

Who is to Blame: A Comprehensive Review of Challenges and Opportunities in Designer-Developer Collaboration

SHUTONG ZHANG, University of Toronto, Canada

TIANYU ZHANG, University of Toronto, Canada

JINGHUI CHENG, Polytechnique Montreal, Canada

SHURUI ZHOU, University of Toronto, Canada

Software development relies on effective collaboration between Software Development Engineers (SDEs) and User eXperience Designers (UXDs) to create software products of high quality and usability. While this collaboration issue has been explored over the past decades, anecdotal evidence continues to indicate the existence of challenges in their collaborative efforts. To understand this gap, we first conducted a systematic literature review (SLR) of 45 papers published since 2004, uncovering three key collaboration challenges and two main categories of potential best practices. We then analyzed designer and developer forums and discussions from one open-source software repository to assess how the challenges and practices manifest in the status quo. Our findings have broad applicability for collaboration in software development, extending beyond the partnership between SDEs and UXDs. The suggested best practices and interventions also act as a reference for future research, assisting in the development of dedicated collaboration tools for SDEs and UXDs.

CCS Concepts: • **Human-centered computing** → **Collaborative and social computing**; • **Software and its engineering** → **Collaboration in software development**.

Additional Key Words and Phrases: Collaborative Software Development, User Experience Design, Designer-Developer Collaboration, Meta-Analysis, Literature Review

ACM Reference Format:

Shutong Zhang, Tianyu Zhang, Jinghui Cheng, and Shurui Zhou. 2025. Who is to Blame: A Comprehensive Review of Challenges and Opportunities in Designer-Developer Collaboration. *Proc. ACM Hum.-Comput. Interact.* 9, 2, Article CSCW207 (April 2025), 33 pages. <https://doi.org/10.1145/3711105>

1 Introduction

In the collaborative software development process, multiple groups of experts with diverse backgrounds need to collaborate to achieve a common goal. With the increasing demand for human-centred design and software usability, the collaboration between designers and developers has become a key factor in the achievement of success of software development. Numerous widely recognized software development models, such as waterfall [12, 15] and agile [13], encompass multiple phases that engage both software development engineers (SDEs) and user experience designers (UXDs), such as requirement elicitation, design exploration and specification, and usability verification and validation [72, 73]. Although there are a lot of discussions in the industry

Authors' Contact Information: Shutong Zhang, shutong.zhang@alumni.utoronto.ca, University of Toronto, Toronto, ON, Canada; Tianyu Zhang, tianyu.theodosia.zhang@alumni.utoronto.ca, University of Toronto, Toronto, ON, Canada; Jinghui Cheng, jinghui.cheng@polymtl.ca, Polytechnique Montreal, Montreal, QC, Canada; Shurui Zhou, shuruiz@ece.utoronto.ca, University of Toronto, Toronto, ON, Canada.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM 2573-0142/2025/4-ARTCSCW207

<https://doi.org/10.1145/3711105>

about how to achieve an effective UXD-SDE collaboration, these discussions are scattered and not synthesized. It remains uncertain how the two groups should coordinate their efforts at specific stages of the development process and how they can collaborate efficiently and communicate effectively.

UX design, first introduced in the human-computer interaction (HCI) community, aims to enhance user satisfaction when interacting with systems or products [47]. After more than a decade of development and evolution, UX design has cultivated its own methodologies and procedures, many of which also follow a central lifecycle comprising stages such as analysis, design, prototyping, and evaluation [45, 47]. When comparing the UX lifecycles with those specified in classic SE literature, several points coincide or exhibit similarities. Nevertheless, the disparity between these two disciplines, potentially introduces complexities and challenges that often hinder effective collaboration.

In recent years, there has been an increasing number of literature exploring how to better integrate software engineering (SE) and UX design [57, 60, 61]. Several systematic literature reviews (SLRs) have focused on the idea of integrating UX design into agile model [19, 70, 85, 103], summarizing the distinction between agile and UX design [103], with a focus on the integration [101], new frameworks that combine common practices and processes [70], and UX management approaches [52]. Other research has examined the integration of SE and UX through various angles, such as processes, artifacts, and training. For example, a literature review explored the training of software development engineers in usability engineering [18], while two mapping studies concentrated on usability evaluation [83] and the artifacts used for communication [40]. However, most of these studies have only concentrated on single aspects of the problem, resulting in a fragmented understanding (see detailed summary in Section. 2). Additionally, it is crucial to refresh the collection of published papers up to date in order to comprehend current practices and concerns and augment academic research findings with the insights and perspectives of practitioners.

Therefore, in this research, we undertake a study that emphasizes a holistic understanding of the interaction between the two groups of experts. We aimed to achieve this through a broad systematic literature review, supplemented by the analysis of discussions in forums and open-source issue trackers. We aim to gain a qualitative understanding of how these two fields overlap in theory, the collaboration challenges, the current status of collaboration, and how we could promote the two groups to work together better. With this goal in mind, we pose the following research questions:

- **RQ1:** *What are the challenges of UXDs and SDEs collaboration?* To answer this question, we aim to investigate the obstacles and barriers that hinder collaboration between UX designers and SE developers.
- **RQ2:** *What are the potential best practices to overcome the challenge/to solve the problem?* For this question, We aim to identify strategies that could be used to overcome the challenges summarized in RQ1.
- **RQ3:** *What are the current practices and tooling support of the collaboration between UXDs and SDEs?* Lastly, we examine how the results obtained from RQ1&2 have manifested in real-world scenarios. Specifically, we evaluate the existing tooling support to determine whether the challenges identified in RQ1 have been tackled, if the best practices outlined in RQ2 have been effectively put into practice, and if there are any remaining areas where enhancements are required.

In this study, we reviewed 45 papers spanning from 2004 to 2023 to address RQ1 and RQ2. In order to explore the present methods and practices (as per RQ3), we conducted searches on posts and issues within four online forums that hosted a substantial community of SDEs and UXDs, as well as investigated the issue discussions in one open-source project that involved both UXDs

and SDEs. Our examination reveals three key collaboration challenges: (1) Separate decision-making procedures; (2) Different professional behaviors; (3) Lack of mutual understanding, and two main categories of potential best practices: (1) Improve development workflow; (2) Enhance communication. Additionally, by examining the search results from forums and the issue discussion in one popular open-source project, we uncovered a disparity between the existing tooling support for development work involving both UXDs and SDEs and the recommended best practices. Together, our findings provided insights related to collaboration suggestions and future research directions.

In summary, we make the following contributions: (1) We present a summary of three key challenges, further categorized into eleven detailed classifications, and connect them with two main best practices, which are divided into six sub-practices to improve the efficiency of collaboration between UXDs and SDEs; (2) we highlight the gaps between current practices and tooling support and the proposed best practices; (3) we identify recommendations for improvement and multiple future research directions.

2 Related Work

In this section, we summarize the prior studies focusing on collaboration challenges during the software development procedure, and then justify the importance of conducting an SLR on UXD-SDE collaboration within the HCI, CSCW, and SE communities.

2.1 Collaboration Scenarios During Software Development Process

The majority of software projects go beyond the capacity of a single SDE, necessitating collaboration among multiple SDEs either within the same team or across different teams. Researchers have explored collaboration practices and challenges among SDEs in diverse contexts, including co-located environments [7, 14, 105], distributed environments [43, 50, 80], as well as the transition [16, 64] and comparison [25, 56] between the two scenarios. Moreover, collaboration scenarios can also be distinguished between asynchronous [48] and asynchronous [97], each presenting distinct tradeoffs that require careful consideration [41]. Researchers have examined various forms of awareness in collaborative work and have developed tools [27] to consolidate team members' activities, enhancing awareness throughout the collaboration process [23, 42, 98, 108]. Recently, with the incorporation of machine learning (ML) techniques has resulted in an increasing number of software systems integrating ML components. This development has necessitated SDEs to engage in collaboration with ML experts, giving rise to interdisciplinary collaborative scenarios and the associated challenges covering different aspects (i.e., Communication, Documentation, Engineering, and Process) [71, 78].

Likewise, the HCI community has explored the collaboration among UXDs in the development of interactive systems [30] to improve communication and coordination efficiencies by analyzing collaborative UX tools [31]. Along the line of designing ML-based systems, researchers looked into how UXDs work with data scientists to build high-quality human-centred AI products [114].

2.2 Related SLRs on SE and UX Integration

The significance of software system usability in recent times has led to a growing body of literature that delves into the integration of SE and UX methodologies and processes [57, 60, 61]. Accordingly, SLR research [19, 95, 101] and systematic mapping studies [70] have been conducted to summarize the overview of how user-centered design is introduced into the Agile development process.

In particular, in 2010, Sohaib et al. conducted a literature review on difficulties and suggested approaches for usability engineering and agile software development integration [103]. However, the study was not conducted systematically and it was unclear about paper selection, and screening procedure, leaving the quality of the study results questionable. Later in 2011, Silva et al. reviewed

the integration of user-centred design and agile methods, with a focus on the merged process model and the artifacts that were used to support the collaboration and did not include human factors [101]. Similarly, in 2016, Caballero et al. conducted a systematic mapping study, covering papers from 2001 to 2013, to understand how agile teams integrate user-centred design tools and work with UXDs to better understand user requirements [70]. The research outlined various team structures that were reported from the paper corpus, such as cross-functional teams comprising both UXDs and SDEs, as well as independent teams of SDEs and UXDs that collaborate on a project basis. Additionally, the paper listed several strategies that prior work reported to ensure collaboration efficiency, such as continuous communication and close collaboration between co-located UX and SE experts. However, the paper only listed the summaries from prior work and did not synthesize the findings into categories and themes.

The closest SLR study to our work was conducted by Salah et al. which summarised 71 research papers published between 2000 and 2012, primarily examining the interaction between agile developers and user-centred designers [95]. They identified several collaboration challenges, including difficulties in performing UCD activities, UX testing, and optimizing work dynamics. However, the primary emphasis of their study was on the challenges of process integration, leaving notable gaps in understanding the distinct professional behaviours and other social aspects during the collaboration process.

Moreover, several SLRs focused on specific activities between SDEs and UXDs, such as the evaluation methods in Agile-UX [83], the training condition of SDEs in UX domain [18], and the artifacts in agile user-centered design [40]. The most recent SLR was conducted by Hinderks et al. in 2022, primarily focusing on the approaches that manage the UX process in the agile development model [52]. This review analyzed 49 relevant studies and identified several approaches for UX management. The study treated “collaboration and communication” as a simple and successful approach to manage UX-related tasks. Unlike previous reviews, our study focuses primarily on the collaboration aspects throughout the whole development process, covering various development models and also open-source context. Additionally, we conducted qualitative analysis on four popular forums that SDEs and UXDs tend to use and one popular open-source project in order to complement the findings from academic papers.

3 Review Protocol

Our study relied on two different sources: (1) a systematic literature review (SLR), and (2) a qualitative analysis of the practitioners’ discussion in forums and issue trackers. Specifically, the SLR study covering five online databases aims to answer RQ1 (collaboration challenges) and RQ2 (best practices) following the guidelines established by Kitchenham et al. [62, 63] and the PRISMA framework [76]. Figure 2 shows an overview of the paper selection process. Then to complement the SLR results, we analyzed posts and issues from four designer-developer forums and investigated the issue discussions from open-source projects hosted on GitHub, respectively. In the following, we detail the methods used to conduct the SLR (Section 3.1) and the qualitative analysis (Section 3.2).

3.1 Systematic Literature Review

3.1.1 Identification: Keyword Searching. Similar to prior SLRs on UX and SE focusing on process integration [58, 101, 103], we divide our search string into SE and UX components. We first manually selected 14 papers as references and keywords from prior SLRs as initial keyword selections [58, 101, 103], then iteratively refined the keyword list by using them to search papers in selected databases until all papers were covered. In specific, the keyword “agile” was selected

because many related papers are from the Agile Conference.¹ As a result, we defined two keyword lists used for querying papers (as shown in Table 1). Combining all keywords, the final string we used for keyword search is presented as follows: UX AND SE.

Table 1. Keywords used to formulate searching string in the SLR. Combining all keywords, the final string we used for keyword search is presented as follows: UX AND SE.

Type	Search String
UX	“designer” OR “usability” OR “ux” OR “user experience” OR “user-centered(centred)” OR “user interface”
SE	“developer” OR “software development” OR “software developing” OR “software process” OR “software engineer” OR “agile”

We applied the above searching string in abstract, title, and keywords (ATK) following the guidelines [63] in the IEEE Xplore digital database, the ACM Digital Library, the SpringerLink Digital Library, the Wiley Online Library, and the Engineering Village. Two co-authors iteratively examined papers randomly chosen from the search results to determine the relevant data source.

3.1.2 Screening, Eligibility, and Inclusion. Following identification, we conducted screening and eligibility assessments, taking into account the specified inclusion criteria (IC) and exclusion criteria (EC) that were iteratively formulated. In particular, an included paper must:

- **IC1:** be published in peer-reviewed journals or event proceedings;
- **IC2:** be written in English;
- **IC3:** be published between 2004 and 2023. Since the fields of SE and HCI are rapidly developing, we restrict the start year to 2004 to focus on papers published in the past 20 years, so that our study covers the most relevant information;
- **IC4:** contains pertinent information related to either collaboration challenges between designers and developers, or best practices to facilitate collaboration between them.

At the same time, we exclude a paper if:

- **EC1:** The paper is a work-in-progress, poster, demo, or an extended abstract;
- **EC2:** The paper is a patent or standard;
- **EC3:** The paper is less than four pages;
- **EC4:** The paper is published in a local conference (Appendix A.3 lists the collection of keywords);
- **EC5:** The main topic is not UXD and SDE collaboration but other fields (Appendix A.3 lists the collection of these keywords);
- **EC6:** The paper constitutes secondary studies (i.e., an SLR or informal literature survey), to ensure concentration on primary studies [63].

3.1.3 Paper Selection Process. The initial keyword search resulted in 5,736 papers from five databases published between January 2004 and August 2023. After removing duplicate papers, we have 5,112 papers in total. During our random checking, we observed that our initial result contained a large number of irrelevant papers. Thus, we implemented a keyword filter to exclude papers irrelevant to software engineering or UX design. Specifically, we established a range of keywords including terms such as “linear algebra,” “computer vision,” “gravity,” and others. A full

¹<https://agile-gi.eu/>

list of keywords is available in Appendix A.3. We filtered out papers that contain at least one keyword from the list in their title, keywords, or abstract, resulting in 3,860 remaining papers.

We then conducted the following three iterative processes of paper selection and information extraction as described below. To ensure accuracy and consistency, we iteratively evaluate and refine the review protocol and criteria before and after each process. For each paper, we define three types of labels, including *Y* for include, *N* for exclude, and *NA* for not sure at each step. Figure. 1 shows the data collection process.

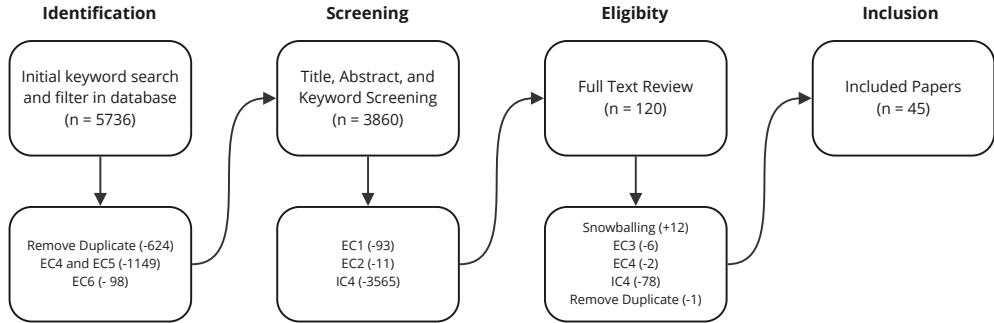


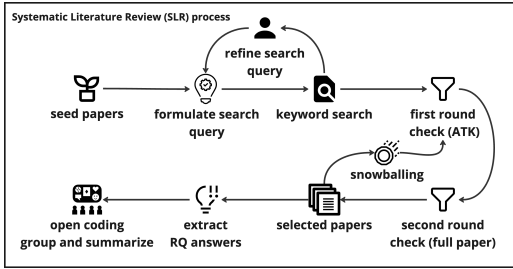
Fig. 1. Data collection process and statistics following Identification, Screening, Eligibility, and Inclusion.

- **(ATK-only)** We reviewed the abstracts, titles, and keywords (ATK) of all the papers, accessing them against the inclusion and exclusion criteria. In terms of eligibility, each author first reviewed 100 papers independently and then held meetings to address conflicts. After completing one round of labeling, we assigned papers with labels *Y*, *NA*, and 20% papers with label *N* to a second reviewer for cross-checking. For papers with different opinions, we held meetings for discussion until the team reached an agreement. This process led to the identification of 120 papers.
- **(Full-text review)** We thoroughly examined the full text of each paper against the criteria. Similar to the previous round, we carried out a pilot check to refine the review criteria and then applied the criteria for paper screening. This step led to 41 papers at the end.
- **(Snowballing)** While reading papers, we also applied backward snowballing to search for missing related papers in the initial keyword search [112]. Similar to the process described before, we filter out irrelevant papers and uncover responses to RQs. The initial snowballing process identified 12 papers; among them, 4 papers were added to the final review, resulting in 45 papers in our corpus.

3.1.4 Data Analysis. Extracting findings. Each author read a set of papers and then highlighted the quotes from the paper that are related to collaboration challenges (RQ1) and best practices (RQ2). Each paper was then assigned to a second author to cross-check and then discussed during meetings to resolve conflicts and uncertainty. We extracted a total of 126 excerpts related to challenges and 66 excerpts related to best practices that could mitigate challenges.

Categorizing findings that share topical similarities using affinity diagramming. Various papers organized their findings employing diverse terminologies and grouping approaches. In our project, we aim to present a comprehensive overview of both the challenges and corresponding solutions through a cohesive and consistent methodology. Therefore, we conducted affinity diagramming [24] to group similar findings together. Adhering to the guidelines of the approach, we

performed the activity on Miro.com by creating one sticky note for every highlighted sentence from our corpus. Two authors created the initial grouping and other authors validated the initial grouping and raised different opinions. Then all authors discussed disagreements during meetings, refined the groupings, and obtained the final themes reported in the results. Regarding the sticky notes associated with challenges, we revisited the relevant sections to grasp the context, noting the possible causes and impacts of the challenges if they are discussed in the papers. For the sticky notes related to practices that mitigate challenges, we grouped them into themes and linked them to corresponding challenges that were identified in the first step.



(a) Paper selection process

Database	Initial Papers	First Round	Final Papers
IEEE	1855	37	13
ACM	1430	25	11
SpringerLink	173	13	5
Weily	106	8	3
Engineering Village	2172	37	9
Snowballing	-	12	4
Total	5736	132	45

(b) Paper selection statistics

Fig. 2. Paper Selection Process and Statistics. We first applied the keyword search and then conducted two rounds of filtering. Sub-figure (a) shows the overview of the selection process and sub-figure (b) shows the number of papers in each stage.

3.2 Analyzing SE&UX Discussions in Forums and Issue Trackers

To triangulate and complement the findings from SLR and further investigate the current state of SE&UX collaboration in real-world settings, we qualitatively analyzed the relevant discussions targeting the SE-UX collaboration issues in four designer-developer forums. Furthermore, we perform a brief analysis of a widely used open-source project (i.e., VS Code) in anticipation of discovering firsthand instances of collaboration communication between UXDs and SDEs during the open-source development process.

3.2.1 Data Selection From Designer-Developer Forums.

Search Strategy. We focused on four forums, including UX Stack Exchange [29], Reddit UX [94], Reddit SE [93], and Stack Overflow [28], that are renowned for hosting communities of professionals that provide experiences and discussions on SDEs&UXDs collaboration. Similar to SLR, we utilized the keyword search strategy in post titles to retrieve relevant posts. In addition to the initial keywords listed in Table 1, we added keywords related to collaboration and tooling to make the search more specific; Table 2 shows additional keywords we used in the post search. Moreover, we did not use the UX keywords in UX forums and likewise for SE keywords, since by default, these forums will touch on the corresponding topics. Combining all keywords, the final string we used for keyword search is presented as follows: (forum_type == software_engineering ? UX : SE) OR Collaboration OR Tool.

Selection Process. Similar to RQ1 and RQ2, we established inclusion and exclusion criteria to identify the relevant information to answer RQ3. In specific, an included post must:

- **IC1:** be published in 2020-2023 to ensure the timeliness of our search results.
- **IC2:** contains pertinent information related to collaboration challenges, best practices, or collaboration tools.

Table 2. Additional keywords used to formulate searching string in the Post search. Combining keywords from Table 1, the final string we used for pose and issue search is presented as follows: (forum_type == software_engineering ? UX : SE) OR **Collaboration** OR **Tool**.





Type	Search String
Collaboration	“collaboration” OR “collaborate” OR “cooperation” OR “cooperate” OR “challenge” OR “gap”
Tool	“tool” OR “platform”

Simultaneously, we consider the following exclusion criteria exclude a post if the post:

- **EC1:** is not related to UXD-SDE collaboration.
- **EC2:** does not contain follow-up comments from either UXDs or SDEs.

Our initial keyword search resulted in 1,299 posts in four different online forums from 2020 to 2023. Two authors first checked the titles and main threads of all posts from the keyword search, ensuring that more than one author reviewed each post, then excluded posts following EC1 and EC2. The initial review resulted in 65 posts. Next, two authors conducted a second round of filtering by reading the entire post and associated comments to check whether the post met IC2, a third author then validated the result. This round of filtering identified 17 irrelevant posts, resulting in 48 posts from the four forums to address RQ3. Table 3 shows the post and issue search statistics.

Table 3. Posts and Selection Statistics. This table shows the number of posts selected after each stage during the search process, the average number of responses, and their distribution.

Forum/Project	Initial Posts	After Filter	Selected Posts	Avg. #Responses	#Responses Distribution
UX Stack Exchange	275	14	13	7.3	
UX Reddit	742	47	32	23.3	
SE Reddit	201	4	3	6.3	
Stack Overflow	81	-	-	-	-
Total	1299	65	48	18.8	

3.2.2 Investigating Issue Discussions in Open-source Project.

Last but not least, we searched on GitHub for popular projects with a graphical user interface (GUI), which might contain instances of UXD-SDE interactions and collaboration. We defined the following inclusion criteria:

- **IC1:** The project contains a GUI;
- **IC2:** The project contains issues labeled with UX.

Correspondingly, we also defined:

- **EC1:** The issue participants should include both UXDs and SDEs;
- **EC2:** The project does not consistently utilize issue labels.

We identify the background of each issue participant by examining their GitHub profile portfolio. Additionally, we conduct searches for their names to locate personal websites or LinkedIn accounts, if available. We initially reviewed the top 20 GitHub repositories with the most stars from Gitstar

Ranking² that meet IC1&2. The results show that the VS Code project³ is the only project among the top 20 that meets our criteria. Since its creation on September 3rd, 2015, the project has accumulated over 1.8k issues that have UX labels out of a total of 195k issues. The earliest issue marked with a UX label dates back to late 2015. To discover more appropriate projects, we conducted a manual review of the top 100 GitHub repositories with the highest number of stars that meet IC1&2. Subsequently, we identified additional four repositories, including *Excalidraw*,⁴ *Terminal*,⁵ *Godot*,⁶ and *Storybook*.⁷ However, we have discovered that none of the four projects are suitable for our investigation purpose. For example, although *Excalidraw* used the UI/UX label to categorize the issues, it is not used consistently and the majority of the issues are not labeled, so we filtered it out by EC2. Moreover, we found it challenging to determine whether a participant is a developer or a designer for issues in *Godot* and *Storybook*. Considering that both *Terminal* and *VS Code* are developed by Microsoft, it is plausible to assume they follow comparable development practices. Therefore, as an initial exploratory case study, our focus will be on *VS Code*, which was created in 2015, two years prior to the creation of *Terminal* in 2017. Future research could further distinguish project practices by examining individual repositories rather than by GitHub account.

We utilized the GitHub API to fetch those issues from the *VS Code* project. Specifically, we initially chose issues labelled as “ux” and those containing video/image attachments, representing instances of UI/UX design artifacts and discussions [68]. This process yielded a total of 1,019 issues. Then, we filtered out issues that violated the following criteria: the issue (1) has no follow-up comments; (2) is still open; (3) has “bug,” “duplicate,” “out of scope,” “debt,” or “as-designed” label; (4) involves no SDE or UXD, as they are not able to reflect sufficient communication and collaboration between SDEs and UXDs. During this stage, 259 issues have been filtered out, leaving us with 760 issues needing review.

To filter out issues irrelevant to RQ3, we applied a second round of checks to eliminate issues that do not contain collaboration/discussion between UXDs and SDEs. The first two authors labeled the 760 issues with Y (include), N (exclude), and NA (not sure). Another author reviewed issues labeled with NA to re-label them as Y or N. Finally, 157 issues were selected. Lastly, the first two authors then read each issue separately to decide whether it complements collaboration challenges or good practices.

3.2.3 Data Analysis.

For the selected posts and issues, two authors read them in detail and extracted information relevant to collaboration challenges and best practices. We then applied affinity diagramming to organize similar information into groups. Following that, we held meetings to address conflicts that arose during grouping and finally used refined groups to answer RQ3.

3.2.4 Limitations and Threats to Validity.

Every research methodology possesses constraints that pose potential threats to the validity and credibility of its findings. Despite the fact that we carefully employ selection methods, it is possible that we may have overlooked certain papers. Some reflections on collaboration between SDEs and UXDs might not be included only in peer-reviewed publications. To address this, we conducted one round of backward snowballing to include additional relevant studies. We also complemented our study by searching online forums and issue discussions of an open-source project on GitHub. In

²gitstar-ranking.com/

³github.com/microsoft/vscode/

⁴github.com/excalidraw/excalidraw/

⁵github.com/microsoft/terminal/

⁶github.com/godotengine/godot/

⁷github.com/storybookjs/storybook/

addition, our selection of included papers could potentially be biased because of the subjectivity of authors. We attempted to mitigate the potential bias by iteratively refining the selection criteria and cross-checking among the researchers. Each researcher indicates their decisions while reviewing the papers. Those results were cross-checked by another researcher, and discussion meetings were held to arrive at a consensus when there was a disagreement.

Another limitation to consider is that our data collection was primarily sourced from English-language forum sites. Consequently, the generalizability of our findings may be limited to the global communities of SDEs and UXDs. This lack of diversity in representation could potentially influence our observations regarding prevailing practices and the adoption of tools, as discussed in Section 3.2. Similarly, we only searched for popular repositories on GitHub with GUI, and filtered out those lacking UX labels in their issue lists. This selection process resulted in the inclusion of just one project from GitHub, respectively. It is important to note that this aspect of our research aimed to establish an initial understanding of the collaboration between UXDs and SDEs, and a comprehensive comparison among various projects as well as an assessment of collaboration efficiency were beyond the scope of this study. The findings from our work serve as a promising first step in identifying best practices and challenges of collaboration between UXDs and SDEs during the development lifecycle.

4 Results

Below we report the analysis results based on the 45 papers selected from the SLR, as well as the 48 posts ('Px' to represent the ID) and 157 issues ('Ix' to represent the ID) from the post and issue search.

4.1 Quantitative Overview

Publication Years. Figure 3(a) shows the distribution of publication years of all selected papers in the SLR. It demonstrates a clear increasing trend in selected papers from 2004 to 2008. Despite fluctuations from 2004 to 2018, it is noteworthy that about 90% of selected papers were published before 2018. This suggests a decline in the topic's popularity in recent years, indicating a waning interest among researchers.

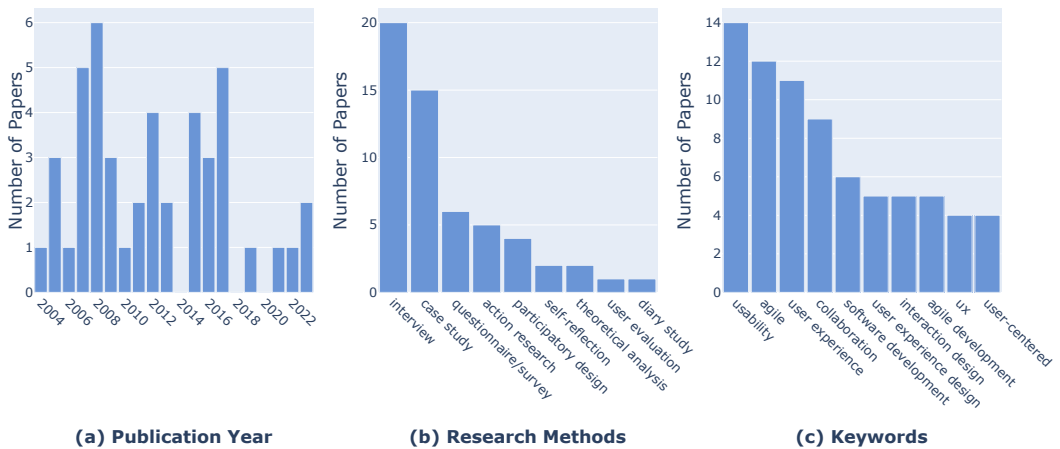


Fig. 3. Statistics summary of the selected papers in terms of their (a) Publication Years, (b) Research Methods, and (c) Author Keywords.

Research Methods. Figure 3(b) shows the frequencies of different types of research methods adopted by the reviewed papers. Noted that one paper may contain multiple research methods.

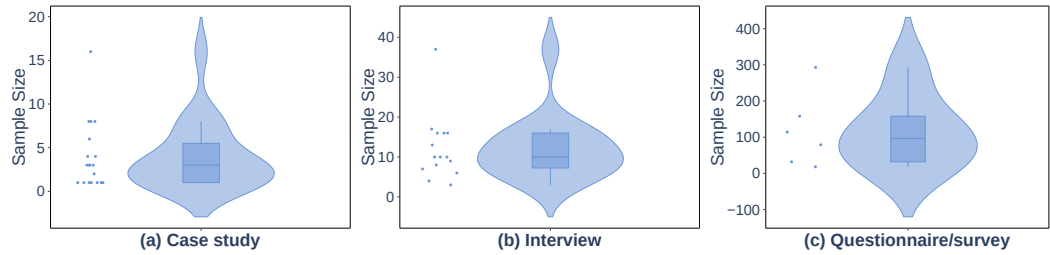


Fig. 4. Research Methods Sample Size: (a) Case study, (b) Interview, (c) Questionnaire/survey.

The three most widely utilized research methods are interview, ethnography/case study, and questionnaire/survey, used by 45%, 34%, and 14% papers, respectively. These three methods all involve human participants; the distributions of the participants’ sample size in the papers that applied these three methods are presented in Figure 4.

Keyword Distribution. We also summarized author keywords of selected papers, presented in Figure 3(c). The most frequently appearing keywords are usability and agile, which appear in 14 and 12 papers, respectively. All popular keywords shown in the figure are relevant to UX, SE, or collaboration. This indicates that the papers we selected for the SLR match our intended research direction.

4.2 (RQ1) Challenges of UX and SE Collaboration

In this research question, we aim to investigate the collaboration challenges between UX designers and SE developers. Among the 45 reviewed papers, 35 directly mentioned challenges regarding inefficient UXD-SDE collaboration. We classified these challenges into three main categories, each including several detailed sub-categories. We compile a summary of the challenges and the relevant academic papers and online posts that discussed these challenges in Table 4.

Table 4. Collaboration challenges identified from related papers and online posts. We present three key challenges which are further categorized into eleven detailed classifications.

Challenges	Detailed classification	Papers	Posts
Separate decision-making procedures	Suboptimal team arrangement due to geographical separation	[5], [22], [35], [55], [79]	-
	Isolated decision making	[36], [68], [72], [79], [81], [90], [115]	-
	Unbalanced involvement	[53], [55], [65]	[P4], [P7], [P9], [P10]
	Blurry boundary	[35], [55]	[P5]
	Lack of group awareness	[6], [9], [34], [35], [36], [67], [61], [59]	[P7], [P12], [P13], [P19], [P4]
Different professional behaviors	Educational disparities	[5], [6], [32], [59], [61], [68], [72], [81]	[P9], [P12], [P14], [P15] [P4], [P16], [P17]
	Methodology detachment	[22], [34], [33], [35], [53], [55], [44] [59], [67], [81], [90], [106], [110], [20]	[P1], [P2], [P3], [P4], [P6], [P7], [P8]
	Different artifacts	[35], [68], [72], [81], [84]	-
Lack of mutual understanding	UX de-emphasized	[6], [9], [34], [54], [44] [55], [61], [81], [110], [115]	[P19]
	Absence of trust	[6], [22], [54], [61], [66], [81]	[P12], [P7]
	Mismatch of expectations	[36], [67]	[P18], [P7]

4.2.1 *Separated Decision-Making Procedure.*

Suboptimal Team Arrangement Due to Geographical Separation. SDEs and UXDs are sometimes separated geographically, often with timezone differences [22, 35, 79]. This geographic separation adds to the discipline separation, contributing to difficulties in communication and further resulting in poor integration of UX design work and its implementation [55]. When the geographic separation creates timezone differences, it could even make one team unable to attend meetings, resulting in a breakdown in communication between two teams [5, 79]. This separation aggregates the isolated decision-making as we elaborate next.

Isolated Decision-Making. Effective communication between two groups during decision-making is important, as exchanging information among knowledgeable individuals can alleviate uncertainties and address questions. Non-participation in such discussions might result in postponed deliverables [36]. A prominent theme discussed in the literature is related to design decision-making in isolation [68, 72, 115]. For example, some design teams make early design decisions without informing the development team, making SDEs unaware of the current state of the design and devoid of opportunities to observe design evolution [68, 72, 115]. As a result, certain design choices might not be incorporated into the final implementation, as developers remain unaware of them [72]. Moreover, upon review by developers, some other designs may be technically infeasible, wasting designers' effort to redesign [6, 81].

A cause for this isolation is the untimely communication between two groups, where UXDs are not informed about whether their design is implemented, resulting in no user research and testing being implemented [79]. Another reason for this separation is the underestimation of UXDs by developers or their organizations. Some developers view UXDs as external entities and perceive UX work as insignificant external resources rather than part of the decision-making process, leaving the usability specialists outside the decision-making and planning process [90]. Consequently, they occasionally modify designers' work after the designer-developer handoff without proper communication with designers. Furthermore, situations arise where developers, facing unclear design documents from designers, rely on their past experience to implement the design rather than request communication with designers [79]. In some organizations, UXD's work is limited to the early design phase of the project. They lack the authority to participate in the development decision-making or even arrange meetings with SDEs [81]. All those challenges result in discrepancies between the initial design and the final implementation. The absence of designers' input during development meetings might potentially give rise to unidentified UX issues, subsequently requiring developers to invest additional time in correcting them afterwards, finally leading to delays in deliverables [79].

Unbalanced Involvement. In some cases, although collaboration teams are geographically close, interaction and communication gaps still exist due to unbalanced monetary compensation or uneven workload [53, 55, 65]. For example, several papers reported cases studied regarding the involvement of UX experts in the organization falling short [55, 65]. The difference in the number of UXDs and SDEs within the team increases the team's specialization. However, this specialization could result in challenges when it comes to distributing tasks equitably and flexibly, potentially leading to misunderstandings among team members during the handoff process [53]. Moreover, this deficiency in the number of UX Designers may also restrict testing and validation activities, making it challenging to sustain high-level collaboration with the development team [55]. This could finally lead to additional workload for UX designers and make them feel frustrated [65].

Similar concerns are observed from the forum analysis, where three discussions from two posts noted that developers often find themselves isolated from the design process with limited chance to contribute [P7, P9]. According to one designer, this issue arose due to the late involvement of developers, resulting in the inability to provide design feedback [P7]. However, another developer argued that they could not provide valuable feedback or make technical contributions even if

they were involved earlier [P7]. Three other posts stated developers often lack clarity on when to consider UX, leaving designers minimal time to offer feedback and make contributions during the implementation phase [P4, P9, P10].

Blurry Boundary. Another factor that intensifies the challenge of separated decision-making is the unclear demarcation between designers' and developers' responsibility [35, 55]. SDEs and UXDs often lack well-defined responsibilities concerning design or other associated activities within the organization; such absence usually leads to a deficiency of visibility of implementation activities and finally results in UX experts not having the opportunity to be involved in development-related decisions [55]. Regarding developers, their decision on design issues during the implementation phase usually results in re-implementation after UX designers suggest changes. Thus, time is wasted due to redoing works [35]. In one post, a designer specified that the boundary between designers and developers and the responsibility for specific tasks is unclear; he mentioned that it was unclear who should be responsible for the CSS colour variable in a previous project [P5].

Lack of Group Awareness. During the collaboration procedure, it is also observed that the two groups of experts are likely to be unaware of each other's work content and timeline, causing the inability to inform joint decisions and finally leading to missing deadlines [6, 35, 36, 61, 67]. For example, software developers and UX professionals often hold differing perspectives on the nature of UX tasks and the timeline of their integration [35, 61]. Many developers view the work of UX designers as primarily focused on user research and completing user stories, which often happen before development. However, UX designers still play a vital role in identifying usability issues and requesting developers to implement necessary changes [6, 67]. To be more precise, this disparity can be viewed as a deficiency in the shared understanding of the project's goal [61]. Moreover, this lack of consensus may create communication barriers between UXDs and SDEs. When knowledge is not shared among the team, team members will likely become isolated [34, 59]. Due to this lack of visibility and isolation, misunderstandings can arise between groups, exacerbating the communication gap and leading to isolated decision-making [9, 36].

It's important to highlight that the challenge of *lack of group awareness* stands out as a prominent recurring theme in our analysis of grey literature, appearing nine times across five posts [P7, P12, P13, P19, P4]. In two posts, four discussions addressed cases in which development teams made alterations to designers' work after the handoff, leading to a deviation between the final implementation and the original design [P7, P13].

4.2.2 Different Professional Behaviors.

Educational Disparities. SDEs and UXDs come from different educational and professional backgrounds and are equipped with distinct perspectives [5, 59, 61, 72]. This divergence often results in increased or overlooked tasks within the workflow, miscommunication, and misunderstanding [68, 72]. For instance, looking at it from the developers' point of view, numerous design concepts fail to take into account the technological limitations [6, 30, 32, 61, 68, 72, 81]. This oversight may introduce divergence that could further result in developers having to invest additional efforts into modifying the design on their own and/or waste effort for both groups [72]. From the viewpoint of UX designers, SDEs' lack of familiarity with usability testing led to a narrow focus on functionality testing rather than encompassing UX-related assessments [61].

Similarly, the analysis of the grey literature review reveals the same challenges, with them appearing in element instances across seven posts out of a total of 48 posts [P9, P12, P14, P15, P4, P16, P17]. According to five posts, the main challenge in the collaboration process arises from designers' limited technical understanding, which makes them unable to understand the complexity of their design, consequently creating challenges for developers during the implementation process [P9, P14, P15, P16]. Three posts highlighted a different aspect that developers lack a comprehensive

UX understanding [P12, P16, P17]. As a result, they struggled to provide feasible design insights and faced difficulties integrating designers' work. One participant, who used to be both a designer and a developer, noted that their divergent focuses caused conflicts during designer-developer collaboration: designers tend to prioritize usability aspects, while developers concentrate more on the functionality and coding aspects of the project [P4].

Methodology Detachment. The reviewed papers have noted that a gap exists in the approaches to development and design due to the fundamental distinctions between these two fields, which often results in a lack of clear communication between the two groups [22, 34, 110]. More precisely, in terms of pace, the iterative cycle of software development moves more rapidly compared to UX design [20, 55]. Estimating the time required for both categories of work, particularly the exploratory nature of UX design, is demanding, making the alignment of these two lifecycles a daunting task [22, 53]. This could lead to the usability-related insights becoming outdated by the time the report is completed [90, 106], or the product being released without the user interface having undergone comprehensive testing [33, 35, 67, 81]. This discrepancy is often caused by the mismatch between the software development lifecycle (specifically, the agile approach) and the process of UX design [67, 106]. Furthermore, the documentation of UX design provided to SDE is sometimes identified as inadequate for effective and accurate implementation [59].

The mismatch between the traditional software development cycle and the UX design process often poses a challenge when one team is transitioning their work to another. Although we do not identify these challenges from SLR, *inefficient designer-developer handoff* is one of the most frequent challenges mentioned by practitioners in their discussions, appearing nine responses in seven posts [P1, P2, P3, P4, P6, P7, P8]. Designers mainly mentioned strained communication [P1], uncertain handoff process [P3, P4], and lack of interaction during the handoff stage [P7, P6, P8]. Practitioners mentioned that the existing approach to the designer-developer handoff is only a transfer of responsibilities and materials.

Different Artifacts. Another aspect that highlights the distinction between the two groups is the artifacts they generate – ‘static and textual’ versus ‘dynamic and visual’ designs [35, 68, 72, 84]. For example, UX designers tend to utilize visual representations like images or videos of their screens to show their design [68]. In contrast, developers primarily depend on text, including programming terminology, to elucidate information about their implementations [72]. When sharing artifacts with the counterpart group, essential information can become lost or create confusion, making it difficult for the receiving team to fully comprehend [72]. As a result, this discrepancy often demands extra effort or results in a misalignment between the initial design and the eventual deliverable [35, 81]. Although this problem has been discussed over decades, it is still unclear what boundary objects for representing interaction can be shared by designers and developers [72]. Thus, time is wasted and misunderstandings arise due to the need for designers to produce multiple files to illustrate their designs, developers having to recreate design documentation, and developers misinterpreting the designs, leading to redundancy and rework [68].

4.2.3 *Lack of Mutual Understanding.*

UX De-emphasized. With distinct professional behaviors and separate decision-making, SDEs and UXDs are likely to develop their own understanding of the collaborative project [6, 55, 81]. This lack of mutual understanding usually results in an unwillingness to accept suggestions, criticism, or decisions from other groups [6, 81]. For example, some UX designers feel they are treated as “artists” by developers; thus, their questions and suggestions are usually ignored [81]. In the context of open-source software (OSS) projects, this situation could lead UX designers to have a shallow understanding of the project's goal and detailed procedures [55]. As a result, they might tend to criticize excessively rather than contribute by enhancing the codebase or resolving issues.

Ultimately, this could lead to their limited contribution to the project as a whole [115]. From the developer's point of view, their limited understanding and knowledge in the UX field often leads to a lack of appreciation for UX work [115]. They usually regard UX as trivial or lacking intellectual contribution and perceive it as the last phase of the project [6, 9, 54], thus reluctant to integrate UX work into their development and exclude UX designers out of the loop [110]. The absence of consensus and the undervaluing of UX contribute to relegating UX work to a lower priority, and in some cases, even dismissing it altogether [61, 81]. In addition, according to [P19], de-emphasizing UX not only happens between individual UX designers and developers but could also be an issue at the organizational level. When companies are development-oriented, they will ignore the importance of UX and focus more on functionality rather than usability. As a result, the management teams usually approve design changes made by development teams without adequate research or input from the design teams [P19].

Absence of Trust. Given that UX work is usually de-emphasized and even dismissed, UX designers may perceive it as a lack of respect [81]. Regarding communication or decision-making, this lack of mutual appreciation and trust between the two expert groups tends to make them defensive, avoiding addressing concerns from the other group [22, 61, 66]. The communication problem caused by lack of trust further isolated two groups of experts, making them unaware of the other group's efforts, resulting in an incomplete development process [6, 61]. Specifically, certain UX-related aspects of development, such as user research, were conducted late or even missing [61]. In addition, certain design works created by UX designers may not be presented in the final product [54], which leads to a waste of effort. P12 explicitly specifies that developers lack trust and are unwilling to engage in discussions with designers.

Mismatch of Expectations. The different education and methodology between UXDs and SDEs usually result in mismatched expectations between two groups of experts [36, 67]. Given that the agile model is technical and functional driven, it could be confusing for SDEs to know when a user story is "done" [67]. In particular, there is a lack of communication regarding how to support each other [36], which often leads to wasted planning effort and team members working overtime to meet deadlines. From our forum post analysis, we observed the same challenge where a designer stated that the expectations between the design and the development team lack clarification and mutual understanding, resulting in two groups of experts working towards different goals [P18]. Moreover, conflicting expectations could arise during the development phase, concerning the primary focus of the work. For instance, as cited in [P7], distinct priorities could emerge between design teams, who prioritize enhancing user experience (UX), and development teams, who prioritize optimizing code and functionality. These divergent emphases significantly contribute to conflicts encountered during the development process [P7].

4.3 (RQ2) Best Practices to Overcome Collaboration Challenges

In this section, we aim to identify strategies that could be used to overcome the challenges summarized in RQ1. Among all the 45 papers we collected, 25 proposed best practices that could be categorized into two high-level themes: (1) improving the processes and (2) enhancing communication, as presented in Table 5. In the subsequent sections, we elaborate on these best practices in detail.

4.3.1 Best Practice 1: Improving Development Workflow. is essential for aligning timelines and enhancing collaboration between SDEs and UXDs. This is a team-level best practice that reduces isolation (Section 4.2.1), integrates different professional behaviors (Section 4.2.2), and fosters better mutual understanding (Section 4.2.3).

UX Design Work Should Start Earlier Than Software Development. One major difference between SE and UX is their iteration speed (as described in Sec. 4.2.2), where software development

Table 5. Best practices identified from related papers and the corresponding collaboration challenges they addressed. We present two main best practices, which are divided into six sub-practices.

Best Practice	Detailed Classification	Papers	Addressed Challenges
Improve development workflow	UX Design work should start earlier than software development	[65], [72], [59], [86], [53], [55], [35], [106], [81], [5], [79]	Methodology detachment
	Enhance collaboration between UXD and SDE throughout the process	[72], [81], [74], [90], [55]	Isolated decision-making UX de-emphasized Absence of trust Mismatch of expectations
	Establish more effective boundary objects	[55], [79], [74], [72], [17]	Different artifacts
Enhance communication	Ensure regular and productive communication	[79], [72], [59], [53], [22], [34], [68], [65], [90], [33], [111], [36], [9]	Lack of group awareness Absence of trust
	Promote UX in development-drive teams	[86], [53], [109], [44]	UX de-emphasized
	Establish shared vocabulary	[68], [72], [35], [81]	Educational disparities

iterates faster than UX [33, 53, 67, 90], which leads to insufficient communication, rework, and delays. To bridge the disparity in timing, nine papers propose that UX activities, particularly UX evaluation, should commence and be completed earlier to synchronize with the software development schedule during the project’s initial phase [35, 53, 55, 59, 65, 72, 81, 86, 106]. The primary cause for this gap is the tendency of UX designers to employ a ‘big design upfront’ approach during the early stages of the design process [5], which leads to too much design work done ahead and redesigning afterwards. Instead of gathering all requirements and including all usability testing at the early phase of the project, UXDs should apply “little design upfront” to create only basic project structures in sprint 0 and increment them in future sprints. Similarly, other studies have also noticed that current practice involves too much upfront design work and recommended designers take the “sprint 0” strategy: create UX vision among the team while preserving the flexibility of UX work [35, 59, 72, 79, 86]. Some papers even proposed that design work should always be one sprint ahead to ensure timeliness and flexibility [53, 55].

On the other hand, developers also prefer the early involvement of designers [65, 81]. Involving UXDs at the early stage can help SDEs avoid significant changes in later sprints [81]. Moreover, it can also facilitate early collaboration to reveal different thinking habits and avoid future misunderstandings [65].

Enhance the Collaboration Between UXD and SDE Throughout the Process. As described before, the current workflow of software development and UX design suffers from a lack of seamless integration, resulting in isolated decision-making (described in Section 4.2.1) and communication breakdowns during collaboration (described in Section 4.2.3), ultimately causing workflow inefficiencies. To address these issues, previous efforts suggested a more effective integration of both groups of stakeholders into each other’s work processes. In our analysis of forum posts, we noticed consistent recommendations for this best practice: a designer emphasized that the handoff process should involve prolonged cooperation and ongoing communication (P8).

Involving SDEs in UX design. Involvement of SDEs during the initial design phase can facilitate the creation of complex interactions and enhance developers’ understanding of design principles [72]. Furthermore, the collaborative team will become better equipped to minimize information gaps, identify edge cases, and establish a well-defined technical scope for designers. Similarly, developers should engage in the design process by collaborating with designers to create sketches and prototypes [81]. This collaborative approach enables SDEs to gain a more comprehensive grasp of the UX

vision. Another study also suggests that SDEs should participate in UX evaluation and testing [74]. In the study, SDEs engaged in the usability testing session, taking on the role of technical assistants to facilitate the testing process. Their recorded notes and comments offered valuable technical insights to the designers.

Involving UXD in development. Involve UXDs in SE activities is also recommended [55, 72, 90]. It is beneficial for UXDs to have technical skills and code access to contribute to development [90]. Similar findings were observed by Maudet et al. [72], highlighting that allowing designers to modify code can avoid misunderstandings and reduce iterations [72]. Designers should also participate in development sprint planning meetings; this will provide them with improved control and greater transparency throughout the UX process, enabling them to maintain effective collaboration and communication with the implementation team [55].

Establishing More Effective Boundary Objects. One challenge mentioned in Section 4.2.2 is that SDEs and UXDs employ distinct types of artifacts, resulting in miscommunication and isolation. To bridge this gap, it is crucial to establish a clear definition of the boundary objects that are mutually shared by both groups of experts to support subsequent interactions and establish shared understanding [17]. Many studies emphasize the importance of sharing design artifacts as boundary objects to facilitate collaboration between SDEs and UXDs. For example, sharing the *UI proposal and UX design* can facilitate early evaluation through comments and discussions [55]. Similarly, sharing *user research and testing results* can help to clarify misinterpretation and enhance designers' participation [79]. Further, the incorporation of *paper prototypes* – initially used for usability testing, into UX storyboards provides a comprehensive guide that aligns the entire team with the project's vision [74]. In addition to design deliverables created by UXDs, contributions from developers should also be valued. For instance, while designers can provide visual representations capturing interaction snapshots, developers offer *component diagrams describing connections between primitive graphical elements and functions*. These artifacts can then be combined to help the team reach a complete and shared understanding [72].

4.3.2 Best Practice 2: Enhance Communication. In addition to efforts aimed at enhancing workflow, another aspect to consider is enabling improved communication. This can foster better mutual understanding (Section 4.2.3) and eliminate the isolated decision-making (Section 4.2.1).

Ensuring Regular and Productive Communication. Twelve papers proposed that communication between SDEs and UXDs should be explicit [53, 59, 72, 79], as well as active and frequent [22, 34, 53, 65, 68, 90]. Developers are encouraged to ask explicitly for design specifications and ideas [72, 79], while designers should actively provide details to resolve misunderstandings and facilitate integration [53, 59, 72]. Similarly, designers should request clarification and confirmation on technical issues [72]. When it comes to the development process, explicit communication between two groups of experts can also help both groups be aware of the project timeline, avoiding late or missing deliverables [53]. In terms of active and frequent communication, collaborative teams are encouraged to communicate daily [22, 53], and face-to-face or live meetings are valued [34, 65, 68]. Engaging in discussions and face-to-face debate enables SDEs and UXDs to avoid missing information, address edge cases, and develop better products [34, 72]. Moreover, communication tools and platforms are another aspect of consideration [65, 90]. Studies also demonstrate that practitioners prefer flexible and lightweight tools, as they believe such tools can enhance communication by making it more accessible and convenient [65]. Despite being explicit and active, the relationship between SDEs and UXDs is also a critical factor affecting communication [33]. The foundation of such communication lies in the willingness of both groups to collaborate [36, 111]. According to interviews conducted by Bach et al. [9], UXDs should build trust by reputation, position and pointing to data, while SDEs should respect UXDs and make them feel valued [9].

Promote UX in development-drive teams. Another challenge discussed in Section 4.2.3 is the lack of mutual understanding between SDEs and UXDs, where SDEs usually de-emphasize and overlook UX. To overcome this challenge, several studies have proposed to promote the position of UX in the overall development process [44, 53, 86, 109]. It is beneficial to integrate UX closer with the development team; to achieve this, all participants should reach a consensus regarding the positive impacts of UX and comprehend the UX vision through training or synchronization with users [44, 86]. An alternative way to promote UX is through workshops and presentations; UX workshops have the potential to cultivate a shared UX understanding among teams, while the presentations by the UXDs can widen the perspectives of developers [53, 109].

Establishing a Shared Vocabulary. Work culture and background differences are another designer-developer collaboration challenge (Section 4.2.2). To overcome this challenge, several studies suggest that both groups of experts should become acquainted with each other's terminology to guarantee effective communication with minimal confusion [68, 72]. An observation study conducted by Maudet et al. [72] found that designers started to incorporate mathematical concepts while developers began formulating ideas using UX widgets to bridge the vocabulary gap. They also explored specific examples to validate the consistency of terminologies. Some other papers further suggest that SDEs and UXDs should consider each other's work tradition and knowledge domain [35, 81]. Instead of solely depending on management efforts, they should clearly understand each other's roles and expectations. Nørgaard and Hornbæk [81] discovered from a case study that having practitioners with both domain expertise and decision powers is a critical factor of success [81].

4.4 (RQ3) Current State of UX & SE Collaboration

In this section, we investigate how the findings from RQ1 and RQ2 were reflected in real-world settings. In particular, we study the existing tooling support to assess whether the challenges identified in RQ1 have been addressed, if the best practices summarized in RQ2 have been fully implemented, and if there are still areas where improvements are needed. We examined four online forums frequented by UX designers and software developers, focusing on discussions relevant to our research. We identified a total of 14 collaboration tools and/or platforms and highlighted challenges identified in Section 4.4.1 for some of these platforms discussed online. Additionally, we performed an initial case study on the VS Code project to complement our analysis of forums (Section 4.4.2), aiming to explore how UX designers and software developers collaborate within a prominent and well-established real-world open-source project, and to assess their adherence to recognized best practices described in Section 4.3.

4.4.1 Collaboration Tools and Platforms.

Having tooling support is essential for the collaboration process to guarantee efficient sharing of information and communication [11, 21]. In the post-search phase across the four forums (i.e., UX Stack Exchange [29], Reddit UX [94], Reddit SE [93], and Stack Overflow [28]), we specifically focused on posts that discussed tools and platforms. We aimed to gather insights and opinions from professionals and practitioners. We identified 26 posts that are relevant to our focus in total. Figure 5 displays the frequency of mentions for each tool/platform within these posts. Among these posts, we only identified eight posts focused explicitly on the pros and cons of five tools from the collaboration perspective, including Figma [37], Adobe XD [1], Axure [8], Protopie [87], and Miro [75]. Other posts primarily emphasize the technical functionality or accessibility of those tools. We carefully review the compiled pros and cons of each tool, comparing them with the collaboration challenges outlined in Section 4.2 to identify whether any of the tools effectively address specific

challenges. Table 6 shows a mapping between the tools and corresponding challenges identified in RQ1. We use ✓ to represent the challenge that can be alleviated by a tool and ✗ to show that the challenge can not be alleviated.

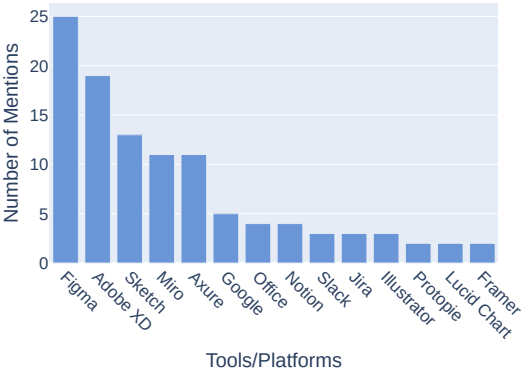


Fig. 5. Popularity of collaboration tools and platforms. The Y-axis shows the frequency of the appearance in collected posts. This figure only presents tools and platforms that appear more than once.

Table 6. Matching the summarized collaboration challenges summarized in Section 4.2 with the relevant features provided by each tooling support in order to address and mitigate these challenges. ✓ represents the challenge can be alleviated and ✗ means the challenge can not be alleviated.

		Tools				
		Figma	Adobe XD	Miro	Axure	Protiole
Challenges	Suboptimal team arrangement					
	Isolated decision making			✓		
	Unbalanced involvement					
	Blurry boundary	✓				
	Lack of group awareness	✓	✗		✓	✓
	Educational disparities					
	Methodology detachment	✓		✗		✓
	Different artifacts	✓		✗		
	UX de-emphasized					
	Absence of trust					
	Mismatch of expectations		✓			

Next, we provide a condensed overview of the collaboration-related comments made by practitioners concerning these five tools/platforms.

Figma is the most popular tool among SDEs and UXDs that has been mentioned in seven out of eight posts for its highly effective *cross-team collaboration features*. It is recommended for designer-developer collaboration for its version control functionality and the “lock and branch” feature [P6], which can effectively specify designs to developers, enabling them to understand their assigned tasks clearly and increase group awareness, and allows designers to continue working on their designs without affecting the work handed to developers. Other posts noted that Figma involves the whole team in the project with different roles [P20, P21, P22, P23, P24]. From the developers’ perspective, Figma links design and development artifacts by allowing them to access and inspect code on top of the design. It also offers style codes and colour variables, facilitating handoff and enhancing overall development efficiency [P21, P23, P24]. Moreover, its artboard feature enables

developers and stakeholders to comment on their designs, simplifying the feedback-gathering process [P23]. However, some developers complained that Figma’s user interface is perplexing and has a steep learning curve [P4].

Adobe XD significantly simplifies the design presentation process by allowing designers to generate prototypes with a simple button click, allowing the collaboration team to quickly understand the expectation [P20]. However, the poor document syncing and low parallelization make it hard for UXDs and SDEs to quickly be aware of what has been done by the other group. **Axure** is recommended for its interaction design tools, making communication and interaction between SDEs and UXDs more accessible [P25]. Similarly, **Protopie** is nominated for its ability to facilitate communication and design handoff; it enhances collaboration and mutual understanding between two groups by operating within a design logic that makes designers’ work easier to understand [P25]. Some designers struggle to find a single software that meets their diverse daily requirements, so they employ a combination of collaboration tools [P4]. For example, **Miro**, though not good for prototyping or designer-developer handoff, is used for problem-solving and decision-making as a complement to Figma [P4]. This will, to some extent, make their work more convenient. However, when it comes to collaborating with developers, this complex situation can pose challenges for developers, making it difficult for them to dive in and work on top of designers’ design.

4.4.2 Understanding the Interaction Between UXDs and SDEs via Issue Discussion From The VS Code Project.

VS Code is a source code editor owned by Microsoft, and it is an open-source project hosted on GitHub. Though the development team might have internal communication channels and meetings, we can still trace designer-developer collaboration by reviewing GitHub issues as documented within the project’s Wiki page.⁸ Figure 6 illustrates the yearly occurrence of UX/UI issues recorded since the project’s creation in September 2015.

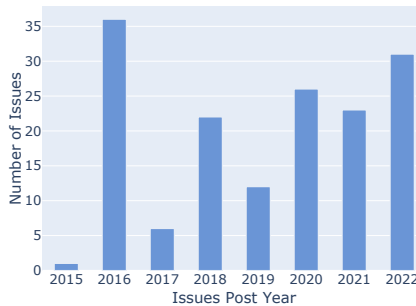


Fig. 6. Number of UX-label issues per year since late 2015 in the VS Code project.

By reviewing the discussion in each collected issue, we compile the following collaboration scenarios. Some of these scenarios conform to the best practices outlined in Section 4.3 (RQ2) and/or could be regarded as best practices capable of mitigating the challenges delineated in Section 4.2 (RQ1). First, designers and developers collaborate closely throughout the issue time span, which aligns well with the best practice of *enhance collaboration between UXD and SDE throughout the process*. Specifically, we observe that designers and developers initiate their collaboration during the ideation stage of an issue by clearly outlining ideas and design proposals [I1, I2]. For example, one designer stated, “From a design perspective, I think we can establish some principles for how

⁸<https://github.com/microsoft/vscode/wiki/UX/127f1567d5155bcd4dab20dee8f170e82f4bc63a>

the interactions should work” and *“Things that are unclear so far”* [I1]. Developers then actively engage by asking for clarification and providing feedback on further discussions [I3, I2, I4, I10]. For example, one developer asked, *“All sounds good to me except the ‘Go to running cell’. Maybe I’m misunderstanding that one though. Is that only available when actually running?”* [I2] During the implementation phases, designers monitor the project progress and address inquiries or concerns from developers [I5]. Ultimately, designers verify that the final implementation aligns with their original design by adding the “verified” label, ensuring consistency with their intended vision [I6, I13].

Another scenario we observed is the respect between SDEs and UXDs. In terms of communication, this is represented by the cordial relationship between two groups of experts. Through our findings, we have discovered the mutual recognition and assistance demonstrated by both groups. This is done by either direct follow-up comments such as *“Love that we are tackling this problem!”* or *“I like where we are heading towards!”* to show merits or positive reaction emojis such as 🍌 under the original thread [I1, I8, I9]. By observing the dynamics of the interaction, it becomes evident that team members *exhibit a shared sense of respect for each other*, which could directly eliminate the challenge of *absence of trust* summarized in Section 4.2 (RQ1).

In terms of cooperation, when SDEs intend to modify design works, they first notify and discuss designers rather than make changes without acknowledging them [I7], showing that *UX is promoted in the development team* and the team members are exerting themselves to *ensure group awareness*. For example, one developer suggested that *“I am not familiar with the details of the interactive window and how its editor is created but this should be possible”* and used *“to verify”* to start a discussion with designers before making changes [I7]. Another notable finding is that both groups clearly understand each other’s field and background. For instance, developers can showcase their ideas by creating simple prototypes, indicating their capacity to convey design concepts [I15]. Designers have basic technical knowledge and understanding of the codebase, enabling them to locate UX bugs or even make coding contributions [I11, I12, I14]. This mutual understanding and proficiency across disciplines mitigate the challenge of *Educational disparities*, contributing to smoother collaboration and effective communication between the two groups of experts.

However, we also identified several collaboration challenges from those issues. For instance, although design artifacts are included in the issue posts, most of the time when developers want to comment on the design artifacts, they are not able to add annotations freely but can only react with emojis, leave text comments or regenerate new visual artifacts under the original thread [I3], indicating that the current platform is inadequate for facilitating efficient communication involving non-textual artifacts and may lead to misunderstandings. Moreover, geographical separation causes further inefficiency in communication. Unlike face-to-face discussions or meetings where team members can provide instant reactions and feedback, sometimes designers do not participate in further discussion [I17]. For the VScode project, we noticed that most developers are based in Zurich Switzerland, whereas two major designers are based in Seattle, USA. The time difference between them causes a delay in responses [I16].

5 Discussion

Through our comprehensive examination of the existing academic literature and non-academic sources, we have pinpointed (1) three significant challenges faced in collaborative efforts between SDEs and UXDs, (2) two main categories of best practices that hold the potential to alleviate these challenges, and (3) how current collaboration tools and practices reflect these challenges and best practices. Figure 7 provides an overview of these findings, representing their connections. Below, we discuss our contributions to advancing previous SLRs on similar topics, as well as the implications of our findings on tooling support, education, and future research.

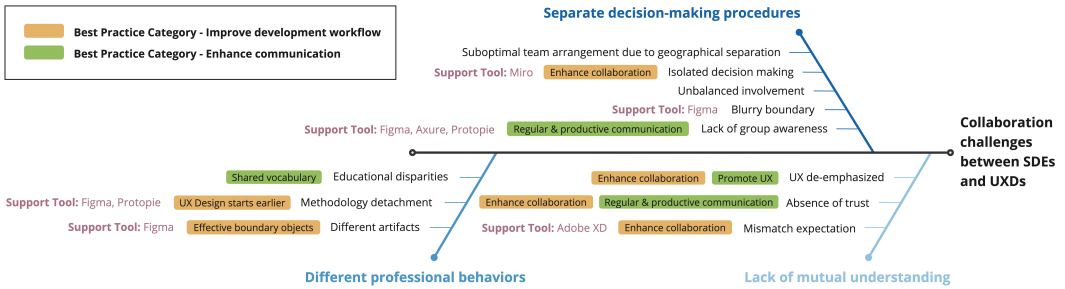


Fig. 7. Fishbone diagram of collaboration challenges, corresponding best practices, and supporting tools.

5.1 Addressing Research Gaps and Advancing Prior SLR Studies

Compared to previous SLR studies, our work has several unique contributions. First, most previous SLR studies on UXD-SDE collaboration have examined the integration of Agile development methodologies and user-centered design, emphasizing process and management aspects [52, 58, 70, 101, 103]. In contrast, our research broadens the focus to include software development as a whole, without limiting it to specific development models. Although the Agile model appears promising, there is no evidence to suggest it has been universally adopted across all software development contexts, including the crucial and influential realm of open-source software development [46, 77]. Agile literature and frameworks emphasize the necessity for teams to be small, stable, and whole, but it is unrealistic to achieve these goals in many software development environments [104]. Our expanded focus on UXD-SDE collaboration has contributed to identifying challenges and best practices that are important for both agile teams and beyond. For example, we identified that the challenge of *UX de-emphasized* is rarely discussed in papers focused on agile teams but more frequently investigated in open-source software projects [6]. At the same time, we also found that papers touching on agile, open-source, and traditional software development models have discussed many common issues regarding UXD-SDE collaboration. These results highlighted the importance and prevalence of collaboration issues between UXDs and SDEs. Regarding this commonality, our review also included papers that are closely related but were published in venues not considered in earlier research, such as the Journal of System Software and Information and Software Technology. This expanded collection has contributed to identifying new challenges, such as blurry boundary [35] and mismatch of expectations [67].

Moreover, our review encompasses a broader scope of resource collection, covering literature from important venues in both HCI and SE and triangulating it with real-world discussions in forums and open-source issue trackers. When it comes to the topic of UXD-SDE collaboration, the boundary between the fields of HCI and SE is blurry, as is the line between practice and research in this area. We found that papers published on this topic in HCI and SE venues discussed similar problems. The differences in their perspectives are much less prominent than the similarities in the problems these two fields targeted and the findings the studies revealed. The previous SLRs have never adopted this broad angle, encompassing both academic literature in multiple fields and real-world scenarios. As a result, our study has identified topics that earlier SLR studies overlooked, particularly related to human factors that tackle specific challenges, such as a lack of mutual understanding, absence of trust, and mismatched expectations, as well as the corresponding best practices, including improved communication, promotion of UX in development-driven teams, and the establishment of a shared vocabulary. By analyzing online discussions and open-source

project repositories, we also used insights from practitioners with expertise in UX and SE fields to triangulate previously identified challenges and understand their practical implications.

5.2 Better Tooling Support to Integrate Both SE and UX Artifacts.

Based on our literature review, a considerable number of identified challenges are related to the isolation between SE and UX, especially their processes and artifacts. While boundary artifacts are often defined between SDEs and UXDs, the iteration of the artifacts is often done in the corresponding field, with little input from the counterpart. When exploring open-source platforms for information on collaborations between UXDs and SDEs, we also discovered a scarcity of projects that actively emphasized discussions about UX-related issues. Likewise, conversations among UX designers on Reddit revealed that these designers often do not feel comfortable participating in code reviews on GitHub because of the barrier faced by those who are not developers [91]. This observation implies the underutilization of GitHub among UX designers or a lack of consideration for UX-related quality attributes during the open-source development process, echoing findings from previous work [49, 89, 96, 107]. In future research, it would be valuable to explore tool support that lowers the barrier for experts from both fields, integrates SE and UX artifacts, and facilitates collaboration between UXDs and SDEs. This integration could be achieved by considering the two potential scenarios:

Scenario 1: Decentralized Platforms With Seamless Integration. Improved tooling support is needed to seamlessly integrate platforms used by software developers (e.g., GitHub, GitLab) with those used by UX designers (as described in Section 3.2). Since UX artifacts are often visual and not supported in software development platforms, UX designers resort to suboptimal workarounds, such as sharing and commenting on screenshots, to convey their ideas [2, 96]. Despite efforts to integrate these two types of platforms (e.g., GitLab Design Management⁹), there remains significant room for improvement. For instance, Sketchboard allows for the embedding of designs into markdown files in GitHub repositories [102] but is not primarily designed for enhancing collaboration purposes. Figma also introduced FigJam [38], which advocates for the integration of collaborative software development and collaborative UX/UI design. Since the feature has been recently introduced, it remains uncertain whether it adequately addresses the needs and resolves the collaboration challenges.

Scenario 2: Centralized Platforms. Alternatively, an ideal platform should accommodate both UX designers and software developers within a shared workspace, despite their distinct objectives, by facilitating the seamless exchange of boundary objects [3], thereby eliminating the need to switch between disparate websites and incompatible artifacts. For example, the current software development tools such as GitHub and Jira include sufficient features to support various SE roles, such as developers, quality assurance experts, and project managers. New features in those tools can be considered incorporating the tasks typically performed by UX roles, such as user researchers and UX designers. Several discussions indicate that using multiple platforms for collaboration is sub-optimal. For instance, a UX designer inquired on Stack Overflow about the process of uploading Figma files directly to GitHub to enhance team collaboration; however, subsequent feedback revealed that this approach inadvertently obstructed the timely editing of Figma files when needed, proving counterproductive [82]. Similarly, inquiries on Reddit underscore the ongoing challenge of streamlining the integration between design and development processes, revealing a lack of universally accepted solutions to address this issue [92]. These problems might be addressed with a centralized platform that puts SE and UX tasks and artifacts together.

⁹https://docs.gitlab.com/ee/user/project/issues/design_management.html

5.3 Educators From HCI, CSCW, and SE Fields Need to Work Together to Design an Integrated and Systematic Training Process.

Out of the 45 papers gathered, only a single paper examined the collaboration between UXDs and SDEs in a classroom setting [4]. This suggests that there may be a necessity to focus on developing more effective training programs to raise awareness of better collaboration. Moreover, many challenges we identified in the literature review, such as *different professional behaviors* and *lack of mutual understanding*, are rooted in discrepancies in the mindsets of SDEs and UXDs. These mindsets, in turn, originated from the education of SDE and UXD practitioners. SE education typically focuses on software development best practices, such as software design and the agile model, which offer numerous advantages, but they frequently neglect concerns associated with UX design [4]. For example, while the proposed curriculum [100] for software engineering programs has acknowledged collaboration with stakeholders, the documentation does not explicitly address the decision-making process and potential trade-offs that can arise during such collaborations. Moreover, even though an increasing number of software engineering courses started to incorporate UX into software development, the short sprints limited by course length make fitting UX into software development challenging [4].

On the other hand, UX and HCI education often addresses user-centered design-related topics, such as user inquiry, prototyping, and usability testing, omitting subjects like software process and development concerns. In the HCI curriculum defined by the ACM SIGCHI [51], software developers (SDEs) were merely mentioned, let alone the aspect of collaboration between these two expert groups. Our results indicate that the fundamental solution for the SDE-UXD collaboration issue might be to incorporate the counterparts in the education of both fields. We believe that such an integration in training would ripple out and result in synchronization in practice. Developers and designers should collaborate in creating enhanced training programs for upcoming generations, aiming to improve the efficiency of collaboration within interdisciplinary teams.

5.4 Future Research Needs to Investigate Better Strategies to Align UX and SE Lifecycles and Activities.

Earlier studies have put forth the idea of integrated software development lifecycles [10, 26, 39, 79, 86, 101, 106], where UX design could be conducted from Sprint 0 before the development starts and also has frequent collaboration points to communicate to ensure the group awareness and avoid misunderstanding. Alhammad and Moreno [4] summarized a case study regarding the lesson learned on integrating UX lean into the agile development model via a software engineering course. Similarly, Sfetsos et al. [99] also reported a case study with a similar setting with the focus on the defects decrease. However, anecdotal evidence still indicates the persistence of inefficient collaboration and challenges outlined in RQ1 (Section 4.2). Subsequent research may necessitate a reexamination of lifecycle designs and the execution of field studies in diverse real-world settings encompassing various organizational structures. These efforts can shed light on trade-offs and support deliberate decision-making in the collaboration between UXDs and SDEs.

The challenges intensify significantly when creating software tailored to specific domains. As the challenges of collaboration between UXDs and SDEs remain unresolved, the situation becomes increasingly dire and complex when additional stakeholders become involved, making resolution more challenging. With the rising popularity of artificial intelligence (AI) and its integration into numerous software systems, researchers in the SE communities have explored the collaboration difficulties associated with constructing AI-enabled software projects [78]. Similarly, in the HCI communities, researchers also investigated how to improve the collaboration efficiency between UX designers and AI experts [69, 113]. Nevertheless, we did not find papers discussing the three

groups of experts working together. We aspire that this research can serve as a catalyst for fostering collaboration between and beyond SDEs and UXDs, facilitating the refinement of best practices and the development of efficient collaboration processes. The same idea would be applicable to any domain-specific software, such as improving the usability of scientific software [88], where collaboration among UXDs, SDEs, and research scientists is essential.

6 Limitations

All research design comes with limitations. Therefore, readers should exercise caution when generalizing findings beyond the boundaries defined by the methods employed. Firstly, despite conducting extensive searches across five academic databases, we may have overlooked some pertinent papers. To address this, two researchers independently retrieved literature data to ensure the most comprehensive data collection. Additionally, forward and backward snowball sampling were employed to thoroughly review moderation research. Secondly, our research primarily concentrated on the collaboration between UXDs and SDEs, excluding studies from other domains. Subsequent research endeavours could explore the collaboration between UXDs and SDEs in the development of domain-specific software, such as AI-based software or scientific software. This broader investigation is warranted as collaboration in these contexts often involves domain experts. Another limitation is that our case study on open-source project issues only focuses on the VS Code project as an initial exploratory case. As a result, the findings may not be applicable to a broader range of projects. Future research should consider examining additional projects to expand the scope and validate the generalizability of the results.

7 Conclusion

We presented an SLR about collaboration challenges between SDEs and UXDs, followed by a post and issue search to examine the current state of collaboration. By analyzing selected papers, we identified three key collaboration challenges and six best practices. We also complement the SLR with an analysis of online forum posts and issue discussions to examine the current state of designer-developer collaboration. Our findings can be applied beyond the partnership between SDEs and UXDs, such as collaboration with product managers and machine learning engineers. Those good practices and interventions suggested also act as a reference for several future research directions.

8 Data Availability

All papers, posts, and issues we reviewed are available in this repository: <https://github.com/FORCOLAB-UofToronto/CSCW2025-WhoIsToBlame>.

Acknowledgments

This work is partially supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), RGPIN2021-03538 and RGPIN2018-04470.

References

- [1] Adobe. 2023. Adobe XD Platform. <https://adobexdplatform.com/>
- [2] Vishakha Agrawal, Yong-Han Lin, and Jinghui Cheng. 2022. Understanding the Characteristics of Visual Contents in Open Source Issue Discussions: A Case Study of Jupyter Notebook. In *Proceedings of the 26th International Conference on Evaluation and Assessment in Software Engineering (EASE '22)*. Association for Computing Machinery, New York, NY, USA, 249–254. doi:10.1145/3530019.3534082
- [3] Sanne F Akkerman and Arthur Bakker. 2011. Boundary Crossing and Boundary Objects. *Review of educational research* (2011), 132–169. doi:10.3102/0034654311404435
- [4] Manal M. Alhammad and Ana M. Moreno. 2022. Integrating User Experience into Agile: an Experience Report on Lean UX and Scrum. In *Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET '22)*. Association for Computing Machinery, New York, NY, USA, 146–157. doi:10.1145/3510456.3514156
- [5] Ohoud Almughram and Sultan Alyahya. 2017. Coordination Support for Integrating User Centered Design in Distributed Agile Projects. In *2017 IEEE 15th International Conference on Software Engineering Research, Management and Applications (SERA)*. 229–238. doi:10.1109/SERA.2017.7965732
- [6] Morten Sieker Andreasen, Henrik Villemann Nielsen, Simon Ormholt Schröder, and Jan Stage. 2006. Usability in Open Source Software Development: Opinions and Practice. *Information Technology And Control* 35, 3 (2006). doi:10.5755/j01.itc.35.3.11776
- [7] Craig Anslow. 2014. Reflections on Collaborative Software Visualization in Co-located Environments. In *2014 IEEE International Conference on Software Maintenance and Evolution*. 645–650. doi:10.1109/ICSME.2014.115
- [8] Axure. 2023. Axure RP - UX prototypes, specifications, and diagrams in one tool. <https://www.axure.com/>
- [9] Paula M. Bach, Robert DeLine, and John M. Carroll. 2009. Designers Wanted: Participation and the User Experience in Open Source Software Development. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. Association for Computing Machinery, New York, NY, USA, 985–994. doi:10.1145/1518701.1518852
- [10] Kristine Bang, Martin Akto Kanstrup, Adam Kjems, and Jan Stage. 2017. Adoption of UX Evaluation in Practice: An Action Research Study in a Software Organization. In *Human-Computer Interaction - INTERACT 2017*. Springer-Verlag, Berlin, Heidelberg, 169–188. doi:10.1007/978-3-319-68059-0_11
- [11] Rajiv D. Banker, Indranil Bardhan, and Ozer Asdemir. 2006. Understanding the Impact of Collaboration Software on Product Design and Development. *Information Systems Research* 17, 4 (2006), 352–373. doi:10.1287/isre.1060.0104
- [12] Youssef Basil. 2012. A Simulation Model for the Waterfall Software Development Life Cycle. arXiv:1205.6904
- [13] K. Beck, M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, ..., and J. Kern. 2001. Manifesto for Agile Software Development. Agile Alliance. <http://www.agilemanifesto.org/>
- [14] Elizabeth Bjarnason, Baldvin Gislason Bern, and Linda Svedberg. 2022. Inter-Team Communication in Large-Scale Co-located Software Engineering: A Case Study. *Empirical Software Engineering* 27, 2 (2022). doi:10.1007/s10664-021-10027-z
- [15] B. W. Boehm. 1988. A Spiral Model of Software Development and Enhancement. *Computer* 21, 5 (1988), 61–72. doi:10.1109/2.59
- [16] David Boland and Brian Fitzgerald. 2004. Transitioning from a Co-located to a Globally-distributed Software Development Team: A Case Study at Analog Devices Inc. In *Third International Workshop on Global Software Development (GSD 2004) in 26th International Conference on Software Engineering*. IEEE. doi:10.1049/ic:20040303
- [17] Judith Brown, Gitte Lindgaard, and Robert Biddle. 2008. Stories, Sketches, and Lists: Developers and Interaction Designers Interacting Through Artefacts. In *Agile 2008 Conference*. 39–50. doi:10.1109/Agile.2008.54
- [18] Anders Bruun. 2010. Training Software Developers in Usability Engineering: A Literature Review. In *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries (NordiCHI '10)*. Association for Computing Machinery, New York, NY, USA, 82–91. doi:10.1145/1868914.1868928
- [19] Leydi Caballero, Ana M. Moreno, and Ahmed Seffah. 2016. How Agile Developers Integrate User-Centered Design Into Their Processes: A Literature Review. *International Journal of Software Engineering and Knowledge Engineering* 26, 08 (2016), 1175–1201. doi:10.1142/s0218194016500418
- [20] Åsa Cajander, Marta Larusdottir, and Johannes L. Geiser. 2022. UX Professionals' Learning and Usage of UX Methods in Agile. *Information and Software Technology* 151 (2022), 107005. doi:10.1016/j.infsof.2022.107005
- [21] Luc Cassivi, Élisabeth Lefebvre, Louis A. Lefebvre, and Pierre- Majorique Léger. 2004. The Impact of E-collaboration Tools on Firms' Performance. *The International Journal of Logistics Management* 15, 1 (2004), 91–110. doi:10.1108/09574090410700257
- [22] Stephanie Chamberlain, Helen Sharp, and Neil Maiden. 2006. Towards a Framework for Integrating Agile Development and User-Centred Design. In *Proceedings of the 7th International Conference on Extreme Programming and Agile Processes in Software Engineering (XP'06)*. Springer-Verlag, Berlin, Heidelberg, 143–153. doi:10.1007/11774129_15

- [23] Laura Dabbish, Colleen Stuart, Jason Tsay, and Jim Herbsleb. 2012. Social Coding in Github: Transparency and Collaboration in an Open Software Repository. In *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work (CSCW '12)*. Association for Computing Machinery, New York, NY, USA, 1277–1286. doi:10.1145/2145204.2145396
- [24] Rikke Friis Dam and Teo Yu Siang. 2022. Affinity Diagrams: How to Cluster Your Ideas and Reveal Insights. <https://www.interaction-design.org/literature/article/affinity-diagrams-learn-how-to-cluster-and-bundle-ideas-and-facts>
- [25] Uri Dekel. 2005. Supporting Distributed Software Design Meetings: What Can We Learn from Co-located Meetings? *ACM SIGSOFT Software Engineering Notes* 30, 4 (2005), 1–7. doi:10.1145/1082983.1083109
- [26] Sowmya Dhandapani. 2016. Integration of User Centered Design and Software Development Process. In *2016 IEEE 7th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. 1–5. doi:10.1109/IEMCON.2016.7746075
- [27] Paul Dourish and Victoria Bellotti. 1992. Awareness and Coordination in Shared Workspaces. In *Proceedings of the 1992 ACM Conference on Computer-Supported Cooperative Work (Toronto, Ontario, Canada) (CSCW '92)*. Association for Computing Machinery, New York, NY, USA, 107–114. doi:10.1145/143457.143468
- [28] Stack Exchange. 2023. Stack Overflow. <https://stackoverflow.com/>.
- [29] Stack Exchange. 2023. User Experience Stack Exchange. <https://ux.stackexchange.com/>.
- [30] K. J. Kevin Feng, Tony W Li, and Amy X. Zhang. 2023. Understanding Collaborative Practices and Tools of Professional UX Practitioners in Software Organizations. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 764, 20 pages. doi:10.1145/3544548.3581273
- [31] KJ Kevin Feng and Amyx Zhang. 2022. From Handoffs to Co-creation: Deepening Collaboration Between Designers, Developers, and Data Science Workers in UX Design. *Proceedings of the InContext: Futuring User-Experience Design Tools Workshop at CHI Conference on Human Factors in Computing Systems* (2022).
- [32] Bruna Ferreira, Silvio Marques, Marcos Kalinowski, Hélio Lopes, and Simone D.J. Barbosa. 2023. Lessons Learned to Improve the UX Practices in Agile Projects Involving Data Science and Process Automation. *Information and Software Technology* 155 (2023), 107106. doi:10.1016/j.infsof.2022.107106
- [33] Jennifer Ferreira, James Noble, and Robert Biddle. 2007. Agile Development Iterations and UI Design. In *Agile 2007 (AGILE 2007)*. 50–58. doi:10.1109/AGILE.2007.8
- [34] Jennifer Ferreira, James Noble, and Robert Biddle. 2007. Interaction Designers on eXtreme Programming Teams: Two Case Studies From the Real World. In *Proceedings of the Fifth New Zealand Computer Science Research Student Conference (NZCSRSC2007)*, Vol. 55. 93–94.
- [35] Jennifer Ferreira, Helen Sharp, and Hugh Robinson. 2011. User Experience Design and Agile Development: Managing Cooperation Through Articulation Work. *Software: Practice and Experience* 41, 9 (2011), 963–974. doi:10.1002/spe.1012
- [36] Jennifer Ferreira, Helen Sharp, and Hugh Robinson. 2012. Agile Development and User Experience Design Integration as an Ongoing Achievement in Practice. In *2012 Agile Conference*. 11–20. doi:10.1109/Agile.2012.33
- [37] Figma. 2023. Figma. <https://www.figma.com/>.
- [38] Figma. 2023. Github and Figma—Better Together. <https://www.figma.com/github-for-figjam/>.
- [39] David Fox, Jonathan Sillito, and Frank Maurer. 2008. Agile Methods and User-Centered Design: How These Two Methodologies are Being Successfully Integrated in Industry. In *Agile 2008 Conference*. 63–72. doi:10.1109/Agile.2008.78
- [40] Andrei Garcia, Tiago Da Silva, and Milene Silveira. 2017. Artifacts for Agile User-Centered Design: A Systematic Mapping. In *Hawaii International Conference on System Sciences*. 5859–5868.
- [41] R Green, THOMAS Mazzuchi, and SHAHRAM Sarkani. 2010. Understanding the Role of Synchronous & Asynchronous Communication in Agile Software Development and Its Effects on Quality. *Journal of Information Technology Management* 21, 2 (2010), 8–23.
- [42] Carl Gutwin and Saul Greenberg. 2004. The Importance of Awareness for Team Cognition in Distributed Collaboration. In *Team cognition: Understanding the factors that drive process and performance*. American Psychological Association, 177–201. doi:10.1037/10690-009
- [43] Carl Gutwin, Reagan Penner, and Kevin Schneider. 2004. Group Awareness in Distributed Software Development. In *Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work (CSCW '04)*. Association for Computing Machinery, New York, NY, USA, 72–81. doi:10.1145/1031607.1031621
- [44] Bengt Göransson, Jan Gulliksen, and Inger Boivie. 2004. The usability design process - Integrating user-centered systems design in the software development process. *Software Process: Improvement and Practice* 8 (04 2004), 111–131. doi:10.1002/spip.174
- [45] Rex Hartson and Pardha S Pyla. 2019. *The UX book* (2 ed.). Morgan Kaufmann, Oxford, England.
- [46] Annemarie Harzl. 2017. Can Foss Projects Benefit From Integrating Kanban: A Case Study. *Journal of Internet Services and Applications* 8, 1 (2017), 7. doi:10.1186/s13174-017-0058-z

- [47] Marc Hassenzahl and Noam Tractinsky. 2006. User Experience - A Research Agenda. *Behaviour & Information Technology* 25, 2 (2006), 91–97. doi:10.1080/01449290500330331
- [48] Lile Hattori. 2010. Enhancing Collaboration of Multi-Developer Projects With Synchronous Changes. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 2 (ICSE '10)*. Association for Computing Machinery, New York, NY, USA, 377–380. doi:10.1145/1810295.1810397
- [49] Jazlyn Hellman, Jinghui Cheng, and Jin L.C. Guo. 2021. Facilitating Asynchronous Participatory Design of Open Source Software: Bringing End Users into the Loop. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems (CHI EA '21)*. Association for Computing Machinery, New York, NY, USA, Article 438, 7 pages. doi:10.1145/3411763.3451643
- [50] James D. Herbsleb. 2007. Global Software Engineering: The Future of Socio-Technical Coordination. In *Future of Software Engineering (FOSE '07)*. 188–198. doi:10.1109/FOSE.2007.11
- [51] Thomas Hewett, Ronald Baecker, Stuart Card, Tom Carey, Jean Gasen, Marilyn Mantei, Gary Perlman, Gary Strong, and William Verplank. 1992. *ACM SIGCHI Curricula for Human-Computer Interaction*. Association for Computing Machinery. doi:10.1145/2594128
- [52] Andreas Hinderks, Francisco José Domínguez Mayo, Jörg Thomaschewski, and María José Escalona. 2022. Approaches to Manage the User Experience Process in Agile Software Development: A Systematic Literature Review. *Information and Software Technology* 150 (2022), 106957. doi:10.1016/j.infsof.2022.106957
- [53] P. Hodgetts. 2005. Experiences Integrating Sophisticated User Experience Design Practices Into Agile Processes. In *Agile Development Conference (ADC '05)*. 235–242. doi:10.1109/ADC.2005.24
- [54] Netta Iivari. 2005. Usability Specialists – ‘A Mommy Mob’, ‘Realistic Humanists’ or ‘Staid Researchers’? An Analysis of Usability Work in the Software Product Development. In *Human-Computer Interaction - INTERACT 2005*. Springer-Verlag, Berlin, Heidelberg, 418–430. doi:10.1007/11555261_35
- [55] Minna Isomursu, Andrey Sirotkin, Petri Voltti, and Markku Halonen. 2012. User Experience Design Goes Agile in Lean Transformation – A Case Study. In *2012 Agile Conference*. 1–10. doi:10.1109/Agile.2012.10
- [56] Rodi Jolak, Andreas Wortmann, Grischa Liebel, Eric Umuhoza, and Michel R. V. Chaudron. 2020. The Design Thinking of Co-located vs. Distributed Software Developers: Distance Strikes Again!. In *Proceedings of the 15th International Conference on Global Software Engineering (ICGSE '20)*. Association for Computing Machinery, New York, NY, USA, 106–116. doi:10.1145/3372787.3390438
- [57] Anirudha Joshi, N.L. Sarda, and Sanjay Tripathi. 2010. Supporting Multidisciplinary Collaboration: Requirements From Novel HCI Education. *Journal of Systems and Software* 83, 11 (2010), 2045–2058. doi:10.1016/j.jss.2010.03.078
- [58] Gabriela Jurca, Theodore D. Hellmann, and Frank Maurer. 2014. Integrating Agile and User-Centered Design: A Systematic Mapping and Review of Evaluation and Validation Studies of Agile-UX. In *2014 Agile Conference*. 24–32. doi:10.1109/AGILE.2014.17
- [59] Natalia Juristo, Ana Moreno, Maria-Isabel Sanchez-Segura, and Maria Cecilia Calani Baranauskas. 2007. A Glass Box Design: Making the Impact of Usability on Software Development Visible. In *Human-Computer Interaction – INTERACT 2007*. Springer Berlin Heidelberg, 541–554. doi:10.1007/978-3-540-74800-7_49
- [60] Pariya Kashfi, Robert Feldt, and Agneta Nilsson. 2019. Integrating UX Principles and Practices Into Software Development Organizations: A Case Study of Influencing Events. *Journal of Systems and Software* 154 (2019), 37–58. doi:10.1016/j.jss.2019.03.066
- [61] Pariya Kashfi, Agneta Nilsson, and Robert Feldt. 2017. Integrating User eXperience Practices Into Software Development Processes: Implications of the UX Characteristics. *PeerJ Computer Science* 3 (2017), e130. doi:10.7717/peerj-cs.130
- [62] Barbara Kitchenham and Pearl Brereton. 2013. A Systematic Review of Systematic Review Process Research in Software Engineering. *Information and Software Technology* 55, 12 (2013), 2049–2075. doi:10.1016/j.infsof.2013.07.010
- [63] Barbara Kitchenham and Stuart Charters. 2007. Guidelines for Performing Systematic Literature Reviews in Software Engineering. *EBSE Technical Report* (2007).
- [64] Harald Klein, Eric Knauss, and Andreas Rausch. 2012. Scaling Software Development Methods from Co-located to Distributed. In *Software Quality. Process Automation in Software Development*, Stefan Biffl, Dietmar Winkler, and Johannes Bergsmann (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 71–83.
- [65] Johanna Kollmann, Helen Sharp, and Ann Blandford. 2009. The Importance of Identity and Vision to User Experience Designers on Agile Projects. In *2009 Agile Conference*. 11–18. doi:10.1109/AGILE.2009.58
- [66] Kati Kuusinen. 2015. Task Allocation Between UX Specialists and Developers in Agile Software Development Projects. In *Human-Computer Interaction – INTERACT 2015*. Springer-Verlag, Berlin, Heidelberg, 27–44. doi:10.1007/978-3-319-22698-9_3
- [67] Marta Larusdottir, Jan Gulliksen, and Åsa Cajander. 2017. A License to Kill – Improving UCSD in Agile Development. *Journal of Systems and Software* 123 (2017), 214–222. doi:10.1016/j.jss.2016.01.024
- [68] Germán Leiva, Nolwenn Maudet, Wendy Mackay, and Michel Beaudouin-Lafon. 2019. Enact: Reducing Designer–Developer Breakdowns When Prototyping Custom Interactions. *ACM Transactions on Computer-Human*

- Interaction* 26, 3, Article 19 (2019), 48 pages. doi:10.1145/3310276
- [69] Q. Vera Liao, Hariharan Subramonyam, Jennifer Wang, and Jennifer Wortman Vaughan. 2023. Designerly Understanding: Information Needs for Model Transparency to Support Design Ideation for AI-Powered User Experience. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 9, 21 pages. doi:10.1145/3544548.3580652
 - [70] Daniel A. Magües, John W. Castro, and Silvia T. Acuña. 2016. Usability in Agile Development: A Systematic Mapping Study. In *2016 XLII Latin American Computing Conference (CLEI)*. 1–8. doi:10.1109/CLEI.2016.7833347
 - [71] Alina Mailach and Norbert Siegmund. 2023. Socio-Technical Anti-patterns in Building ML-Enabled Software: Insights From Leaders on the Forefront. In *Proceedings of the 45th International Conference on Software Engineering (ICSE '23)*. IEEE Press, 690–702. doi:10.1109/ICSE48619.2023.00067
 - [72] Nolwenn Maudet, Germán Leiva, Michel Beaudouin-Lafon, and Wendy Mackay. 2017. Design Breakdowns: Designer-Developer Gaps in Representing and Interpreting Interactive Systems. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing (CSCW '17)*. Association for Computing Machinery, New York, NY, USA, 630–641. doi:10.1145/2998181.2998190
 - [73] Thomas Memmel, Fredrik Gundelsweiler, and Harald Reiterer. 2007. Agile Human-Centered Software Engineering. In *Proceedings of the 21st British HCI Group Annual Conference on People and Computers - Volume 1 (BCS-HCI '07)*. BCS Learning & Development Ltd., Swindon, GBR, 167–175.
 - [74] Gerard Meszaros and Janice Aston. 2006. Adding Usability Testing to an Agile Project. In *Conference on AGILE 2006 (AGILE '06)*. IEEE Computer Society, USA, 289–294. doi:10.1109/AGILE.2006.5
 - [75] Miro. 2023. Miro. <https://miro.com/>.
 - [76] D. Moher, A. Liberati, J. Tetzlaff, and D. G. Altman. 2009. Preferred Reporting Items for Systematic Reviews and Meta-Analyses: The Prisma Statement. *BMJ* 339, jul21 1 (2009), b2535. doi:10.1136/bmj.b2535
 - [77] Matthias Müller. 2018. Agile Challenges and Chances for Open Source: Lessons Learned From Managing a Floss Project. In *2018 IEEE Conference on Open Systems (ICOS)*. IEEE, 1–6. doi:10.1109/ICOS.2018.8632819
 - [78] Nadia Nahar, Shurui Zhou, Grace Lewis, and Christian Kästner. 2022. Collaboration Challenges in Building ML-Enabled Systems: Communication, Documentation, Engineering, and Process. In *Proceedings of the 44th International Conference on Software Engineering (ICSE '22)*. Association for Computing Machinery, New York, NY, USA, 413–425. doi:10.1145/3510003.3510209
 - [79] Maryam Najafi and Len Toyoshiba. 2008. Two Case Studies of User Experience Design and Agile Development. In *Proceedings of the Agile 2008 Conference (AGILE '08)*. IEEE Computer Society, USA, 531–536. doi:10.1109/Agile.2008.67
 - [80] John Noll, Sarah Beecham, and Ita Richardson. 2011. Global Software Development and Collaboration: Barriers and Solutions. *ACM Inroads* 1, 3 (2011), 66–78. doi:10.1145/1835428.1835445
 - [81] Mie Nørgaard and Kasper Hornbæk. 2010. Working Together to Improve Usability: Exploring Challenges and Successful Practices. *International Journal of Technology and Human Interaction* 6, 1 (2010), 33–53. doi:10.4018/jthi.2010091703
 - [82] Stack Overflow. 2023. How to Upload Figma File(Wireframe) on Github? <https://stackoverflow.com/questions/70094767/how-to-upload-figma-filewireframe-on-github>.
 - [83] Freddy Paz and José Antonio Pow-Sang. 2015. Usability Evaluation Methods for Software Development: A Systematic Mapping Review. In *2015 8th International Conference on Advanced Software Engineering & Its Applications (ASEA)*. 1–4. doi:10.1109/ASEA.2015.8
 - [84] Anathan Pereira, Abner Cleto Filho, Eduardo Guerra, and Luciana Zaina. 2022. Towards a Pattern Language to Embed UX Information in Agile Software Requirements. In *Proceedings of the 26th European Conference on Pattern Languages of Programs (EuroPLOP '21)*. Association for Computing Machinery, New York, NY, USA, Article 18, 8 pages. doi:10.1145/3489449.3489991
 - [85] Julio Cesar Pereira and Rosaria de F.S.M. Russo. 2018. Design Thinking Integrated in Agile Software Development: A Systematic Literature Review. *Procedia Computer Science* 138 (2018), 775–782. doi:10.1016/j.procs.2018.10.101
 - [86] Narendren Pillay and Jeanette Wing. 2019. Agile UX: Integrating good UX development practices in Agile. In *2019 Conference on Information Communications Technology and Society (ICTAS)*. 1–6. doi:10.1109/ICTAS.2019.8703607
 - [87] ProtoPie. 2023. ProtoPie. <https://www.protopie.io/>.
 - [88] Francisco Queiroz, Raniere Silva, Jonah Miller, Sandor Brockhauser, and Hans Fangohr. 2017. Good Usability Practices in Scientific Software Development. In *Proceedings of WSSPE5.1*.
 - [89] Mikko Rajanen and Netta Iivari. 2015. Power, Empowerment and Open Source Usability. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. Association for Computing Machinery, New York, NY, USA, 3413–3422. doi:10.1145/2702123.2702441
 - [90] Mikko Rajanen, Netta Iivari, and Arto Lanamäki. 2015. Non-response, Social Exclusion, and False Acceptance: Gatekeeping Tactics and Usability Work in Free-Libre Open Source Software Development. In *Human-Computer Interaction – INTERACT 2015*. Springer-Verlag, Berlin, Heidelberg, 9–26. doi:10.1007/978-3-319-22698-9_2

- [91] Reddit. 2023. Designers - Do You Go in Github? https://www.reddit.com/r/DesignSystems/comments/1bjj95r/how_do_you_manage_release_versions_with_figma/.
- [92] Reddit. 2023. Is There a Way to Streamline What We Create in Figma and What the Developers Have Created on Their End? https://www.reddit.com/r/FigmaDesign/comments/1278o7z/is_there_a_way_to_streamline_what_we_create_in/.
- [93] Reddit. 2023. Reddit Software Engineering. <https://www.reddit.com/r/SoftwareEngineering/>.
- [94] Reddit. 2023. Reddit UXDesign: For Experienced and Veteran UX Practitioners. <https://www.reddit.com/r/UXDesign/>.
- [95] Dina Salah, Richard F. Paige, and Paul Cairns. 2014. A Systematic Literature Review for Agile Development Processes and User Centred Design Integration. In *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering (EASE '14)*. Association for Computing Machinery, New York, NY, USA, Article 5, 10 pages. doi:10.1145/2601248.2601276
- [96] Arghavan Sanei and Jinghui Cheng. 2024. Characterizing Usability Issue Discussions in Open Source Software Projects. *Proceedings of the ACM on Human-Computer Interaction* 8, CSCW1, Article 30 (2024), 26 pages. doi:10.1145/3637307
- [97] Raghvinder S. Sangwan, Kathryn W. Jablokow, and Joanna F. DeFranco. 2020. Asynchronous Collaboration: Bridging the Cognitive Distance in Global Software Development Projects. *IEEE Transactions on Professional Communication* 63, 4 (2020), 361–371. doi:10.1109/TPC.2020.3029674
- [98] Anita Sarma, David F. Redmiles, and André van der Hoek. 2012. Palantir: Early Detection of Development Conflicts Arising from Parallel Code Changes. *IEEE Transactions on Software Engineering* 38, 4 (2012), 889–908. doi:10.1109/TSE.2011.64
- [99] Panagiotis Sfetos, Lefteris Angelis, Ioannis Stamelos, and Pashalis Raptis. 2016. Integrating User-Centered Design Practices Into Agile Web Development: A Case Study. In *2016 7th International Conference on Information, Intelligence, Systems & Applications (IISA)*. 1–6. doi:10.1109/IISA.2016.7785424
- [100] Mary Shaw. 2005. Software Engineering for the 21st Century: A Basis for Rethinking the Curriculum. In *School of Computer Science Carnegie Mellon University*.
- [101] Tiago Silva da Silva, Angela Martin, Frank Maurer, and Milene Silveira. 2011. User-Centered Design and Agile Methods: A Systematic Review. In *2011 Agile Conference*. 77–86. doi:10.1109/AGILE.2011.24
- [102] Sketchboard. 2014. Online Collaborative Whiteboard. <https://sketchboard.io/blog/2014/03/06/github-sketchboard>.
- [103] Osama Sohaib and Khalid Khan. 2010. Integrating Usability Engineering and Agile Software Development: A Literature Review. In *2010 International Conference On Computer Design and Applications*, Vol. 2. V2–32–V2–38. doi:10.1109/ICCD.2010.5540916
- [104] Daniel Ståhl. 2023. The Dynamic Versus the Stable Team: The Unspoken Question in Large-Scale Agile Development. *Journal of Software: Evolution and Process* 35, 12 (2023), e2589. doi:10.1002/smr.2589
- [105] Diane E. Strode, Sid L. Huff, Beverley Hope, and Sebastian Link. 2012. Coordination in Co-located Agile Software Development Projects. *Journal of Systems and Software* 85, 6 (2012), 1222–1238. doi:10.1016/j.jss.2012.02.017
- [106] Desirée Sy. 2007. Adapting Usability Investigations for Agile User-Centered Design. *Journal of Usability Studies* 2, 3 (2007), 112–132.
- [107] Michael Terry, Matthew Kay, and Ben Lafreniere. 2010. Perceptions and Practices of Usability in the Free/Open Source Software (FoSS) Community. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. Association for Computing Machinery, New York, NY, USA, 999–1008. doi:10.1145/1753326.1753476
- [108] Christoph Treude and Margaret-Anne Storey. 2010. Awareness 2.0: Staying Aware of Projects, Developers and Tasks Using Dashboards and Feeds. In *Proceedings of the ACM/IEEE 32nd International Conference on Software Engineering*, Vol. 1. 365–374. doi:10.1145/1806799.1806854
- [109] Jim Ungar. 2008. The Design Studio: Interface Design for Agile Teams. In *Agile 2008 Conference*. 519–524. doi:10.1109/Agile.2008.51
- [110] Adeola Wale-Kolade and Peter Axel Nielsen. 2015. Apathy Towards the Integration of Usability Work: A Case of System Justification. *Interacting with Computers* 28, 4 (2015), 437–450. doi:10.1093/iwc/iwv016
- [111] Adeola Yetunde Wale-Kolade. 2015. Integrating Usability Work Into a Large Inter-Organisational Agile Development Project: Tactics Developed by Usability Designers. *Journal of Systems and Software* 100 (2015), 54–66. doi:10.1016/j.jss.2014.10.036
- [112] Claes Wohlin. 2014. Guidelines for Snowballing in Systematic Literature Studies and a Replication in Software Engineering. In *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering (EASE '14)*. Association for Computing Machinery, New York, NY, USA, Article 38, 10 pages. doi:10.1145/2601248.2601268
- [113] Qian Yang, Alex Scuito, John Zimmerman, Jodi Forlizzi, and Aaron Steinfeld. 2018. Investigating How Experienced UX Designers Effectively Work with Machine Learning. In *Proceedings of the 2018 Designing Interactive Systems Conference (DIS '18)*. Association for Computing Machinery, New York, NY, USA, 585–596. doi:10.1145/3196709.3196730

- [114] Sabah Zdanowska and Alex S Taylor. 2022. A study of UX practitioners roles in designing real-world, enterprise ML systems. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*. 1–15.
- [115] Görkem Çetin and Mehmet Göktürk. 2008. A Measurement Based Framework for Assessment of Usability-Centricness of Open Source Software Projects. In *2008 IEEE International Conference on Signal Image Technology and Internet Based Systems*. 585–592. doi:[10.1109/SITIS.2008.106](https://doi.org/10.1109/SITIS.2008.106)

A Appendix

A.1 Post links

This section presents a list of all the referenced post links in the main text.

- [P1] <https://ux.stackexchange.com/questions/144725/>
- [P2] <https://ux.stackexchange.com/questions/125859/>
- [P3] <https://ux.stackexchange.com/questions/138215/>
- [P4] <https://www.reddit.com/r/UXDesign/comments/lrpngo/>
- [P5] <https://ux.stackexchange.com/questions/140837/>
- [P6] <https://www.reddit.com/r/UXDesign/comments/uhn981/>
- [P7] <https://www.reddit.com/r/UXDesign/comments/v0sh4c/>
- [P8] <https://www.reddit.com/r/UXDesign/comments/11xupv0/>
- [P9] <https://www.reddit.com/r/SoftwareEngineering/comments/sytmb2/>
- [P10] <https://ux.stackexchange.com/questions/132945/>
- [P11] <https://www.reddit.com/r/UXDesign/comments/pp15aj/>
- [P12] <https://ux.stackexchange.com/questions/135804/>
- [P13] <https://www.reddit.com/r/UXDesign/comments/z2nis5/>
- [P14] <https://ux.stackexchange.com/questions/131656/>
- [P15] <https://ux.stackexchange.com/questions/133644/>
- [P16] <https://www.reddit.com/r/UXDesign/comments/129rds/>
- [P17] <https://www.reddit.com/r/SoftwareEngineering/comments/107jqt/>
- [P18] <https://ux.stackexchange.com/questions/136765/>
- [P19] <https://ux.stackexchange.com/questions/131927/>
- [P20] <https://www.reddit.com/r/UXDesign/comments/gcm2o0/>
- [P21] <https://www.reddit.com/r/UXDesign/comments/e25iuu/>
- [P22] <https://www.reddit.com/r/UXDesign/comments/z3foqp/>
- [P23] <https://www.reddit.com/r/UXDesign/comments/rlorbm/>
- [P24] <https://www.reddit.com/r/UXDesign/comments/pklqo7/>
- [P25] <https://www.reddit.com/r/UXDesign/comments/10kz2zw/>

A.2 Issue Links

This section presents a list of all the referenced issue links in the main text.

- [I1] <https://github.com/microsoft/vscode/issues/156179>
- [I2] <https://github.com/microsoft/vscode/issues/98614>
- [I3] <https://github.com/microsoft/vscode/issues/130611>
- [I4] <https://github.com/microsoft/vscode/issues/63152>
- [I5] <https://github.com/microsoft/vscode/issues/147903>
- [I6] <https://github.com/microsoft/vscode/issues/131641>
- [I7] <https://github.com/microsoft/vscode/issues/133622>
- [I8] <https://github.com/microsoft/vscode/issues/144324>
- [I9] <https://github.com/microsoft/vscode/issues/146806>
- [I10] <https://github.com/microsoft/vscode/issues/100347>
- [I11] <https://github.com/microsoft/vscode/issues/101375>
- [I12] <https://github.com/microsoft/vscode/issues/89533>
- [I13] <https://github.com/microsoft/vscode/issues/144198>
- [I14] <https://github.com/microsoft/vscode/issues/115808>
- [I15] <https://github.com/microsoft/vscode/issues/144823>
- [I16] <https://github.com/microsoft/vscode/issues/130516>

[I17] <https://github.com/microsoft/vscode/issues/119776>

A.3 List of Keywords

In the data processing process, we utilized specific keywords in both the paper titles and publication titles to exclude studies in other fields that are not related to UX and SE, or those presented at minor or regional conferences.

Small and local conferences: symposium, workshop, seminar, congress, phd school association, country, jubilee, Asia-Pacific, latin, north american, asia, australasian, arabic, euromicro, brazilian, malaysian

Other fields: avionics, cyber security, security, distributed computing, image processing, blockchain, wearable, api, java, reality, augmented, virtual, 3d, hardware, image, visual, algorithm, query, machine, cloud, network, companion, extended abstracts, smart contract, crypto, dataset, computing, data, graph, machine learning, operating system, computational, securing, health, secure, power, deep learning, memory, prediction, optimization, optimize, xml, sql, clone, generator, generation, calculation, calculate, auto, toolkit

In this step, we filter out 824 papers. All filtered papers are reviewed by the first author to ensure there is no false negative.

Received 16 January 2024; revised 11 July 2024; accepted 09 December 2024