange network is comprised of 177 Q&A websites that focus on an assortment of topics in various fields. Each website is dedicated to a specific topic, though all websites are based on the same concept: receiving answers to one's questions. On each website, a user can ask a question and the said question will be answered by other users, who can be experts or amateurs. Questions must include at least one tag, which is a label or keyword that categorizes a question with other similar questions. Moreover, a question can also have up to five tags; it can be related to many subjects. Including tags to questions aids in facilitating the search process. Additionally, a user can comment on a question or an answer. Questions, answers, and comments can be upvoted or downvoted by other users based on their usefulness and appropriateness, and the number of votes determine which questions, answers, or comments appear first. The websites are self-moderated through a reputation system. The reputation system grants users rewards based on their respective questions, answers, and comments. The reputation score of a user increases when other users upvote on the initial user's questions, comments, answers, and edits. A user's reputation can also decrease when other users downvote their content.

Stack Overflow (SO) is a public platform that was established in 2008 to serve programmers, both professional and amateur, to find and contribute answers to technical challenges. The website consists of a collection of coding questions and answers, and users can ask questions related to a specific programming problem, software algorithm, software tool, software development problem, and more. The website is the oldest, most popular, and the largest Stack Exchange community website. As of May 30, 2021, the website has over 14,839,627 users, 21,286,479 questions, 80,673,644 comments, and 31,692,495 answers.

[4]stackoverflow.com
[5]softwareengineering.stackexchange.com
[6]pm.stackexchange.com

Software Engineering (SE) is a Stack Exchange website specifically for programmers who are interested in discussions on software development. Programmers can include, but are not limited to, academics, professionals, and students working within the systems development life cycle. The website's interests span over topics related to all aspects of the systems development life cycle, such as software development methods and practices, software requirements, software architecture, software design, software quality assurance, testing, etc. The website was established in 2010 and was initially named programmers until it was renamed to software engineering in 2016. The website has a 332,475 user base who asked 59,351 questions, posted 169,049 answers and 508,246 comments, as of May 30, 2021.

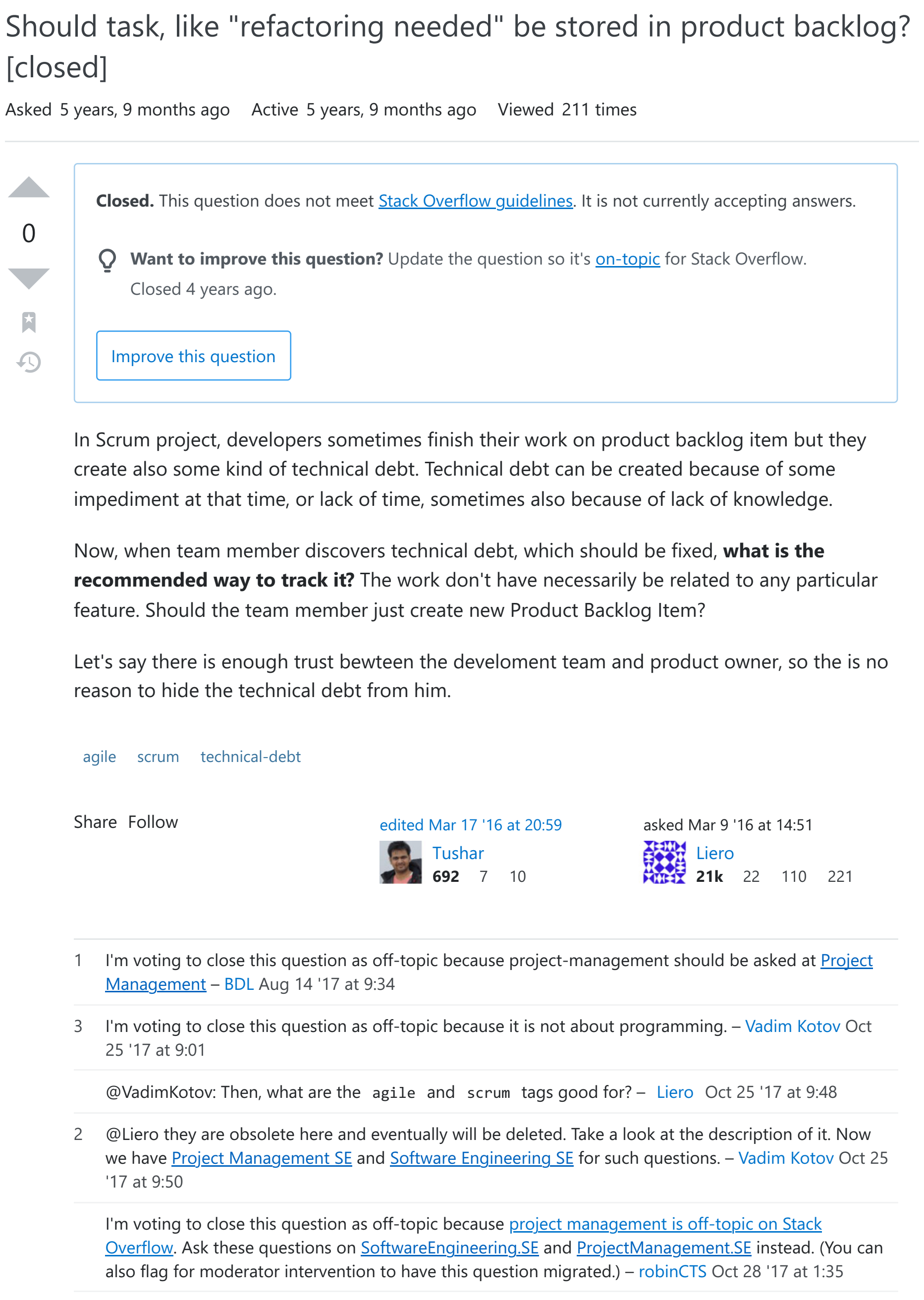
Project Management (PM) is a Stack Exchange website that was established in 2011 and is focused on project management. The website caters towards both project management practitioners and academics. Topics on the website commonly relate to the following: the practice or profession of project management, project management frameworks, how to use tools to solve project management-related problems, and more. As of May 30, 2021, it has attracted 32,245 users who have asked 5,868 questions, provided 17,724 answers, and have left 25,466 comments.

3.2.2 Data Collection Procedure

Exploratory Search Prior to designing the data collection procedure, we explored the SO website by searching for the term "technical debt". The search resulted in questions that contained the term "technical debt" in their body or title. Reviewing a few of these retrieved questions resulted in three observations that influenced the study's data collection procedure: **First,** while Stack Exchange requires its users to tag their questions with at least one tag to facilitate the searching and browsing of said questions, it was realized that not all retrieved TD-related questions utilize the technical-debt tag; users have instead made use of multiple other tags to refer to TD-related questions. As a result, if one wishes to retrieve the majority of TD-related questions, they need to search for "technical debt" in the title or body of each question. **Second,** not all users utilize the "technical debt" term to express TD. A few users used one or more of the following terms: tech debt, technology debt, or more. This observation highly influenced the design of the study's search string and data extraction procedure, which is explained in more detail below. **Third,** while the search returned around 500 questions, a few of the questions related to TD were found to be closed, as they were no longer considered to be "on topic" on SO. These questions were suggested instead to be posted on SE or PM. An example of a closed TD-related question is displayed in Fig. 2. Consequently, we expanded the study's data sources to include questions from PM and SE websites as well.

Data Extraction We initially made use of the Stack Exchange Data Explorer tool,[7] which is an open-source software (OSS) tool that grants its users the ability to query the Stack Exchange network's data. The tool is updated weekly to continually include the most recent questions added to the network. A query of the term "technical debt" was first made to find any question that contained the term in its title. However, after a few attempts, we found that the tool does not properly retrieve all of the data that it should, due to built-in time limitations and caching issues. As a result, the authors made the decision to instead

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



Should task, like "refactoring needed" be stored in product backlog? [closed]

Asked 5 years, 9 months ago Active 5 years, 9 months ago Viewed 211 times

0

Closed. This question does not meet Stack Overflow guidelines. It is not currently accepting answers.

Want to improve this question? Update the question so it's on-topic for Stack Overflow.

Closed 4 years ago.

Improve this question

In Scrum project, developers sometimes finish their work on product backlog item but they create also some kind of technical debt. Technical debt can be created because of some impediment at that time, or lack of time, sometimes also because of lack of knowledge.

Now, when team member discovers technical debt, which should be fixed, **what is the recommended way to track it?** The work don't have necessarily be related to any particular feature. Should the team member just create new Product Backlog Item?

Let's say there is enough trust bewteen the develoment team and product owner, so the is no reason to hide the technical debt from him.

agile scrum technical-debt

Share Follow

edited Mar 17 '16 at 20:59
Tushar
692 7 10

asked Mar 9 '16 at 14:51
Liero
21k 22 110 221

1 I'm voting to close this question as off-topic because project-management should be asked at Project Management – BDL Aug 14 '17 at 9:34

3 I'm voting to close this question as off-topic because it is not about programming. – Vadim Kotov Oct 25 '17 at 9:01

@VadimKotov: Then, what are the `agile` and `scrum` tags good for? – Liero Oct 25 '17 at 9:48

2 @Liero they are obsolete here and eventually will be deleted. Take a look at the description of it. Now we have Project Management SE and Software Engineering SE for such questions. – Vadim Kotov Oct 25 '17 at 9:50

I'm voting to close this question as off-topic because project management is off-topic on Stack Overflow. Ask these questions on SoftwareEngineering.SE and ProjectManagement.SE instead. (You can also flag for moderator intervention to have this question migrated.) – robinCTS Oct 28 '17 at 1:35

Fig. 2 An example of a closed TD-related question from SO

download the Stack Exchange Data Dump.[8] The dump consists of XML files that contain anonymized data of all content contributed by users on the Stack Exchange websites. The downloaded dump utilized in this study includes data up until May 30, 2021.

When analyzing data, it is imperative to be meticulous when selecting a search string, as the quality of the results are highly dependable on the quality of the retrieved data. To ensure the quality of the study's dataset and produce high quality results, we followed a subjective approach to select a search string (Kitchenham et al. 2015). We then conducted trial experiments to derive the string, with all experiments searching for questions that included a given search string in their body, title, or tags. Moreover, we altered the search string in each experiment until a decision was reached. The experiments began using the search string "debt". Unfortunately, the search string "debt" produced an overwhelming number of results, and the majority of these results reference "debt" in a financial or economic context. Consequently, we tailored the search string to the term "technical". However, the search also resulted in an enormous number of results, with most of the results not being TD-related.

We then decided to look for any question that contains the terms "tech" and "debt" in its title, body, or tag in any order. This resulted in 785 questions, as presented in Table 1, where column "S1" refers to the number of questions resulted from retrieving any question that contains the terms "tech" and "debt" in its title, body, or tag. Following an inspection of the results of the search, we noticed that the majority of the results are TD-related, though a few unrelated questions still remained. Given that manually checking the results for unrelated questions requires a huge effort, we conudcted an additional search for any questions that contained the term "technical_debt" or "tech_debt", where "_" represents any character. This last search resulted in 572 questions, as depicted in column "S2" in Table 1.

We then compared the two searches mentioned above using Algorithm 1. The algorithm subtracts the set of questions that resulted from search S2 from the questions that resulted from search S1 and returns the questions that were retrieved by S1 but not retrieved by S2. It should be noted that questions resulting from S2 but not in S1 were discarded.

```
  Input: S1_Q: list of questions resulting from S1 search
         S2_Q: list of questions resulting from S2 search
  Output: Diff_Q: list of questions that results from S1 search, but not in S2 search
results
 Diff_Q = []
 for Question in S1_Q do
     if Question Not in S2_Q then
         Diff_Q.append(Question)
     end if
 end for
 return Diff_Q
```

Algorithm 1 Finding different questions

Once the algorithmic comparison had been completed, two authors independently reviewed the resulting set's questions and categorized said questions as TD-related and non-TD-related, following the guidelines in Kitchenham et al. (2015) and Ralph et al. (2020).

[8]https://archive.org/details/stackexchange

Table 1 Questions obtained from the employed search procedure

Website	S1	S2	Excluded	Final
Stack Overflow (SO)	598	395	202	396
Software Engineering (SE)	145	138	5	140
Project Management (PM)	42	39	0	42
Total	785	572	207	578

Deciding to exclude a question was a simple endeavor, as each of these questions fall under one of the following scenarios: containing a code snippet that has the terms "debt", "tech", or both; encompassing a Uniform Resource Locator(URL) that has both or one of the terms; or containing a thankful statement that contains the term "debt", such as " I would be forever in your debt" statement. When calculating the Cohen's Kappa as advised by Ralph et al. (2020), the categorization step resulted in a Cohen's Kappa coefficient= 1.00, which indicates that the two authors were in complete agreement when categorizing the questions (Landis and Koch 1977).

The review resulted in the exclusion of 207 questions, as presented in Table 1. The total number of included TD-related questions in the final dataset was 578, where 396, 140, 42 were retrieved from SO, SE, and PM, respectively.

3.2.3 Data Preliminary Descriptive Analysis

This section presents a descriptive analysis on the acquired TD-related questions, aiming to provide context to obtained results.

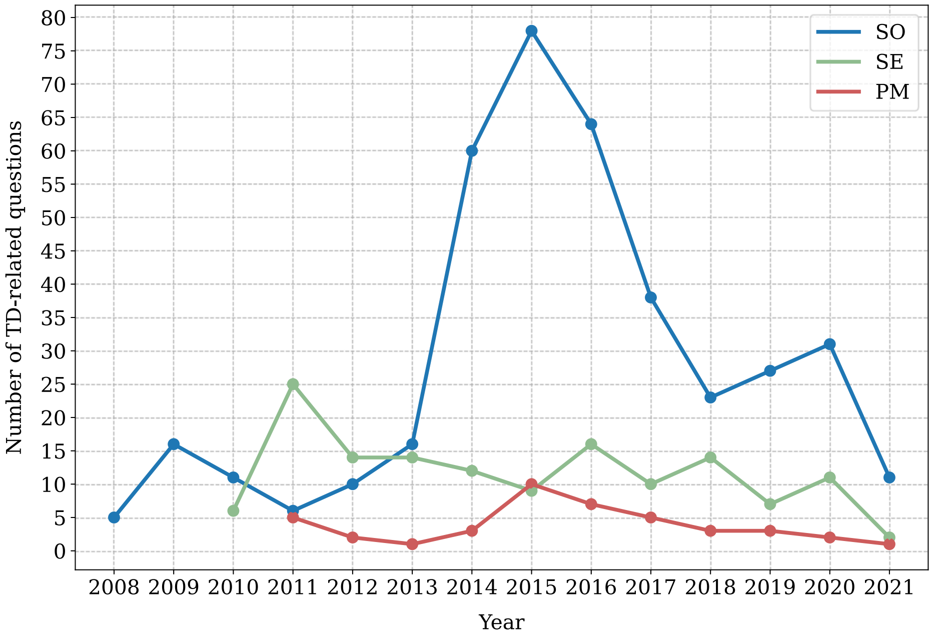


Fig. 3 Distribution of TD-related questions based on year and the included Q&A websites

Users asked 578 TD-related questions on the three included Stack Exchange Q&A websites. SO have the highest number of TD-related questions (396 questions), followed by SE (140 questions) and PM (42 questions). We expected this variation, as SO has been in operation longer than SE and PM, and it contains the largest number of asked questions.

Figure 3 presents the distribution of asked questions relating to TD for each website by year. On SO, TD-related questions were found to be quite lacking until 2015, where the number of asked questions peaked significantly. The peak of the number of questions relating to TD in 2015 is most likely due to SonarQube adding TD as a core feature in the same year. This is further evidenced by the revelation that a large number of TD-related questions included a SonarQube-related tag. In the following years, TD-related questions began to decline until 2018, where it began to increase again, only to drop in 2021. In regards to SE, the number of TD-related questions fluctuated throughout the examined time period, with there only being a peak in 2011. Lastly, the number of TD-related questions on PM remained relatively low since its inception, with a small peak in 2015.

To provide clarity on whether TD-related questions are typically asked by a small number of users that frequently ask or a large, diverse group, we measured the number of unique users who asked at least one TD-related question. In total, 556 unique users asked at least one TD-related question. SO had the largest number (385 unique users), followed by SE (135 unique users) and lastly PM (36 unique users). Additionally, we explored the number of TD-related questions that each user posted. Of the 385 unique users on SO, around 98% only asked one, 2% asked two, and less than 1% asked three TD-related questions. Similarly, on SE, 96% of users who asked a TD-related question only asked one question and close to 4% asked two. PM had nearly 92% of its counted users ask only one TD-related question, where around 3% asked two, three, and four questions, respectively.

TD-related questions are present on all of the three selected Stack Exchange Q&A websites. The number of asked TD-related questions peaked in 2015. Furthermore, the majority of the questions were asked by users who only asked one TD-related question.

4 Results

This section presents the findings of this observational study. The results are based on the aforementioned resultant set of 578 TD-related questions acquired from three Stack Exchange Q&A websites. It should be noted that the results section only includes a high level summary of the data. A more detailed view of the results, which includes statistics on each included Q&A website, is presented on an online appendix in our dataset repository.

4.1 RQ1: What TD-related Categories are Addressed on Stack Exchange Q&A Websites?

Approach To categorize the TD-related questions and summarize their topics, the authors followed the guidelines outlined in Cruzes and Dyba (2011), Ralph et al. (2020), and Kitchenham et al. (2015) and conducted a manual approach to analyze and categorize the TD-related questions. Specifically, we utilized an integrative approach that encompasses a deductive approach and an inductive approach (Cruzes and Dyba 2011). When utilizing a deductive approach, one needs to utilize a start list of categories. For this study, we derived the start list from Li et al. (2015)'s summarized eight TDM activities, described previously in Section 2. On the other hand, when utilizing an inductive approach, one needs to develop a list of categories as they categorize the data. Two authors separately categorized each question and, independently, created new categories if a question did not fit within one of the previously defined eight categories. Subsequently, the two authors held a follow up discussion to compare the results. The two authors initiated the meeting by calculating the Cohen's Kappa coefficient. The calculated statistic was equal to 0.75, which indicates a substantial agreement (Landis and Koch 1977). The two authors then collaboratively reviewed their individual categories together and aimed to combine similar categories together if provided the opportunity. For example, one author categorized a few TD-related questions as TD-acknowledgement, referring to questions that a questioner admits that their proposed solution contains TD. On the contrary, the other author categorized the same questions as TD incurring. Authors resolved such disagreements by discussing what is the most suitable category among the two similar categories to describe the questions and then assigned the given question to the one that was agreed upon. Afterward, the two authors recalculated the Cohen's Kappa coefficient and received a value of 0.89, which indicates an almost perfect agreement (Landis and Koch 1977). The previously two mentioned authors held a second discussion to resolve any remaining category disagreements. Disagreements in categorization were resolved through the retrieval of each question under disagreement from its associated website, reviewing all of its answers and comments, and debating a given categorization until the authors reached a consensus. Eventually, the authors categorized all TD-related questions with full agreement. It should be noted that the categories are not mutually exclusive, meaning that a TD-related question may fall under more than one category.

Results When reviewing the acquired dataset of questions, we realized that Stack Exchange Q&A website users ask questions on a wide variety of TD-related topics, which range from seeking advice on communicating the importance of TD repayment to which tools can be utilized for TDM. We devised a total of 14 unique TD-related categories to accommodate the wide range of asked questions' topics. Each TD-related category is described in more detail below, sorted based on their frequency of occurrence (with exception to Other).

TD Tools The majority of the identified TD-related questions were related to TD tools (213 questions). Questions in this category either ask for a recommendation for a tool to aid in TDM (3 questions) or about a specific tool (210 questions). When reviewing questions that are focused on a specif tool, we found that the majority of these questions ask about SonarQube (91%). The remaining questions focused on tools such as PMD, Checkstyle, FindBugs, NDepend, OpenCover, and landscape.io, though no questions focused on CAST[9] or Kiuwan.[10] Not only was Cast and Kiuwan lacking from TD tool-related questions, but from all included TD-related questions. We did not expect the lack of focus on CAST or Kiuwan, given their popularity and their ability to aid in TDM.

TD Repayment Questions categorized under TD repayment focus on the techniques and associated problems of the TD repayment process (134 questions). The majority of TD-repayment questions (96 questions) focused on TD repayment at the code level aiming to get assistance on how to fix a given piece of code to repay TD, such as "How to handle the

[9] www.castsoftware.com/
[10] https://www.kiuwan.com/

'else' clause when converting an 'if..elif..else' statement into a dictionary lookup?". The remaining questions include seeking advice on how to conduct TD repayment in general by either proposing strategies and asking for opinions on each of these strategies from experienced individuals (19 questions), such as "When rewriting medium to large applications, which of these approaches are useful?", or asking for advice with no proposed specific strategy (19 questions), such as "How to approach legacy code? Desperate case".

TD Prevention TD prevention encompasses 75 questions that focus on strategies, experiences, or methodologies that aid in the prevention of accumulating TD. The majority of questions in this category (50 questions) are focused on specific technical issues in the code level. Typically, users ask a very specific question that proposes a given solution and ask for opinions on its potential effect on TD, such as "What are the consequences of using verbs instead of nouns in REST API URI?". The remaining questions (25 questions) include seeking experiences and specific practices that help in preventing TD from accumulating in general, such as "What are the most effective techniques to stop a codebase from becoming difficult to maintain as it grows?".

TD Communication A total of 51 questions were categorized under TD communication, which is the imparting of knowledge regarding the TD concept and its associating long term detrimental effects to all stakeholders. The majority of questions (25 questions) in this category concentrated on how to convince management or a given client to repay TD and convey the potential negative consequences that may be occurred if the TD was left untouched, such as "Any tips on how to 'sell' the necessity of dealing with technical debt to nontechnical stakeholders?". Other TD communication-related questions (20 questions) centered around convincing management that investing in software quality is not a waste of resources, with an example being "How to convince my boss that quality is a good thing to have in code?". Lastly, the remaining questions (6 questions) focused on the challenge in communicating the various effects of TD to fellow software engineers in the same project and that there is sometimes an ad-hoc mindset in place, which refers to not caring about the software's quality or the refusal to follow any software engineering best practices, such as "How to deal with ad-hoc mindsets?".

TD Representation/documentation Questions in this category (36 question) revolve around how to effectively document TD to not neglect its management, and what are the best ways to represent TD to different stakeholders. The majority of these questions (22 questions) focused on how TD should be documented to be effectively managed in agile development. In other words, should TD be added to the backlog, assigned its own user story, etc. An examples of such questions is "Should we create an Epic for tech-debts". For the remaining questions they either asked what is the most effective way to document TD and what information needs to be recorded in general (6 questions) or in specific tool (8 questions), such as Jira, GitHub, and Visual Studio Team Foundation Server (TFS). Examples of such questions are "What are the crucial key items in recording technical debt?" and "Where do you record Technical Debt in TFS?".

TD Consequences TD consequences questions (21 questions) entail the various ramifications one may face when incurring TD. The majority of TD consequences questions are related to TD consequences at the code level (15 questions), where a given questioner would seek advice on how to achieve a goal and state that said goal cannot be achieved using a normal approach due to the presence of TD. One should note that the questioner does not intend to repay the present TD, but rather circumvent it. An example of such a question is "I have an old library (technical debt) that needs to stay in place for now, and it uses the AWT AffineTransform. Our new graphics code uses the JavaFX Affine. Is there a clear 1:1 mapping between the two?". The remaining questions (6 questions) include asking how to overcome the consequences of TD on the project level, such as decreased productivity and inability to accurately estimate cost of change. "Dealing with inherited development team" is an example of such a question.

TD Measurement The TD measurement category includes 19 questions that focus on the process of accurately measuring TD. The questions can be focused on one or more of the following topics: how can one quantify and measure the amount of TD and its associated repayment costs and/or impact on a business in general (9 questions), how to measure the expected value of repaying TD (5 questions), how to measure the impact of specific TD items and determine their associated repayment value (4 questions), and how to bill for TD repayment fairly (1 question).

TD Incurring Questions that were categorized as TD incurring (15 questions) revolve around finding workarounds to code cases that may result in intentional TD. Questioners asked for "quick", "dirty", and "workaround" methods to achieve a given goal with code. They typically acknowledged that such solutions may necessitate the accumulation of TD. However, they needed to achieve their goals to make an upcoming deadline, adhere to a sudden change in requirements, or achieve a quick win, thus they asked for assistance. "Quick and Dirty Solution to Load Balancing Batch Jobs" is an example of such questions.

TD Prioritization A total of 12 questions were categorized as TD prioritization. Questions within the TD prioritization category can focus on how to determine the tipping point in which a specific TD item cannot be endured anymore (5 questions); how one can prioritize multiple TD items repayments, with either a very pressing deadline or a cost and value tradeoff present (4 questions); how one can prioritize TD repayment when one is pressured to implement new features or fix bugs that have a more tangible ROI (2 questions); and who is responsible for prioritizing TD (1 question).

TD Management The TD management category includes 11 questions that address the different employed strategies and steps taken when managing TD, without a specific focus on one of the previously defined TDM activities, such as "How do you manage technical debt?".

TD Identification TD identification includes 10 questions that are concerned about approaches that aid in revealing TD. Questions either address identifying TD in specific scenarios (8 questions), such as identifying dead and unused code and reasons behind a sudden performance slowdown, or identifying TD in general (2 questions), such as questions on static analysis tools that can identify TD and how TD can be identified in huge projects.

TD Definitions The TD definitions category encompasses 6 questions that address what defines and can be included as TD. Half of the questions asked about specific cases in one's code base and what they should be called, such as parts of code that need to be changed assuming the changes do not affect the behavior of the system. Given that these questions revolve around TD, one should refer to said types of cases as TD, and the questions themselves were edited later to include TD in their tag or body. It should be noted that questioners were not aware of the TD metaphor when originally posting their respective question. An example of such questions is “Errors that don’t make code behave wrong from user’s point of view - how would you call them?”. The other half of questions include asking about whether bugs should be considered as part of TD, what is the definition of TD, or whether shortages in the software engineer process can be considered as TD. An example of such questions is “Are bugs part of technical debt?”.

TD Monitoring TD monitoring includes 4 questions that focus on the most effective approaches to track and monitor TD, such as “How often should a development team review code quality metrics so that we keep reasonable control over technical debt and code quality”.

Other Any TD-related question that did not fit within one of the previously defined categories, did not occur frequently enough to warrant its own category, or was not directly-related to TD was categorized under Other. A total of 16 questions were found to fit under the Other category. Among the 16 total questions, 7 questions asked whether a product should be built fast or well. For the rest of questions, only one instance of each question was present. The following topics were addressed: code ownership, cultural reasons and/or insights behind TD occurrence, if a lack of documentation and refactoring is the norm in big companies, why sacrifice good software engineering practice in favor of making a deadline, best practices to overcome one’s learning TD, TD financial responsibility in frameworks and 3rd party code, examples of obscure software engineering terms (e.g., TD), how to self-improve as a programmer so that one’s old code becomes TD in their eyes, and will a deviation from a given standard introduce a bug or TD.

The acquired set of TD-related questions were able to be categorized into 14 different categories with questions being categorized into one or multiple categories based on their topic. The 14 TD-related categories are as follows: TD tools, TD repayment, TD prevention, TD communication, TD representation/documentation, TD consequences, TD measurement, TD incurring, TD prioritization, TD management, TD identification, TD definitions, TD monitoring, and Other.

4.2 RQ2: What Tags are Used in TD-related Questions and Specific TD-related Categories on Stack Exchange Q&A Websites?

Approach As stated previously, each TD-related question contains a set of tags designated by users. A question must contain at least one tag and can have up to a maximum of 5 tags. First, we calculated the number of uses for a given tag across all of the selected Q&A websites to provide better context on how TD is referred to across all of the selected Q&A websites. Second, we also calculated the total number of utilization of each tag for each of the previously identified TD-related categories, with an aim to unveil the most frequently utilized tags for each of the categories. Acquiring information on what tags are most frequently utilized can provide further context to the domain in which the topic of TD and specific TD-related categories are discussed.

Results Across all three reviewed Stack Exchange Q&A websites, users utilized a total of 636 unique tags in the identified TD-related questions. Of the 636 tags, 68% of the utilized

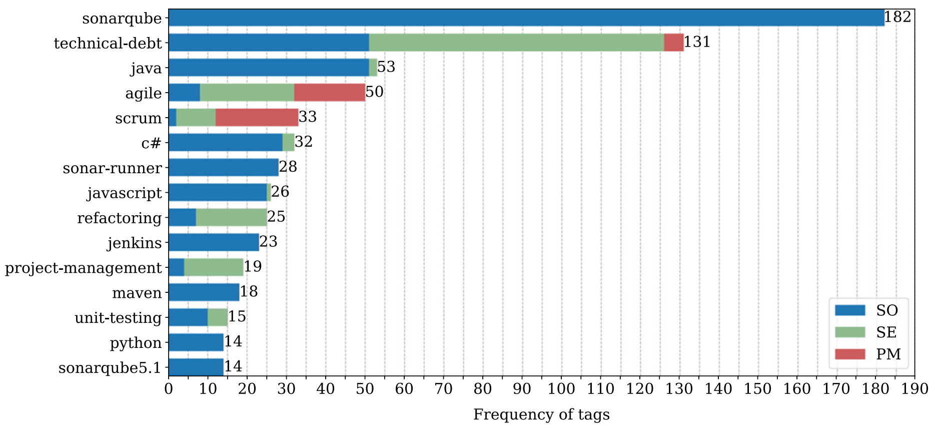


Fig. 4 Distribution of the most utilized tags based on the included Q&A websites

tags were only used once, and only 23% of TD-related questions utilized the technical-debt tag.

The top 15 most frequently utilized tags across all of the three websites are presented in Fig. 4. Blue represents the amount of uses of a given tag from SO, green represents the amount from SE, and red represents the amount from PM. Sonarqube was the most utilized tag (in nearly 32% of questions), followed by technical-debt, java, agile, and scrum. In fact, amongst the top 15 most utilized tags, 3 of them are related to SonarQube in some capacity, even though they were found to only be utilized on SO. Given the popularity of SO in comparison to SE and PM, it was not entirely unexpected that Sonarqube was found to be one of the most utilized tags across all three websites. However, one would assume that the tag technical-debt would have been in top spot, rather than Sonarqube given that only TD-related questions were analyzed.

In addition to analyzing the most frequently utilized tags across the three selected Q&A websites, we also examined the most frequently utilized tags for each of the previously identified TD-related categories. Table 2 displays the top ten most used tags of each category, where one should note that all tags were included if a tie were to occur. In the said table, the frequency of utilization of a given tag is presented as a percentage in parentheses to best convey utilization of a given tag within a given TD-related category, as the number of questions within each category is not the same. Of the 14 categories, the tag technical-debt was the most frequently utilized tag in the majority of them. While technical-debt was in the top ten of most frequently utilized tags of all of the TD-related categories, the tag was seldomly utilized in a few, such as the case with TD consequence where it was only utilized in 10% of questions in the category. Additionally, we found that questions within a few TD-related categories had many unique tags, such as TD communication, while others were tagged using the same few tags, such as questions within TD prioritization only being tagged with five unique tags.

We found that a total of 636 unique tags were utilized to designate one or multiple of the topics addressed within the acquired set of TD-related questions. Moreover, we uncovered that SonarQube was the most utilized tag, followed by technical-debt, java, agile, and scrum. Additionally, the technical-debt tag was in the top 10 of most used tags in all of the TD-related categories.

Table 2 Most utilized tags in each TD-related category and their percentage of utilization

Category	Tags
TD communication	technical-debt(49%), management(20%), communication(18%), project-management(18%), agile(14%), scrum(12%), maintenance(8%), architecture(6%), code-quality(6%), design(6%), estimation(6%), maintainability(6%), planning(6%), product-owner(6%), programming-practices(6%)
TD consequence	agile(14%), database(10%), java(10%), scrum(10%), technical-debt(10%), unit-testing(10%)
TD definition	technical-debt(67%), terminology(67%), agile(50%)
TD identification	python(20%), static-analysis(20%), technical-debt(20%)
TD incurring	java(20%), c#(13%), javascript(13%), spring-boot(13%)
TD management	technical-debt(55%), planning(18%), project-management(18%)
TD measurement	technical-debt(37%), code-quality(16%), refactoring(16%), c#(11%), code-metrics(11%), maintainability(11%), measurement(11%), metrics(11%), performance(11%), project-management(11%)
TD monitoring	scrum(75%), agile(50%), technical-debt(50%), technical-leader(50%)
TD prevention	technical-debt(19%), agile(9%), c#(8%), javascript(8%), python(7%), sql(5%), angular(4%), programming-practices(4%), quality(4%), ruby-on-rails(4%), scrum(4%), sql-server(4%)
TD prioritization	technical-debt(50%), planning(33%), scrum(25%), agile(17%), prioritization(17%)
TD repayment	technical-debt(22%), refactoring(11%), java(9%), javascript(9%), agile(7%), c#(7%), scrum(6%), unit-testing(5%), maintenance(4%), database(4%), design(4%), design-patterns(4%), ios(4%), php(4%), rewrite(4%), ruby-on-rails(4%), testing(4%)
TD representation/ documentation	agile(53%), technical-debt(47%), scrum(31%), estimation(14%), project-management(14%), jira(11%), branching(6%), git(6%), kanban(6%), pivotal-tracker(6%), product-backlog(6%), refactoring(6%), story-points(6%), user-stories(6%), user-story(6%)
TD tools	sonarqube(85%), java(15%), sonar-runner(13%), technical-debt(12%), jenkins(10%), maven(8%), sonarqube5.1(7%), c#(5%), sonarqube-4.5(5%), jacoco(4%)
Other	technical-debt(50%), code-quality(25%), design(25%), development-process(12%), refactoring(12%)

4.3 RQ3: What TD-related Categories are the Most Difficult?

Approach This study's assessment of the difficulty of the previously identified TD-related categories is similar to that previous research (Haque et al. 2020; Abdellatif et al. 2020; Ahmed and Bagherzadeh 2018; Zou et al. 2017; Rosen and Shihab 2016). Two heuristic measures were computed to estimate the difficulty of each category: the percentage of TD-related questions that have no accepted answer and the average median time required for a question of a given category to receive an accepted answer. We chose the first heuristic as the absence of an accepted answer may suggest that the questioner did not find any answer to be suitable and that the said questioner is still facing a challenge that they require assistance with from the Q&A website's users (Rosen and Shihab 2016). We chose the second heuristic due to the fact that the success of a crowd-sourced platform, such as SO, depends heavily on the said site's users ability to provide satisfactory and quick answers. One should also note that the median was chosen as a calculated metric to compensate for potential skewing of the mean due to long-latency answers. This decision was based on the observation that on SO the majority of a question's answers are posted within the first hour of the question's creation, as evidenced by Abdellatif et al. (2020), Rosen and Shihab (2016), and Haque et al. (2020).

We expanded the scope of the research question by calculating the number of views and answers each question had received. Though these two measures are not related to the difficulty of a given category, they may prove fruitful when providing further context on the chosen heuristic measures. Subsequently, we calculated the monotonic association between number of views and answers and the two difficulty heuristics using the Kendall correlation test, with an aim to provide more clarity on how these measures are associated. We chose to utilize the Kendall correlation test due to the test being less sensitive to outliers and ability to produce more robust results (Puth et al. 2015; Abdi 2007).

The authors note that the measures mentioned above have their limitations, as they are merely heuristics that can be confounded by other factors. For example, a lack of an accepted answer could be the result of difficulty arising in locating questions that are within the expertise of a given user, and longer median time to receive an accepted answer might be influenced by a user's availability to check and answer questions or their desire to answer such questions. Therefore, one should only interpret the findings presented in this study with the provided context of the utilized heuristics in mind, while acknowledging the many confounding factors that may have influenced the utilized heuristics.

Results We found that the majority of the compiled set of TD-related questions received an answer. However, more than half of all of the TD-related questions (i.e., 54%) did not receive an accepted answer. The rate of questions that lacked an accepted answer within each TD-related category is presented in column "% W/O AA" in Table 3. The categories TD identification, TD consequences, TD incurring, and TD tools have highest rate of questions without an accepted answer.

In regards to the median waiting time to receive an accepted answer, questions categorized under TD tools, TD incurring, TD consequences, and TD management have the longest average median times required to receive an accepted answer, with values of 909, 406, 232, and 228 minutes, respectively.

The receiving of an accepted answer and the elapsed time to receive an accepted answer may be related to both the number of views the said question received and the overall number of answers received. Therefore, we explored both metrics and computed the correlation. We found that all of the acquired TD-related questions received at least 9 views, with an

Table 3 Average number of views (AVG(V)), percentage of questions with answers (%A), average number of answers (AVG(A)), percentage of questions without an accepted answer (% W/O AA), and median time to receive an accepted answer in minutes (Med(AA)) in each TD-related category

Category	AVG(V)	%A	AVG(A)	% W/O AA	Med(AA)
TD communication	2432	98.04	5.75	49.02	47
TD consequences	458	90.48	1.9	76.19	232
TD definition	1710	100.0	3.83	33.33	206
TD identification	1856	100.0	2.5	80.0	28
TD incurring	324	80.0	1.0	66.67	406
TD management	1399	90.91	4.27	36.36	228
TD measurement	3270	100.0	4.47	26.32	51
TD monitoring	497	100.0	3.0	50.0	104
TD prevention	1407	86.67	2.45	50.67	110
TD prioritization	1105	100.0	3.75	25.0	27
TD repayment	1443	83.58	2.23	52.99	52
TD representation/ documentation	2422	100.0	3.19	33.33	64
TD tools	1785	83.57	1.16	59.62	909
Other	3881	100.0	6.94	43.75	43

average mean of 1,729 views per question and a median of 552 across the three selected Q&A websites. The mean number of views received of a given question in each TD-related category is presented in column "AVG(V)" in Table 3. As displayed in the table, questions that were categorized as Other received the highest average number of views, followed by TD measurement, and TD communication. Additionally, the majority of the gathered TD-related questions received an answer (i.e., 87% of all questions), with a mean of 2 answers per question. When comparing the answer rates of questions among the identified TD-related categories, half of the TD categories were found to have a perfect answer rate, with column "%A" in Table 3 presenting the findings. On the contrary, TD-related questions that fell under TD incurring, TD tools, and TD repayment were found to be the least answered questions, with answering rates of 80%, 84%, and 84%, respectively.

Lastly, the Kendall rank correlation test resulted in statistically significant associations, where all resulting P-values were less than or equal to .01. The obtained correlation coefficients are presented in Table 4. Both the number of answers received and the number of views have a negative, weak correlation with both of the number of questions without an accepted answer and time elapsed to receive an accepted answer (Schober et al. 2018). These results may indicate that a TD-related question that can be considered difficult has a lower chance of receiving an accepted answer at all and a higher median time to receive an accepted answer, as said types of questions have a smaller number of views and less received answers.

Table 4 Correlation coefficients of analyzed characteristics

	# of answers	Views
W/O AA	−0.3	−0.2
Time to receive an accepted answer	−0.3	−0.1

Questions categorized under TD consequences, TD incurring, and TD tools could be considered the most difficult questions for Stack Exchange Q&A website users, as they were all within the top 4 of the examined difficulty heuristics (i.e., percentage of questions that lack an accepted answer and the median time to receive an accepted answer).

5 Discussion

This study observes how the term TD is utilized on three Stack Exchange Q&A websites (i.e., SO, SE, and PM). To do as such, TD-related questions originating from the selected Q&A websites were analyzed and categorized. We found that the TD-related questions could be categorized into one or several of 14 TD-related categories. Furthermore, the results highlight that the TD term is primarily utilized in the context of TD-related tools and code TD, as evidenced by the total number of identified TD tools-related questions in addition to the number of TD repayment and TD prevention questions that are focused on the code level. The results also divulged that SonarQube is the most frequently discussed tool when referencing TD, as evidenced by the number of questions that focus on the tool, which was also observed in previous studies (Gama et al. 2019; Avgeriou et al. 2021).

The results unveiled that Stack Exchange Q&A website users utilized a total of 636 unique tags when referring to TD-related questions. The majority of the tags span over the entire software engineering life cycle and were typically only used once. The Sonarqube and technical-debt tags were the most frequently utilized tags in the set of examined TD-related questions. Given that only TD-related questions were included in this study, one might assume that the technical-debt tag would be the most frequently utilized tag. However, we found Sonarqube to be the most frequently utilized tag, which may have been due to the large number of questions that were categorized under TD-tools. Further examination of the tags of TD-related questions in the context of each TD-related category revealed that the technical-debt tag was the most utilized tag in the majority of the identified categories and was within the top 10 most frequently utilized tags for the remaining categories. Additionally, users utilized a large number of tags within a few categories or made use of the same few tags in other categories. The diversity of the tags can be seen in Table 2. The findings on the diversity of tag utilization may reinforce the assertion that TD relates to a multitude of topics that span the entire software engineering life cycle (Li et al. 2015; Behutiye et al. 2017).

In addition to identifying TD-related categories and reviewing the most frequently utilized tags of said categories, this study attempts to determine which of the previously identified TD-related categories are the most difficult. We determined the level of difficulty of a given category by reviewing the percentage questions that lacked an accepted answer and the average median time required for a question to receive an accepted answer within the said category. Based on the chosen heuristics, it could be concluded that the most difficult TD-related questions on the included Q&A websites lie within the TD consequences, TD incurring, or TD tools categories. The TD consequences, TD incurring, and TD tools categories lack in both of the analyzed heuristics, on average, in comparison to the other TD-related categories. Additionally, further examination of the association among the previously mentioned difficulty heuristics, number of views, and answers a question receive revealed that there exists a statistically significant weak negative correlation, which may suggest that the more difficult topics have a less chance of receiving an accepted answer within a short time frame given that they lack views and answer attempts. However, it is important to note that the definition of what is most difficult is subjective in nature and may vary depending on the perspective at hand. The results of this study are based on heuristics that can be influenced by many other factors. Hence, one should only interpret the results within the context of the utilized heuristics.

6 Implications

Stack exchange Q&A websites can provide valuable, real world insights into a given software engineering topic, most notably ones relating to software engineering. Such insights can prove to be fruitful to those who can utilize them to gain a better understanding on a given topic (Barua et al. 2014; Silva et al. 2021). In the context of this observational study, there are three primary benefits to observing the use of the TD term on Q&A websites. First, software engineering researchers can utilize the results as a guide on where to focus future research efforts. For example, TD tools was one of the most discussed and difficult categories. Researchers can use this information to seek out TD tools-related questions to spot the exact issues the users are facing, where new solutions can then be applied. Second, software practitioners can utilize the results to better identify what are the most common issues regarding TD and its management to, ideally, avoid them or make proactive choices with an aim of mitigating potential consequences to their respective organizations and teams. For example, software practitioners can observe that a few TD-related issues are commonplace within organizations, such as communicating the significance of repaying TD. This study can aid software practitioners in realizing that Q&A websites can be a potential resource to gather more real world experiences and opinions of others who had previously dealt with the same TD-related hurdles. Third, the results can be utilized by software educators to better tailor their respective TD-related curricula to better focus on topics that individuals have questions about in a real world environment.

7 Threats to Validity

The validity of this study is subject to external, construct, and reliability threats. This section discusses each threat and their corresponding mitigation strategies, if applicable (Runeson et al. 2012; Malhotra 2016).

7.1 External Validity

The external validity of a study is concerned with defining a domain to which the acquired results can be applied beyond the circumstances of the study itself, but rather to more general situations (Malhotra 2016). This study and its findings are based on three Stack Exchange Q&A websites: SO, SE, and PM, and the primary goal of the study is to observe how the term TD is utilized on Stack Exchange Q&A websites. One potential threat to the external validity of this study is the identification of TD-related questions on the three included Stack Exchange Q&A websites. The utilized search procedure retrieved questions based on their tag, title, and body. It aimed to obtain all TD-related questions through utilizing a search string that matches different forms of the TD term. However, TD can be referred to using many other terms, such as software maintenance, software evolution, code smells, and more. Therefore, the authors acknowledge that this study and its findings are limited to the search string that was utilized. Additionally, another potential threat to the external validity of this study arose when deciding whether to include or exclude a potential TD-related question. Two authors independently reviewed each question and made an independent decision on if the said question should be included, which may introduce subjectivity. The authors followed up their independent reviews with a joint meeting, compared their results, and calculated the Cohen's Kappa coefficient, a statistical test that is often used to assess agreement between two reviewers (Gisev et al. 2013). The resulting Cohen's Kappa coefficient equaled 1.00, which indicates that the authors were in full agreement (Landis and Koch 1977). Moreover, one should note that the categorization process and the usage of the Cohen's Kappa coefficient are in accordance with the empirical standards for software engineering research (Ralph et al. 2020). Lastly, though the study revealed several findings on TD on the included websites, the generalizability of the study's findings cannot be claimed. One should note that the results presented in this study are only limited to the context of the utilization of the TD term on the three included Q&A websites. There is also a possibility that the study failed to include TD-related categories that are discussed by users on other platforms. To mitigate such a threat, the authors included all Stack Exchange websites that were suggested in various SO TD-related questions as a suitable platform for TD-related questions.

7.2 Construct Validity

The construct validity of a study is concerned with the degree to which the utilized scales, metrics, and instruments actually measure the properties they are supposed to measure (Malhotra 2016; Kitchenham et al. 2015). A potential threat to this study's construct validity is the measurement of difficulty of the identified TD-related categories, given that we solely derived the results from heuristics. Though this study only utilized heuristics that have been commonly employed before in related studies (Haque et al. 2020; Abdellatif et al. 2020; Ahmed and Bagherzadeh 2018; Zou et al. 2017; Rosen and Shihab 2016; Yang et al. 2016), the authors acknowledge that these heuristics are merely estimates and suffer from limitations, as they can be confounded by many other factors, such as the availability of users, the ability to locate questions, and whether users care about the given category. Therefore, one should only interpret the study's findings based on the context of the utilized heuristics bearing in mind their various limitations.

Another potential threat to this study's construct validity is the use of statistical measures and tests. The study utilized common statistical measures, such as percentages and median, to convey its findings, the Cohen's Kappa coefficient to measure inter-rater agreement, and the Kendall correlation test to assess the association between each difficulty heuristic and number of views and answers a question had received. These statistical measures and tests are often utilized to measure what is intended to be measured, and the usages of most of these tests and measures is recommended by the empirical standards for software engineering research (Ralph et al. 2020; Gisev et al. 2013).

7.3 Reliability

A study's reliability is concerned with the reproducibility of the study and its findings. In other words, future researchers should be able to reach the same findings and conclusions if they followed the same steps of the study (Runeson et al. 2012). A potential threat to this study's reliability is the reliance on human judgment in determining the relationship of a question to TD in general and its association with the identified TD-related categories, which may introduce biases to the study. To mitigate any potential introduced biases, we conducted a review process in accordance with the empirical standards for software engineering research and the recommended steps for thematic synthesis in software engineering (Ralph et al. 2020; Cruzes and Dyba 2011). It is also important to note that any biases resulting from human judgment when categorizing TD-related questions into TD-related categories could not be avoided by using an automated technique, such as Latent Dirichlet Allocation (LDA), due to the small number of TD-related questions, as it can harm the quality of obtained results (Fukunaga 2013). Moreover, even the LDA requires human judgment to assign a topic name for the identified topics. The LDA will only aggregate questions into word clusters based on their similarity without providing a name, a human-understandable meaning. For each cluster of topics, one needs to read through the different clusters and manually assign a name, which presents an opportunity for the introduction of human bias or error (Silva et al. 2021; Rosen and Shihab 2016).

8 Related Work

This section includes studies that analyzed Stack Exchange Q&A websites to better understand what is being asked about a given topic. Specifically, the included summaries of studies detailed ones that utilized SO data to further explore TD. A few additional studies that utilized Stack Exchange Q&A websites to investigate a topic other than TD were also listed below, as they aided in serving as the foundation for the design of this study.

Gama et al. (2019) investigated how TD is discussed in SO. The authors identified and analyzed 195 TD-related questions. The results revealed that code TD, infrastructure TD, and architecture TD are the most discussed types of TD. Moreover, TD identification and repayment were the most discussed TD-related activities, as opposed to this study's results of the most discussed TD-related topics being TD tools, TD repayment, and TD prevention. In regards to TD strategies and tools, the study found that TD management is the most discussed strategy, and SonarQube is the most discussed tool. It is important to note that this summary is solely based on the English abstract provided by the authors, as the study was published in a language other than English. Additionally, one should also note that this study was included due to its significance, as it is the first study that investigates how TD is discussed on Q&A websites, and the fact that it reveals several important findings.

Gama et al. (2020) conducted a study on SO's website data to understand how TD is identified by developers. The study utilized the SO Torrent dataset (Baltes et al. 2018) to collect 140 TD-related questions that were manually analyzed. The study found that 71% of the questions included a discussion regarding TD identification. In stark contrast, this study found that TD identification is only discussed in 6 questions on SO. Additionally, the study by Gama et al. (2020) found that code TD was the most frequently discussed topic, followed by infrastructure, architecture, test, defect, and usability TD. The authors identified high-level and low-level indicators for TD items. They also identified the relationship between each indicator and each TD type, based on the frequency of simultaneously occurring in a question, and developed a conceptual framework for identifying TD that was derived from their findings.

Avgeriou et al. (2021) provided an overview of the current landscape of TD measurement tools through the means of a set of objective criteria related to the offered features and their respective popularity. The authors conducted a literature and web search to identify, select, and study nine TD tools, where SO was one of the online media sources included by the authors. Similar to this study's findings, Avgeriou et al. (2021) found that SonarQube was the most referenced TD tool on SO. Moreover, the authors came to a similar conclusion that even though SO has a tag dedicated to TD, questions related to TD primarily inquire about various capabilities of the SonarQube tool.

Other studies that have analyzed Stack Exchange Q&A website questions to understand multiple software engineering topics, without a specific focus on TD, are listed here: Haque et al. (2020), Abdellatif et al. (2020), Han et al. (2020), Ahmed and Bagherzadeh (2018), Zou et al. (2017), Rosen and Shihab (2016), Yang et al. (2016), Barua et al. (2014), and Bajaj et al. (2014).

9 Conclusion

TD has been gaining an increasingly amount of interest from those within the software engineering research field. The topic has many avenues to explore and delve further into. Fortunately, Stack Exchange Q&A websites can provide valuable, real world perspectives on a number of software engineering topics. This study presents an observational study that aims to observe how the TD term is utilized on Stack Exchange Q&A websites: SO, SE, and PM. We identified a total of 578 questions as being TD-related and, subsequently, analyzed for their content. We then categorized the acquired set of TD-related questions into 14 different categories, with questions being categorized into one or multiple categories based on their topic. Additionally, the results revealed that 636 unique tags have been utilized to designate one or multiple of the topics addressed within the acquired set of TD-related questions. The five most utilized tags were the following: Sonarqube, technical-debt, java, agile, and scrum. Lastly, the study also attempted to identify if any of the previously identified TD-related categories were more difficult than others and if so, which of the said categories are the most difficult. We estimated the difficulty of a given category by reviewing two heuristic measures: the percentage of questions that lacked an accepted answer and the average median time to receive an accepted answer. We found that the TD consequences, TD incurring, and TD tools categories could be considered the most difficult questions for Stack Exchange Q&A website users, as they were all within the top 4 of the examined difficulty heuristics.

This observational study serves as evidence that TD is addressed on Stack Exchange Q&A websites. Moreover, it allows individuals to gain a broader perspective on TD-related questions on Q&A websites. Specifically, this study observed that TD-related questions can relate to a multitude of varying topics, the tags utilized on the TD-related questions span the entire software engineering life cycle, and that a few TD-related categories both lack accepted answers and have a longer median time to receive an accepted answer than other categories. Individuals that are in the process of attempting to address TD may find it beneficial to review Q&A websites for more information on the topic.

Acknowledgements This research project was supported by a grant from the "Research Center of College of Computer and Information Sciences, Deanship of Scientific Research, King Saud University".

Data Availability The datasets generated during and/or analysed during the current study are available in the Open Science Foundation (OSF), https://t.ly/_Fqi

Declarations

Conflict of Interests The authors declare no potential conflicts of interest.

References

Abdellatif A, Costa D, Badran K, Abdalkareem R, Shihab E (2020) Challenges in chatbot development: A study of stack overflow posts. In: Proceedings of the 17th international conference on mining software repositories, MSR '20. Association for Computing Machinery, New York, pp 174–185. https://doi.org/10.1145/3379597.3387472

Abdi H (2007) The kendall rank correlation coefficient. Encycl Meas Stat 2:508–510

Ahmed I, Brindescu C, Mannan UA, Jensen C, Sarma A (2017) An empirical examination of the relationship between code smells and merge conflicts. In: 2017 ACM/IEEE international symposium on empirical software engineering and measurement (ESEM), pp 58–67. https://doi.org/10.1109/ESEM.2017.12

Ahmed S, Bagherzadeh M (2018) What do concurrency developers ask about? A large-scale study using stack overflow. In: Proceedings of the 12th ACM/IEEE international symposium on empirical software engineering and measurement, ESEM '18. Association for Computing Machinery, New York. https://doi.org/10.1145/3239235.3239524

Allman E (2012) Managing technical debt. Commun ACM 55(5):50–55. https://doi.org/10.1145/2160718.2160733

Alves NS, Mendes TS, de Mendonça MG, Spínola RO, Shull F, Seaman C (2016) Identification and management of technical debt: A systematic mapping study. Inf Softw Technol 70:100–121. https://doi.org/10.1016/j.infsof.2015.10.008

Ampatzoglou A, Ampatzoglou A, Chatzigeorgiou A, Avgeriou P (2015) The financial aspect of managing technical debt. Inf Softw Technol 64(C):52–73. https://doi.org/10.1016/j.infsof.2015.04.001

Avgeriou P, Kruchten P, Ozkaya I, Seaman C (2016) Managing technical debt in software engineering (Dagstuhl Seminar 16162). Dagstuhl Rep 6(4):110–138. https://doi.org/10.4230/DagRep.6.4.110

Avgeriou PC, Taibi D, Ampatzoglou A, Arcelli Fontana F, Besker T, Chatzigeorgiou A, Lenarduzzi V, Martini A, Moschou A, Pigazzini I, Saarimaki N, Sas DD, de Toledo SS, Tsintzira AA (2021) An overview and comparison of technical debt measurement tools. IEEE Softw 38(3):61–71. https://doi.org/10.1109/MS.2020.3024958

Bajaj K, Pattabiraman K, Mesbah A (2014) Mining questions asked by web developers. In: Proceedings of the 11th working conference on mining software repositories, MSR 2014. Association for Computing Machinery, New York, pp 112–121, https://doi.org/10.1145/2597073.2597083

Baltes S, Dumani L, Treude C, Diehl S (2018) Sotorrent: Reconstructing and analyzing the evolution of stack overflow posts. In: Proceedings of the 15th international conference on mining software repositories, MSR '18. Association for Computing Machinery, New York, pp 319–330. https://doi.org/10.1145/3196398.3196430

Barua A, Thomas SW, Hassan AE (2014) What are developers talking about? An analysis of topics and trends in stack overflow. Empir Softw Engg 19(3):619–654. https://doi.org/10.1007/s10664-012-9231-y

Behutiye WN, Rodríguez P, Oivo M, Tosun A (2017) Analyzing the concept of technical debt in the context of agile software development: A systematic literature review. Inf Softw Technol 82:139–158. https://doi.org/10.1016/j.infsof.2016.10.004

Biffl S, Aurum A, Boehm B, Erdogmus H, Grünbacher P (2006) Value-based software engineering. Springer Science & Business Media, Berlin

Codabux Z, Williams B (2013) Managing technical debt: An industrial case study. In: 2013 4th International workshop on managing technical debt (MTD), pp 8–15. https://doi.org/10.1109/MTD.2013.6608672

Cruzes DS, Dyba T (2011) Recommended steps for thematic synthesis in software engineering. In: 2011 international symposium on empirical software engineering and measurement, pp 275–284. https://doi.org/10.1109/ESEM.2011.36

Cunningham W (1992) The WyCash portfolio management system. SIGPLAN OOPS Mess 4(2):29–30. https://doi.org/10.1145/157710.157715

Curtis B, Sappidi J, Szynkarski A (2012) Estimating the principal of an application's technical debt. IEEE Softw 29(6):34–42. https://doi.org/10.1109/MS.2012.156

Erdogmus H (1999) Comparative evaluation of software development strategies based on net present value. In: International workshop on economics-driven software engineering research EDSER, p 1

Fernández-Sánchez C, Garbajosa J, Yagüe A, Perez J (2017) Identification and analysis of the elements required to manage technical debt by means of a systematic mapping study. J Syst Softw 124(C):22–38. https://doi.org/10.1016/j.jss.2016.10.018

Fowler M (2018) Refactoring: improving the design of existing code. Addison-Wesley Professional, Reading

Fukunaga K (2013) Introduction to statistical pattern recognition. Elsevier, New York

Gama E, Paixao M, Freire ESS, Cortés MI (2019) Technical debt's state of practice on stack overflow: A preliminary study. In: Proceedings of the XVIII Brazilian symposium on software quality, SBQS'19. Association for Computing Machinery, New York, pp 228–233. https://doi.org/10.1145/3364641.3364668

Gama E, Freire S, Mendonça M, Spínola RO, Paixao M, Cortés MI (2020) Using stack overflow to assess technical debt identification on software projects. In: Proceedings of the XXXIV Brazilian symposium on software engineering, SBES ’20. Association for Computing Machinery, New York, pp 730–739. https://doi.org/10.1145/3422392.3422429

Gisev N, Bell JS, Chen TF (2013) Interrater agreement and interrater reliability: Key concepts, approaches, and applications. Res Soc Adm Pharm 9(3):330–338. https://doi.org/10.1016/j.sapharm.2012.04.004

Guo Y, Seaman C (2011) A portfolio approach to technical debt management. In: Proceedings of the 2nd workshop on managing technical debt, MTD ’11. Association for Computing Machinery, New York, pp 31–34. https://doi.org/10.1145/1985362.1985370

Guo Y, Seaman C, Gomes R, Cavalcanti A, Tonin G, Da SilvaFQB, Santos ALM, Siebra C (2011) Tracking technical debt — an exploratory case study. In: 2011 27th IEEE international conference on software maintenance (ICSM), pp 528–531. https://doi.org/10.1109/ICSM.2011.6080824

Guo Y, Spínola RO, Seaman C (2016) Exploring the costs of technical debt management — a case study. Empirical Softw Engg 21(1):159–182. https://doi.org/10.1007/s10664-014-9351-7

Han J, Shihab E, Wan Z, Deng S, Xia X (2020) What do programmers discuss about deep learning frameworks. Empir Softw Eng 25(4):2694–2747. https://doi.org/10.1007/s10664-020-09819-6

Haque MU, Iwaya LH, Babar MA (2020) Challenges in docker development: A large-scale study using stack overflow. In: Proceedings of the 14th ACM / IEEE international symposium on empirical software engineering and measurement (ESEM), ESEM ’20. Association for Computing Machinery, New York. https://doi.org/10.1145/3382494.3410693

Izurieta C, Vetrò A, Zazworka N, Cai Y, Seaman C, Shull F (2012) Organizing the technical debt landscape. In: 2012 3rd international workshop on managing technical debt (MTD), pp 23–26. https://doi.org/10.1109/MTD.2012.6225995

Kitchenham BA, Budgen D, Brereton P (2015) Evidence-based software engineering and systematic reviews, vol 4. CRC Press, Boca Raton

Kruchten P, Nord RL, Ozkaya I (2012) Technical debt: From metaphor to theory and practice. IEEE Softw 29(6):18–21. https://doi.org/10.1109/MS.2012.167

Landis JR, Koch GG (1977) The measurement of observer agreement for categorical data. Biometrics, 159–174

Lenarduzzi V, Orava T, Saarimäki N, Systa K, Taibi D (2019) An empirical study on technical debt in a finnish SME. In: 2019 ACM/IEEE international symposium on empirical software engineering and measurement (ESEM), pp 1–6. https://doi.org/10.1109/ESEM.2019.8870169

Li Z, Avgeriou P, Liang P (2015) A systematic mapping study on technical debt and its management. J Syst Softw 101(C):193–220. https://doi.org/10.1016/j.jss.2014.12.027

M Bomfim M, A Santos V (2017) Strategies for reducing technical debt in agile teams. In: Silva da Silva T, Estácio B, Kroll J, Mantovani Fontana R (eds) Agile methods. Springer International Publishing, Cham, pp 60–71

Malhotra R (2016) Empirical research in software engineering: concepts, analysis, and applications. CRC Press, Boca Raton

Martini A, Bosch J, Chaudron M (2014) Architecture technical debt: Understanding causes and a qualitative model. In: 2014 40th EUROMICRO conference on software engineering and advanced applications, pp 85–92. https://doi.org/10.1109/SEAA.2014.65

de Mello RM, Oliveira R, Garcia A (2017) On the influence of human factors for identifying code smells: A multi-trial empirical study. In: 2017 ACM/IEEE international symposium on empirical software engineering and measurement (ESEM), pp 68–77. https://doi.org/10.1109/ESEM.2017.13

Morgenthaler JD, Gridnev M, Sauciuc R, Bhansali S (2012) Searching for build debt: Experiences managing technical debt at Google. In: Proceedings of the 3rd international workshop on managing technical debt, MTD ’12. IEEE Press, pp 1–6

Nord RL, Ozkaya I, Kruchten P, Gonzalez-Rojas M (2012) In search of a metric for managing architectural technical debt. In: 2012 Joint working IEEE/IFIP conference on software architecture and european conference on software architecture, pp 91–100. https://doi.org/10.1109/WICSA-ECSA.212.17

Parnas DL (1994) Software aging. In: Proceedings of the 16th international conference on software engineering, ICSE ’94. IEEE Computer Society Press, Washington, pp 279–287

Puth MT, Neuhäuser M, Ruxton GD (2015) Effective use of Spearman’s and Kendall’s correlation coefficients for association between two measured traits. Anim Behav 102:77–84. https://doi.org/10.1016/j.anbehav.2015.01.010

Ralph P, Ali Nb, Baltes S, Bianculli D, Diaz J, Dittrich Y, Ernst N, Felderer M, Feldt R, Filieri A et al (2020) Empirical standards for software engineering research. arXiv:201003525

Rios N, de Mendonça Neto MG, Spínola RO (2018) A tertiary study on technical debt: Types, management strategies, research trends, and base information for practitioners. Inf Softw Technol 102:117–145. https://doi.org/10.1016/j.infsof.2018.05.010

Rios N, Spínola RO, Mendonça M, Seaman C (2020) The practitioners' point of view on the concept of technical debt and its causes and consequences: A design for a global family of industrial surveys and its first results from Brazil. Empirical Softw Engg 25(5):3216–3287. https://doi.org/10.1007/s10664-020-09832-9

Rosen C, Shihab E (2016) What are mobile developers asking about? a large scale study using stack overflow. Empir Softw Engg 21(3):1192–1223. https://doi.org/10.1007/s10664-015-9379-3

Rubin K (2012) Essential scrum: A practical guide to the most popular agile process (Addison-Wesley Signature Series (Cohn)). Addison-Wesley Professional, Reading

Runeson P, Host M, Rainer A, Regnell B (2012) Case study research in software engineering: Guidelines and examples. Wiley, New York

Schober P, Boer C, Schwarte LA (2018) Correlation coefficients: appropriate use and interpretation. Anesth Analg 126(5):1763–1768. https://doi.org/10.1213/ANE.0000000000002864

Silva CC, Galster M, Gilson F (2021) Topic modeling in software engineering research. Empir Softw Engg 26(6). https://doi.org/10.1007/s10664-021-10026-0

Spínola RO, Vetrò A, Zazworka N, Seaman C, Shull F (2013) Investigating technical debt folklore: Shedding some light on technical debt opinion. In: 2013 4th international workshop on managing technical debt (MTD), pp 1–7. https://doi.org/10.1109/MTD.2013.6608671

Sterling C (2010) Managing software debt: Building for inevitable change. Addison-Wesley Professional, Reading

Suryanarayana G, Samarthyam G, Sharma T (2014) Refactoring for software design smells: managing technical debt. Morgan Kaufmann, San Mateo

Tom E, Aurum A, Vidgen R (2013) An exploration of technical debt. J Syst Softw 86(6):1498–1516. https://doi.org/10.1016/j.jss.2012.12.052

Yang XL, Lo D, Xia X, Wan ZY, Sun JL (2016) What security questions do developers ask? a large-scale study of stack overflow posts. J Comput Sci Technol 31(5):910–924. https://doi.org/10.1007/s11390-016-1672-0

Yin RK (2009) Case study research: Design and methods, vol 5. Sage, Newbury Park

Zazworka N, Seaman C, Shull F (2011a) Prioritizing design debt investment opportunities. In: Proceedings of the 2nd workshop on managing technical debt, MTD '11. Association for Computing Machinery, New York, pp 39–42. https://doi.org/10.1145/1985362.1985372

Zazworka N, Shaw MA, Shull F, Seaman C (2011b) Investigating the impact of design debt on software quality. In: Proceedings of the 2nd workshop on managing technical debt, MTD '11. Association for Computing Machinery, New York, pp 17–23. https://doi.org/10.1145/1985362.1985366

Zou J, Xu L, Yang M, Zhang X, Yang D (2017) Towards comprehending the non-functional requirements through developers eyes. Inf Softw Technol 84(C):19–32. https://doi.org/10.1016/j.infsof.2016.12.003