Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 4, Issue 3, 2026

SYNCHRONOUS COMMUNICATION DEFICIT IN GLOBAL SOFTWARE DEVELOPMENT: A SYSTEMATIC LITERATURE REVIEW OF CHALLENGES, MITIGATION STRATEGIES, AND A ROBUST CONCEPTUAL FRAMEWORK

Atta Ur Rahman[*1], **Fahim Muhammad Khan**[2]

[*1]*School of Computer & Software, Nanjing University of Information Science &Technology, Nanjing, 210044, China*
[2]*Department of Software Engineering, Institute of Management Sciences (IMSciences), Peshawar, 25000, KPK*

[*1]202451200014@nuist.edu.cn, [2]fahim.muhammad@imsciences.edu.pk

DOI: https://doi.org/10.5281/zenodo.19382727

Keywords
Synchronous communication deficit, Global Software Development, distributed teams, communication challenges, mitigation strategies, coordination, systematic literature review, conceptual framework

Article History
Received: 31 January 2026
Accepted: 15 March 2026
Published: 31 March 2026

Corresponding Author: *
Atta Ur Rahman

Abstract

Global Software Development (GSD) has become popular in the modern software development industry because of the advantages it offers, such as cost savings, global access to expertise, and faster time-to-market. However, the distributed nature of GSD creates several challenges in communication, coordination, and collaboration among geographically dispersed teams. Among them, the absence of synchronous communication has emerged as a major issue with a significant impact on real-time interaction, decision-making processes, and the team's performance as a whole. In GSD environments, teams often use asynchronous communication due to different time zones and geographical dispersion, which can lead to delayed responses, misunderstandings, lower coordination efficiency, and reduced team cohesion. Ineffective synchronous communication mechanisms can contribute to these challenges, resulting in stalled projects and low productivity. This study aims to systematically identify the major issues associated with the lack of synchronous communication in Global Software Development and to analyse the mitigation strategies proposed in the extant literature. A Systematic Literature Review (SLR) following a structured protocol was conducted. Relevant studies were collected from major digital databases, including IEEE Xplore, Science Direct, Google Scholar, Semantic Scholar, Wiley Online Library, and Research Gate. After the stringent application of the inclusion, exclusion, and quality assessment criteria, a selection of primary studies was selected for detailed analysis. This study identifies major problems arising from the lack of synchronous communication in GSD environments, including delayed responses, limited overlapping working hours, communication breakdowns, and a lack of informal communication among team members. Additionally, some mitigation strategies were identified, such as scheduling working hours at the same time, using a real-time communication tool, adopting a structured communication protocol, and implementing collaborative practices to foster synchronous interaction. A total of 8 synchronous communication issues and 16 best practices in GSD were identified from the literature, Based on these findings, this study proposes a conceptual framework that shows the relationships between issues in synchronous communication issues and their corresponding mitigation strategies in GSD

environments. The proposed framework provides a structured understanding that can help researchers and practitioners build more effective communication, coordination, and collaboration among distributed software development teams.

1. INTRODUCTION

Global Software Development (GSD) has emerged as a leading approach in modern software engineering, driven by the advantages of accessing global talent, lowering operational expenses, and accelerating software delivery. As technological advancements continue to accelerate, more organizations are adopting GSD to remain competitive. Yet, the geographically dispersed structure of such projects introduces significant challenges related to communication, coordination, and teamwork across distant locations [1], [2].

Effective communication is critical to the success of any distributed software initiative, serving as the foundation for task management, knowledge exchange, and decision-making across sites. In contrast to collocated teams, where face-to-face interaction enables immediate feedback and nuanced dialogue, GSD depends largely on digital communication tools. Numerous studies have identified communication inefficiencies as a primary obstacle in distributed development settings [3], [4].

A key issue is the limited opportunity for synchronous communication—real-time interaction—due to differences in time zones, physical distance, and organizational structures. This often forces teams to depend on asynchronous methods like email and online forums, which can delay responses and hinder collaborative efficiency [5], [6]. The absence of immediate dialogue complicates coordination efforts and may lead to misunderstandings, slower decisions, and weakened team unity [7], [8].

While real-time communication tools—such as virtual meetings, video calls, and instant messaging—are vital for smooth coordination in GSD, their use is frequently constrained by minimal overlap in work schedules, inconsistent team availability, and insufficient communication systems [9], [10]. Additionally, heavy reliance on asynchronous channels limits spontaneous, informal interactions, which play a crucial role in building trust and shared understanding among distributed team members [11], [12].

Existing research has examined several communication- related issues in GSD, particularly in agile and collaborative work, including temporal distance, cultural differences, and technological limitations [13], [14]. Empirical studies further show that insufficient synchronous communication negatively impacts coordination, knowledge sharing, and the overall performance of the project [15], [16].

Despite the extensive body of research available, the issue of insufficient synchronous communication has not been thoroughly examined as a distinct topic. It is often grouped within broader communication challenges, resulting in limited insight into its unique causes, consequences, and targeted solutions.

Existing studies associate the lack of real-time interaction in Global Software Development (GSD) with several key problems: minimal overlap in team working hours, slow feedback cycles, reduced informal exchanges, communication interruptions, and increased misunderstandings. To address these issues, various approaches have been suggested, such as improved scheduling and time coordination, adoption of real- time communication tools, implementation of structured communication procedures, and stronger team collaboration practices.

To fill this research gap, this study undertakes a Systematic Literature Review (SLR) to identify and analyze the specific challenges tied to asynchronous communication in GSD, along with the mitigation strategies discussed in the literature. Drawing from the findings, a relational framework— presented as a conceptual model—is developed to map the connections between these challenges and their corresponding solutions. The research contributes by providing a clear classification of communication issues,

integrating existing mitigation strategies into a cohesive structure, and proposing a conceptual model to support future research and improve communication practices in global software development settings.

The remainder of this paper is organized as follows: Section 2 discusses the literature review, Section 3 presents the methodology used to conduct the systematic literature review, Section 4 Categorization of Synchronous Communication Issues and Mitigation Strategies in GSD, section 5 Overview Issues and Sub-Issues of Lack of Synchronous Communication in GSD, section 6 section 5 introduces the proposed conceptual framework and hypothesis development, Section 6 Overview of Mitigation Strategies for Lack of Synchronous Communication in GSD, while section 7 presents the hypotheses of the proposed framework, section 8 Hypothesis of the proposed conceptual framework and Section 9 concludes the paper and outlines future research directions.

2. Literature Review

Global Software Development (GSD) has attracted significant attention in both academic and industrial circles because of opportunities to leverage globally distributed expertise, lower development costs, and faster software product development. Despite these advantages, GSD introduces challenges that can make communication, coordination, and collaboration among distributed teams difficult [1], [3].

Among these challenges, communication has always been identified as one of the most important factors affecting the success or otherwise of distributed software development projects [4], [5]. In GSD environments, software development activities are performed by teams located in different geographical areas, often with time, cultural, and organizational differences.

Communication becomes harder when these elements block smooth interaction, limiting chances for instant exchanges across team members. Earlier research points out how clear dialogue supports better coordination, exchange of know-how, and overall success in remote teams [7]. What stands out is that live interactions - where people communicate at once - help solve problems faster, support clearer choices, even strengthen teamwork [8], [9].

Still, across numerous global software development contexts, face-to-face interaction becomes rare due to vast geographical separation, along with workdays overlapping just briefly - sometimes not at all. Evidence suggests groups operating across several time zones struggle to find mutual availability for live discussion; hence, their reliance grows on delayed methods like written messages or task logs [10]. Although staggered messaging allows freedom in timing, it typically brings slower replies, thinner exchange of ideas, and less effective teamwork alignment [12], [13].

Findings from Herbsleb and Mockus indicate fewer real-time talks result in more miscoordination, which drags down performance within dispersed setups [14]. Time gaps between team members have long influenced how live communication works in global software development. When people work across different hours, chances drop for immediate discussions - delays grow, research confirms [15]. In one study, wider spans in local times meant fewer talks and slower choices [16]. Other findings back this: without shared working periods, teamwork weakens, coordination lags [17]. Without overlapping schedules, constant contact becomes hard - a key hurdle since steady exchange supports strong joint efforts.

A key reason behind frequent communication issues within global software development settings lies in missing real-time dialogue - this absence blocks smooth information flow. Feedback often arrives late, dragging down momentum in development cycles while raising the chances of setbacks. Research shows that when coordination slows across distant team members, critical decisions stall along with essential work [18], [19].

Without instant back-and-forth exchanges, small confusion points grow into larger breakdowns more easily. Misinterpreted needs and uneven outcomes in building systems tend to emerge where quick confirmation isn't possible, as noted by Damian and Zowghi [20].

Starting off differently, missing real-time talks tends to reduce unplanned conversations across teammates. These offhand exchanges - like quick remarks or sudden brainstorming moments - help grow mutual confidence, strengthen group unity, while also boosting cooperative efforts. Studies indicate remote groups with scarce spontaneous dialogue often struggle forming reliable bonds or smooth work dynamics [21], [22]. A different angle reveals limited live interaction results in lower visibility into shared tasks, leaving individuals playing catch-up on current developments and workflow status [23].

When teams work across distances, staying in touch relies heavily on digital tools made for live exchanges. Video meetings, quick message apps, together with shared online spaces, help people interact without delay. Still, research shows success often comes down to how well these tools fit into daily workflows - choice matters, so does setup and connection to existing systems [24], [25]. Poor handling of such technology may split conversations, reduce involvement, or weaken teamwork within far-flung groups [26]. Fresh insights suggest inventive thinking is needed when shaping unified platforms that blend various features for smooth, immediate coordination [27], [28].

Synchronous talk matters deeply when agile work happens across locations. Because agility leans on steady dialogue, fast replies, and tight teamwork, delays cause strain. When people cannot speak at once, research shows agile methods weaken in remote setups [29], [30]. Distance - both physical and time-based - creates friction; Paasivaara and Lassenius found it hampers how well teams align and deliver [31]. From another angle, Moe and colleagues stressed real- time interaction helps distant groups stay coordinated and linked [32].

When teams work across distant locations, staying in sync becomes difficult - a gap some studies aim to close. Overlapping shifts help, letting members meet live even when separated by clocks [33]. Shifting start times back and forth

makes contact easier for those far apart [34]. Video tools and quick chat apps appear to bridge delays, smoothing how people connect now and then [24], [27]. Meetings on a fixed schedule, along with set rules for how information flows and thorough written records, support better interaction among remote team members [35], [36]. While some rely on strict formats, others find value in casual chats - these unplanned exchanges often strengthen mutual confidence. Even so, scheduled in-person gatherings, though rare, contribute to stronger group unity [21], [37]. With distance adding complexity, learning sessions that highlight practical ways to communicate prove useful across global software projects [38].

Focusing on gaps in current knowledge, research into global software development often overlooks how delayed interactions affect autonomous teams. While much work touches on communication difficulties, few explore what drives misalignment, its effects, or ways to reduce it when timing is off. Another gap lies in the absence of clear models linking specific time-related coordination problems to practical solutions.

One reason behind the problem lies in how teams communicate at different times. A closer look becomes necessary when patterns of misalignment keep appearing across projects. Through careful review of existing studies, this work explores what blocks real-time interaction among developers. Instead of assuming solutions fit all cases, attention shifts toward understanding specific barriers first. Insights emerge not just from data but also from connecting findings across diverse settings. By piecing together scattered evidence, a clearer picture forms about why fixes often fall short.

The approach does not promise quick answers yet reveals where efforts might be better directed. What results is a model shaped by gaps noticed in prior work rather than assumptions. Patterns in software teams' struggles point toward structural roots more than individual errors. From these threads, an explanation grows - not rigid, but adaptable to varied contexts.

3. Research Methodology

This part outlines how the current research was carried out. To uncover issues tied to poor timing

in communication within global software projects, a structured review of existing studies was performed. Instead, it focused on gathering solutions suggested by prior work to address these coordination gaps.

A suggested model emerges from the SLR results, showing ways teams handle real-time communication problems in remote software projects. How challenges unfold connects directly to which solutions appear effective across different settings.

The structure reflects patterns seen where distance shapes interaction needs. Instead of assuming one fix fits all, responses shift depending on team context. What works often depends less on tools, more on coordination habits formed over time.

3.1 Research Design

This part outlines how the study was carried out. Starting with a structured examination of existing writings, it pinpoints
key challenges tied to missing real-time interaction in worldwide software projects. Following that approach ensures clarity on what problems arise when teams work across distances without instant contact. Gathering insights this way builds a foundation based on prior findings rather than assumptions.

After identifying these issues, the mitigation strategies proposed in previous studies are analyzed and synthesized. Finally, a conceptual framework is developed based on the identified issues and mitigation strategies.

Figure 1. Research Design.

3.2. Systematic Literature Review

A Systematic Literature Review (SLR) is a method that is structured and systematic in order to identify, evaluate, and synthesize the existing research that pertains to a certain topic. The advantage of the SLR method is that it enables researchers to gather information from previous research studies to get reliable evidence and combine research findings in an objective and transparent manner.

The main purpose of SLR is to select the relevant studies that are relevant in their respective field, and this can be achieved by using predefined inclusion and exclusion criteria and by determining the quality of the selected research articles.

In this research, the SLR approach has been applied to identify problems arising from the lack of synchronous communication in Global Software Development and to identify mitigation approaches proposed to address such cases. The methodology pursued in this research has been based on the guidelines proposed by Kitchenham for conducting systematic literature reviews in software engineering.

The SLR process consists of three main phases:

- Planning the review
- Conducting the review
- Reporting the review

The phases of the SLR and their corresponding steps are illustrated in Figure 2.

Figure 2. SLR Protocol.

Furthermore, these phases are further divided into subsections.

- Planning the review phase is used to define the plan that we have developed to conduct the SLR.
- Conducting the review phase is an important phase of SLR. This phase implements

the search strategy that is defined in the first phase of SLR in order to obtain data from the literature.

- The third and the last phase of SLR are reporting the review. It is used to report the results, obtain from the earlier two phases. All the steps are discoursed in detail below.

3.2.1 **Planning The Review:** This phase involves on the following stages

- Research Objective
- Research Question
- Electronic Databases
- Inclusion Criteria
- Exclusion Criteria
- Quality Criteria for Study Selection.

All the steps are discussed in detail below.

3.2.1.1 Research Objective

The primary objective of this study is to systematically identify, analyze, and synthesize existing literature on synchronous communication deficits in Global Software Development (GSD). To achieve this overarching aim, the study addresses the following specific research objectives:

RO1: To identify and categorize the major challenges and issues arising from the lack of synchronous communication in GSD environments (addressing RQ1).

RO2: To identify, evaluate, and classify the mitigation strategies proposed in the literature to overcome synchronous communication deficiencies in GSD (addressing RQ2).

By achieving these objectives, the study seeks to provide a holistic understanding of the problem space and to develop a robust conceptual framework that consolidates the relationship between synchronous communication deficits, their associated challenges, and the corresponding mitigation strategies in global software development

3.2.1.2 Research Questions

The The objective of this study is to identify the issues related to lack of synchronous communication in Global Software Development and to investigate the mitigation strategies proposed to address these challenges in the literature.

For achieving the objective of the study, the following research questions were addressed and formulated.

RQ1: What are the major issues caused by lack of synchronous communication in Global Software Development?

RQ2: What mitigation strategies have been proposed to address synchronous communication issues in Global Software Development?

3.2.1.3 Electronic Database

To collect relevant studies, several widely recognized digital libraries and academic databases were used. These databases are commonly used in software engineering research and provide access to peer-reviewed journal and conference publications. Based on the research knowledge and recommendations provided in [34], a total of 6 suitable data repositories have been used.

- IEEE Explorer
- Science Direct
- Research Gate
- Semantic Scholar
- Wiley Online Library
- Google Scholar

3.2.1.4 Search String

A search string was developed to retrieve relevant studies related to technology selection issues in Global Software Development. The search string was constructed by combining keywords related to lack of synchronous communication , distributed software development, and mitigation strategies. The search string used in this study is shown below:

("synchronous communication" OR "real-time communication" OR "communication delay" OR "instant communication" OR "live communication") AND

("global software development" OR "distributed software development" OR "globally distributed teams" OR "distributed development") AND

("mitigation strateg*" OR "communication strateg*" OR "coordination strateg*" OR "solution*" OR "approach*")

3.2.1.5 Inclusion Criteria

The following criteria were used to include studies in the systematic review:

- "Studies must be published in Journal or Conference."
- "Studies must be written in the English language.
- "Studies having full text."
- "Studies that are answering questions of the research.
- "Studies published from 2009 to 2026"
- "Studies Studies that discuss technology selection "issues in Global Software Development"
- "Studies Studies that propose mitigation strategies or solutions related to lack of synchronous communication challenges"
- "Studies proposing mitigation strategies for lack of synchronous communication challenges"

3.2.1.6 Exclusion Criteria

The following criteria were used to exclude studies from the review:

- "Books and various blogs are excluded".
- "Slides", "tutorial", "editorials" etc. Are discarded."
- "Studies excluded which language not English."
- "Technical reports and white papers."
- "Studies were not available in full-text."
- "Duplicated or repeated studies."
- "Studies not related to Global Software Development"
- "Studies not related to lack of synchronous communication challenges"
- "Studies not related to proposing mitigation strategies for lack of synchronous communication challenges"

3.2.1.7 Quality Criteria for Study Selection

To ensure the quality of the selected studies, quality assessment criteria were applied. Each study was evaluated based on predefined quality questions.

Studies that fully addressed the quality assessment criteria were assigned a score of 1, partially addressed studies were assigned 0.5, and studies that did not address the criteria were assigned 0. Quality assessment criteria questions are defined in Table 1.

Table 1. Quality criteria questions

S No.	Quality Assessment Question	Score
QA1	Does the study discuss synchronous communication issues in GSD?	Yes =1, Partial = 0.5, No= 0
QA2	Does the study propose mitigation strategies for synchronous communication issues in GSD?	Yes = 1,Partial = 0.5, No= 0
QA3	Does the study propose or evaluate a framework or model related to synchronous communication issues in GSD?	Yes = 1,Partial = 0.5, No= 0
QA4	Is the study related to GSD?	Yes = 1, Partial = 0.5, No=0

3.2.1.8 Conducting The Review:

The conducting phase of the SLR consists of three main steps:

- Primary Study Selection
- Data extraction
- Data synthesis

All the steps are deliberated in detail below.

3.2.1.8.1 Primary Study Selection

The Tollgate approach was used for selecting primary studies. This approach consists of five levels of filtering to identify the most relevant studies.

The five levels include:

- **Level 1:** Identify studies using the search string and inclusion criteria
- **Level 2:** Screening based on title and abstract
- **Level 3:** Screening based on introduction and conclusion
- **Level 4:** Screening based on full text
- **Level 5:** Final selection based on quality assessment criteria

The Tollgate approach ensures that only high-quality and relevant studies are included in the systematic review. Table 2. Tollgate Approach for Primary Study Selection

Table 2 .Tollgate Approach for Primary Study Selection.

Electronic Databases	L1	L 2	L3	L4	L 5
Google Scholar	420	145	88	30	15
IEEE Explore	360	110	57	20	13
Semantic Scholar	290	92	34	15	5
Science Direct	240	68	12	8	6
Research Gate	310	75	21	9	4
Willey online library	28	70	23	5	3
Total	1900	560	235	87	50

Initially, 1900 studies were identified through database searches. After applying the tollgate selection process, 45 primary studies were selected for the systematic literature review. Figure 3. Tollgate Approach Distribution.

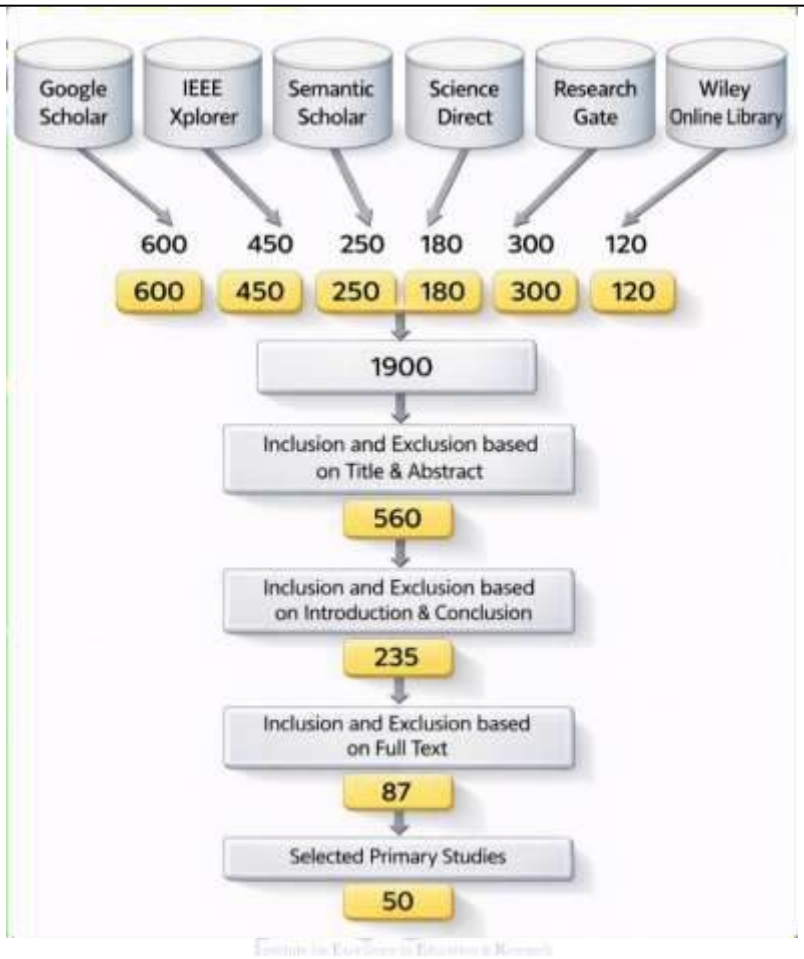

Figure 3 .Tollgate Approach Distribution.

3.2.1.8.2 **Data Extraction**

To answer the research questions, relevant information was extracted from the selected studies. The extracted data included:

- Title of the study
- Publication year
- Identified technology-related issues in GSD
- Proposed mitigation strategies

To ensure reliability, three reviewers independently analyzed selected articles and evaluated the extracted data. The Kendall (W) coefficient was used to measure the level of agreement between reviewers.

3.2.1.8.3 **Data Synthesis**

The extracted data from the chosen primary studies were analyzed and synthesized to identify lack of synchronous communication and mitigation strategies in Global Software Development.

Through this synthesis process, some of the key challenges related to lack of synchronous communication and possible mitigation strategies were identified. These findings were used to form a conceptual framework that shows the relationship between lack of synchronous communication issues and mitigation strategies in GSD environments.

3.2.1.9 Reporting The Review

After completing the review process, the results were reported in the final phase of the SLR. This phase includes

- Quality assessment of selected studies

- Classification of selected studies
- Temporal distribution of selected primary studies All the steps are discussed in detail below.

3.2.1.9.1 **Quality Assessment**

Quality assessment uality assessment was performed to ensure that only high-quality studies were included in the final analysis. Studies that scored less than 50% in the quality assessment criteria were excluded.

50 primary studies were considered in the systematic literature review. Quality assessment scores can be found in Table 3. Below is a SLR quality-assessment table with correct QA scoring logic based on: To examine the quality of the selected studies, a quality assessment process was conducted using the criteria defined in Table 1. Each study was evaluated based on its relevance to technology selection issues in Global Software Development and the mitigation strategies proposed.

Studies scoring less than 50% were excluded from the final analysis. After applying the quality assessment criteria, 50 primary studies were included in the final systematic literature review.

Table 3. Quality Assessments of the Studies.

Ref No	Year	QA1	QA2	QA3	QA4	Total Quality Score
[1]	2010	1	1	0	1	3
[2]	2011	0.5	1	1	1	3.5
[3]	2011	1	1	1	1	4
[4]	2012	1	1	0	1	3
[5]	2013	1	0.5	0	1	2.5
[6]	2013	1	1	0	1	3
[7]	2014	1	1	0	1	3
[8]	2014	1	0.5	0	1	2.5
[9]	2015	1	1	0	1	3
[10]	2015	0.5	1	0	1	2.5
[11]	2016	1	1	0	1	3
[12]	2016	0.5	1	1	1	3.5
[13]	2017	1	1	0	1	3
[14]	2017	0.5	1	0	1	2.5
[15]	2017	1	1	0	1	3
[16]	2018	1	1	0	1	3
[17]	2018	1	0.5	0	1	2.5
[18]	2018	0.5	1	1	1	3.5
[19]	2019	1	1	1	1	4
[20]	2019	1	0.5	0	1	2.5
[21]	2019	1	1	0	1	3
[22]	2020	1	1	0	1	3
[23]	2020	0.5	1	1	1	3.5
[24]	2020	1	0.5	0	1	2.5
[25]	2021	1	1	1	1	4
[26]	2021	1	1	0	1	3
[27]	2021	1	0.5	0	1	2.5
[28]	2021	1	1	0	1	3
[29]	2022	1	1	1	1	4
[30]	2022	1	0.5	0	1	2.5

[31]	2022	0.5	1	0	1	2.5
[32]	2022	1	1	1	1	4
[33]	2023	1	1	1	1	4
[34]	2023	0.5	0.5	1	1	3
[35]	2023	1	1	0	1	3
[36]	2023	1	1	1	1	4
[37]	2024	1	1	1	1	4
[38]	2024	1	0.5	1	1	3.5
[39]	2024	1	1	0	1	3
[40]	2024	1	1	1	1	4
[41]	2025	1	1	1	1	4
[42]	2025	0.5	1	0	1	2.5
[43]	2025	1	0.5	1	1	3.5
[44]	2025	1	1	0	1	3
[45]	2025	1	1	1	1	4
[46]	2026	0.5	1	0.5	1	3
[47]	2026	1	1	0	1	3
[48]	2026	0.5	1	0	1	2.5
[49]	2026	1	0.5	1	1	3.5
[50]	2026	1	1	1	1	4

3.2.1.1.1 Temporal Distribution

The temporal distribution of the selected studies was analyzed to understand research trends related to synchronous communication issues in GSD. The selected studies were published between 2009 and 2026, indicating a research interest in communication challenges in distributed software development.

Early studies focused on basic communication challenges and coordination mechanisms, while recent studies emphasize advanced communication technologies, real-time collaboration tools, and structured communication frameworks. This trend highlights the increasing importance of synchronous communication in modern distributed software development environments.

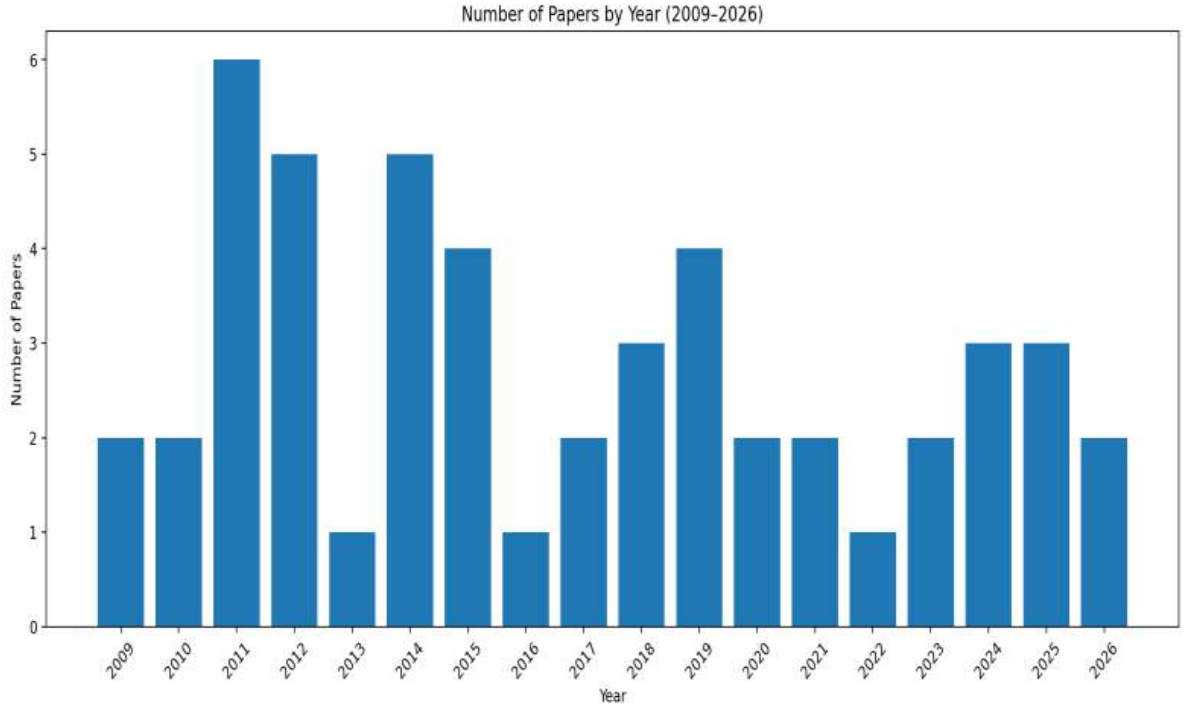

Figure 4. Temporal distribution of the selected primary studies

4. Categorization of Synchronous Communication Issues and Mitigation Strategies in GSD

The categorization of synchronous communication issues and mitigation strategies in GSD is presented in tabular form, providing a structured overview of the key challenges and corresponding countermeasures identified in the literature.

Table 4. Issues and Sub-Issues of Lack of Synchronous Communication in GSD

ID	Major Issue	Sub-Issues	References
IS1	Limited Overlapping Working Hours	Difficulty in scheduling meetings across time zones; reduced real-time interaction windows; coordination delays	[3], [4], [7][5], [15], [16]
IS2	Delayed Response and Feedback	Slow decision-making; increased task waiting time; feedback latency; reduced productivity	[6], [8], [11][12], [18], [19]
IS3	Communication Breakdown and Misunderstanding	Ambiguity in requirements; lack of immediate clarification; misinterpretation of messages	[3], [9], [10][20], [23],[24].[50]
IS4	Reduced Informal Communication	Lack of spontaneous interaction; weak team bonding; reduced trust and social cohesion	[14], [15], [17][21], [22], [37]
IS5	Ineffective Real-Time Coordination	Difficulty in synchronizing tasks; poor meeting participation; coordination inefficiencies	[13], [19], [20], [17], [31], [32]

IS6	Reduced Team Awareness	Lack of shared understanding; poor visibility of team activities; limited situational awareness	[23], [24], [28],[13], [23],[26].[48]
IS7	Dependency Management Delays	Waiting for responses from remote teams; blocked tasks; workflow interruptions	[24], [25], [26][18], [33],[34],[43],[44] ,[49]
IS8	Time Zone-Induced Communication Gaps	Non-overlapping work cycles; fragmented communication; handoff delays	[27], [28], [29][15], [16], [21]

Table 5. Mitigation Strategies for Lack of Synchronous Communication in GSD

ID	Mitigation Strategy	Sub-Strategies	References
MS1	Scheduling and Time Management Strategies	Establish overlapping working hours; Implement flexible work schedules; Rotate meeting times across regions; Apply follow-the-sun development model; Plan meetings in advance; Use shared calendars for coordination	[15],[16],[31],[33],[30],[34],
MS2	Adoption of Real-Time Communication Technologies	Use video conferencing tools (Zoom, Teams); Implement instant messaging platforms; Use collaborative tools (Slack, Teams); Integrate communication tools with DevOps platforms; Enable real-time notifications; Adopt unified communication systems	[37],[38],[24],[27], [28].[39], [40],[44].[47]
MS3	Structured Communication Practices	Establish clear communication protocols; Define response time policies; Conduct daily stand-up meetings; Document meeting outcomes; Use structured communication channels; Encourage concise and clear communication	[35], [36],[15], [16][43],[41],[42].[45],[46]
MS4	Team Collaboration and Coordination Enhancement	Promote informal communication; Build trust among team members; Conduct periodic face-to-face meetings; Provide communication training; Assign communication coordinators; Encourage team engagement activities	[21], [22], [37]

5. Overview of Issues for Lack of Synchronous Communication in GSD

S1: Limited Overlapping Working Hours

The geographical dispersion of team members across multiple time zones creates a severely restricted window for synchronous interaction. This limitation makes scheduling real-time meetings a logistical challenge, often forcing team members to participate outside of their standard working hours, which can lead to fatigue and reduced engagement. Consequently, the team loses the ability to engage in spontaneous real-time conversations, turning what could be a quick clarification into a scheduled event. This constraint delays coordination efforts and slows the overall pace of collaboration, as asynchronous communication becomes the primary, yet inefficient, default mode of interaction [3], [4], [7], [5], [15], [16].

IS2: Delayed Response and Feedback

In a distributed environment, communication often shifts from synchronous to asynchronous modes such as email or messaging platforms, which inherently introduce latency. This latency creates a ripple effect across the development lifecycle, where decision-making processes become protracted as teams wait for input from

geographically distant colleagues. Task waiting time increases significantly, as a single blocked task cannot proceed without a crucial piece of feedback. This persistent delay in the feedback loop not only reduces individual productivity but also disrupts workflow continuity, making it difficult to maintain momentum on time-sensitive deliverables [6], [8], [11], [12], [18], [19].

IS3: Communication Breakdown and Misunderstanding

The absence of non-verbal cues and the inability to seek immediate clarification often result in a high degree of ambiguity in distributed communication. When requirements or technical specifications are conveyed through text-based channels, the risk of misinterpretation increases significantly.

Team members may make incorrect assumptions about the context or intent behind a message, leading to errors that require rework. Without the ability to engage in a rapid, iterative dialogue to resolve confusion, small ambiguities can escalate into major discrepancies between expected and delivered outcomes, undermining project quality and stakeholder confidence [3], [9], [10], [20], [23], [24], [50].

IS4: Reduced Informal Communication

Physical distance eliminates the casual, spontaneous interactions—often referred to as "watercooler conversations"—that are essential for building social bonds and trust within co-located teams. In distributed settings, communication becomes predominantly task-oriented, leaving little room for social exchange or team bonding. This lack of informal interaction weakens interpersonal relationships, reduces social cohesion, and impedes the development of a shared team identity. Over time, this deficiency can erode trust among team members, making it more difficult to engage in the constructive conflict and collaboration necessary for innovation and effective problem-solving [14], [15], [17], [21], [22], [37].

IS5: Ineffective Real-Time Coordination

When real-time interaction is possible, it is often plagued by technical challenges, poor meeting structures, or disparate participation quality. Team members joining from different locations may struggle with audio or video issues, leading to disengagement or information asymmetry where remote participants feel like outsiders.

This often results in ineffective meetings where decisions are not clearly captured or agreed upon. Furthermore, the effort required to coordinate synchronous sessions often leads to fewer of them, forcing teams to rely on tools that are ill-suited for complex, real-time problem-solving, thereby reducing overall coordination efficiency [13], [17], [19], [20], [31], [32].

IS6: Reduced Team Awareness

Team awareness—the intuitive understanding of who is doing what, the current project status, and potential upcoming bottlenecks—is naturally eroded in distributed environments. Co-located teams benefit from passive information absorption through overhearing conversations or noticing a colleague's workload. In contrast, remote teams suffer from limited visibility of activities, as work is often conducted in isolated silos.

This lack of shared understanding and situational awareness makes it difficult for team members to proactively offer help, anticipate dependencies, or align their efforts with the team's immediate priorities, leading to duplicated effort or overlooked tasks [13], [23], [24], [26], [28], [48].

IS7: Dependency Management Delays

Modern software development is characterized by intricate task dependencies, which become critical failure points in distributed teams. When a team member's progress is contingent on input, code review, or a decision from a remote counterpart, the delay inherent in asynchronous communication can bring workflow to a halt. These dependency management delays result in prolonged task blocking, where developers are idle while waiting for external responses.

Such interruptions fragment the workflow, increase context-switching costs, and extend the overall lead time for feature delivery, representing one of the most tangible sources of inefficiency in global software engineering [18], [24], [25], [26], [33], [34], [43], [44], [49].

IS8: Time Zone-Induced Communication Gaps

Beyond the challenge of limited overlap, time zone differences create fundamental gaps in work cycles. When one team finishes its workday, another is just beginning, leading to a "follow-the-sun" model that, while potentially beneficial for speed, introduces significant fragmentation.

Information handoffs are frequently delayed by an entire work cycle, meaning a question asked at the end of a day in one region will not receive an answer until the start of the next day in another region. This creates a stop-start rhythm of communication that is difficult to manage, often resulting in lost context, repetitive explanations, and a persistent sense of lag in collaborative momentum [15], [16], [21], [27], [28], [29].

6. Overview of Mitigation Strategies for Lack of Synchronous Communication in GSD

MS1: Scheduling and Time Management Strategies

This strategy focuses on proactively managing temporal boundaries to maximize synchronous collaboration while respecting individual work-life balance. By establishing clearly defined overlapping working hours, teams create a reliable window for real-time interaction without requiring all members to work outside their normal schedules. Implementing flexible work schedules allows team members to adapt their personal hours to better align with global counterparts, while rotating meeting times ensures that no single region bears the disproportionate burden of attending early-morning or late-night sessions.

The follow-the-sun development model can be leveraged to pass work seamlessly across time zones, reducing handoff delays. These scheduling practices, supported by shared calendars and advance meeting planning, transform time-zone differences from a persistent obstacle into a manageable operational variable [15], [16], [30], [31], [33], [34].

MS2: Adoption of Real-Time Communication Technologies

The strategic deployment of real-time communication technologies serves as the technological backbone for reducing latency and fostering synchronous interaction in distributed teams. Video conferencing tools such as Zoom and Microsoft Teams enable richer communication by conveying non-verbal cues, thereby reducing misunderstandings that are common in text-only exchanges. Instant messaging platforms and persistent chat tools (e.g., Slack) support both quick clarifications and ongoing threaded discussions, effectively bridging the gaps left by limited overlapping hours.

Integrating these communication tools with DevOps platforms creates a unified workflow where notifications, code reviews, and deployment alerts are centralized, reducing context switching. Adopting a unified communication system ensures that team members can seamlessly transition from asynchronous to synchronous modes, making real-time coordination more accessible and effective [24], [27], [28], [37], [38], [39], [40], [44], [47].

MS3: Structured Communication Practices

While technology provides the channels, structured communication practices ensure that those channels are used effectively. This strategy emphasizes the establishment of clear communication protocols—such as specifying which channels to use for different types of messages (e.g., urgent issues via instant messaging, strategic decisions via documented email)—and defining explicit response time policies to manage expectations.

Daily stand-up meetings, adapted for distributed teams, create a predictable rhythm for synchronization and early identification of blockers. Documenting meeting outcomes and decisions prevents ambiguity and provides a shared record that is especially valuable when participants are in different time zones. By encouraging concise, clear communication and adhering to structured formats, teams can reduce information overload and ensure that critical messages are both received and understood [15], [16], [35], [36], [41], [42], [43], [45], [46].

MS4: Team Collaboration and Coordination Enhancement

Addressing the human and social dimensions of distribution, this set of strategies focuses on

building the trust, cohesion, and collaborative culture that are naturally cultivated in co-located settings. Promoting informal communication—for instance, through dedicated virtual watercooler channels or non-work-related team activities—helps recreate the spontaneous interactions that foster interpersonal bonds.

Periodic face-to-face meetings, even if infrequent, serve as powerful catalysts for trust building and resetting team alignment. Providing communication training equips team members with the skills to navigate cross-cultural

7. Propose Conceptual Framework

Quality assessment ensured only high-quality studies were included in the final analysis. Studies with a score below 50% on the assessment criteria were excluded. After the screening, Finally, 50 primary studies were selected for inclusion in the systematic literature review. Table 3 shows the detailed quality assessment scores for the selected studies.

The Systematic Literature Review identified several important issues related to the lack of synchronous communication in Global Software Development (GSD). These include a lack of overlapping working hours, delayed responses, communication breakdowns, limited informal communication, and limited real-time coordination. Different mitigation strategies were extracted and categorized to address these problems.

A conceptual framework has been proposed to provide a structured understanding of the relationship between synchronous communication issues and mitigation strategies. The framework shows how various strategies contribute to reducing the effect of these issues in distributed software development environments.

The proposed framework includes four major categories of mitigation strategies: scheduling and time management, adopting real-time communication technologies, structured communication practices, and enhancing team collaboration. These categories address the main problem of the lack of synchronous communication in GSD. They show potential to improve communication effectiveness, coordination, and collaboration.

8. Propose Hypothesis of the proposed conceptual framework

H1: Mitigation strategies related to scheduling and time management have a significant impact on reducing the lack of synchronous communication in Global Software Development (GSD).

H2: Mitigation strategies related to structured communication practices have a significant impact on improving synchronous communication in GSD.

H3: Mitigation strategies for incompatible development tools have a significant impact on reducing improper selection of technology in GSD.

H4: Mitigation strategies related to team collaboration and coordination enhancement have a significant impact on reducing synchronous communication challenges in GSD.

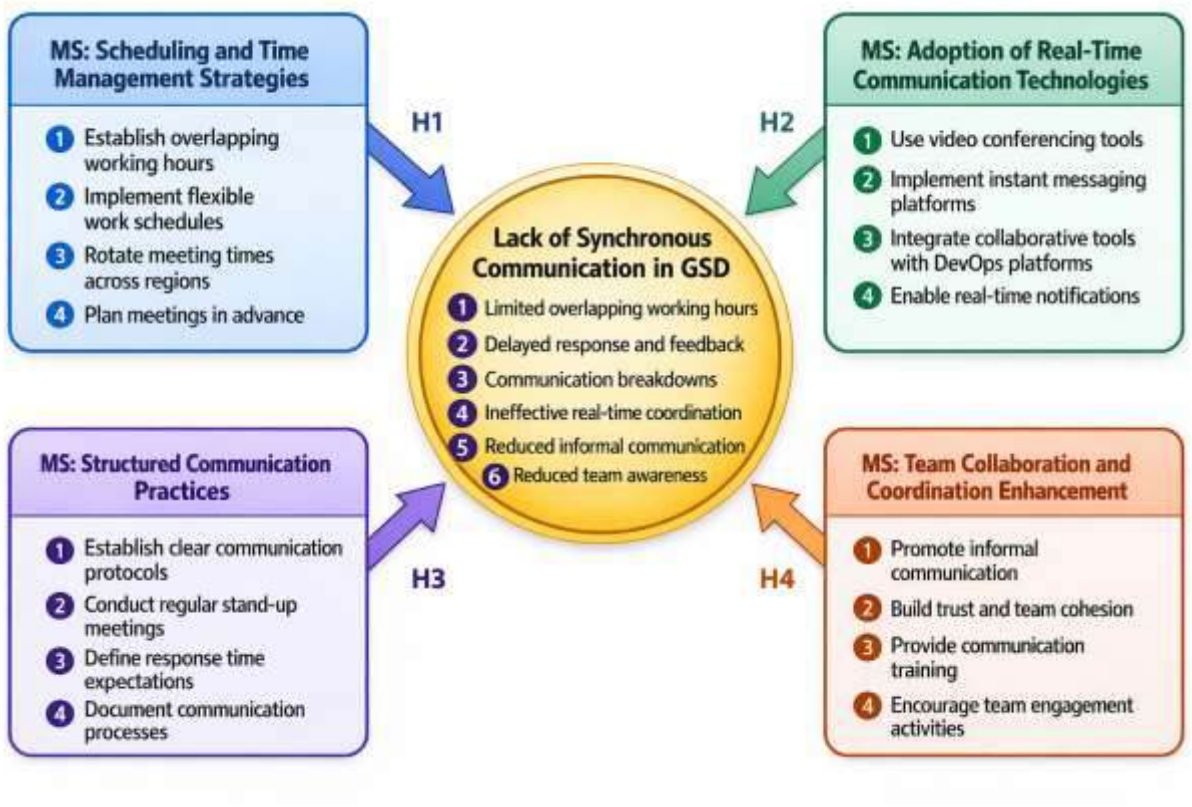

Figure 5. Proposed conceptual framework

9. Conclusion and future work

Global Software Development (GSD) has embraced a global software engineering approach as an essential practice in modern software engineering, and the benefits include cost efficiency, access to global expertise, and faster software delivery. However, the distributed nature of GSD poses several challenges that affect communication, coordination, and collaboration among geographically dispersed teams. Among these challenges, the lack of synchronous communication has been identified as a critical issue that affects real-time interaction, decision-making, and overall team performance.

This study proceeded with a Systematic Literature Review (SLR) to identify key issues related to the reduction of synchronous communication in GSD and to investigate the mitigation strategies suggested in the literature. Need for and description of interviews: 50 primary studies published during 2009-2026 were analyzed systematically using predefined criteria for inclusion and exclusion, as well as for quality assessment.

The results of this research showed several major problems, including low overlap in working hours, delayed responses and feedback, communication breakdowns, less informal interaction, and poor real-time coordination. These issues create significant barriers to collaboration and adversely impact productivity in a distributed software development environment.

To address these problems, several mitigation strategies were identified, including overlapping working hours, real-time communication tools, structured communication practices, and improved team collaboration. These strategies play a crucial role in improving communication efficiency and coordination among distributed teams.

Based on a systematic literature review, a conceptual framework was developed to assess the degree of correspondence between issues in synchronous communication and their respective mitigation strategies. The proposed framework

provides a structured framework for understanding and addressing communication issues in Global Software Development.

This study contributes to the existing body of knowledge by offering an in-depth analysis of the issues of synchronous communication and mitigation strategies in GSD. The proposed framework can serve as a valuable reference for researchers and practitioners in order to better communicate practices and improve collaboration in distributed software development environments.

There are, however, some limitations of this study. It only uses literature-based reviews; however, it does not include empirical validation. Future work will focus on the empirical validation of the proposed framework through studies of actual software development environments (surveys, interviews, case studies). Additionally, future research may investigate advanced communication technology, automated communication support systems, and intelligent collaboration tools to further improve GSD synchronous communication.

Overall, this study demonstrates the significance of efficient synchronous communication in the context of Global Software Development and the necessity of structured strategies to improve collaboration, coordination, and efficiency within a distributed software team.

10. Declaration

10.1 Author Contributions

Atta ur Rahman: Sole author and primary contributor of this study, responsible for the complete research process from conception to final manuscript preparation. Conceived the research problem, defined objectives, and designed the study. Conducted the Systematic Literature Review (SLR), including database searching, study selection, screening, and quality assessment. Performed data extraction, analysis, and synthesis to identify key issues related to lack of synchronous communication in Global Software Development (GSD) and their corresponding mitigation strategies. Developed the conceptual framework, interpreted the findings, and prepared the manuscript, including writing, revision, and final editing.

Fahim Muhammad Khan: Provided critical review and intellectual input throughout the research process. Assisted in refining the research objectives and validating the study selection criteria during the Systematic Literature Review (SLR). Contributed to the interpretation of synthesized data and provided key insights during the development of the conceptual framework.

Participated in manuscript preparation by reviewing drafts, ensuring methodological rigor, and contributing to the final editing and revision of the paper.

10.2 acknowledgement

No additional acknowledgments are applicable.

10.3 Conflicts of Interest

The authors report no conflict of interest related to this work.

10.4 Institutional Review Board Statement

School of Computer and Software, NUIST, Pukou District, Nanjing, China, 211544

10.5 Informed Consent Statement

Not applicable.

10.6 Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request. This study is based on a systematic review of previously published literature, and no new experimental or primary data were generated.

11. References

[1]. J. Noll, S. Beecham, and I. Richardson, “Global collaboration: Software development solutions,” ACM Inroads, vol. 1, no. 3, pp. 66–78, 2010.

[2]. J. C. Tang, C. Zhao, X. Cao, and K. Inkpen, “Your time zone or mine? A study of globally time zone-shifted collaboration,” in Proc. ACM Conf. Comput. Supported Coop. Work (CSCW), 2011.

[3]. J. A. Espinosa, J. N. Cummings, and C. Pickering, "Time separation, coordination, and performance in technical teams," IEEE Trans. Eng. Manage., vol. 59, no. 1, pp. 91–103, 2012.

[4]. M. Nordio et al., "How do distribution and time zones affect software development? A case study on communication," in Proc. IEEE Int. Conf. Global Softw. Eng. (ICGSE), 2011.

[5]. S. Dorairaj, J. Noble, and P. Malik, "Effective communication in distributed agile teams," in Proc. Agile Conf., 2011.

[6]. E. Hossain, M. A. Babar, and H. Paik, "Using Scrum in global software development: A systematic literature review," in Proc. Asia-Pacific Softw. Eng. Conf. (APSEC), 2009.

[7]. V. Casey and I. Richardson, "Uncovering the reality within virtual software teams," Int. J. e-Collab., vol. 5, no. 1, pp. 1–26, 2009.

[8].D. Šmite, C. Wohlin, T. Gorschek, and R. Feldt, "Empirical evidence in global software engineering: A systematic review," Empirical Softw. Eng., vol. 15, no. 1, pp. 91–118, 2010.

[9]. I. Richardson et al., "Global software engineering: A systematic literature review," Inf.
Softw. Technol., vol. 54, no. 4, pp. 347–360, 2012.

[10]. M. Paasivaara and C. Lassenius, "Collaboration practices in global software development," J. Syst. Softw., vol. 87, pp. 1–16, 2014.

[11]. S. Beecham et al., "Global software engineering: A systematic literature review," Inf.
Softw. Technol., vol. 57, pp. 767–781, 2015.

[12]. Z. Ahmed, et al., "Enhancing task related knowledge sharing in agile virtual teams: From influencing factors to practical guidelines–KSAVT framework," J. Theor. Appl. Inf. Technol., vol. 104, no. 2, 2026.

[13]. D. Harding, et al., "Geographical indication in Indonesia: A review on the spatial distribution and classification of geographical indication-registered products and-related publications," J. World Intellect. Prop., vol. 28, no. 1, pp. 263–285, 2025.

[14]. A. ur Rahman et al., "The temporal distance issues and their mitigation strategies in GSD: A systematic literature review," in Proc. 2021 4th Int. Conf. Computing & Information Sciences (ICCIS), 2021.

[15]. J. Herbsleb and A. Mockus, "Retrospective: An empirical study of speed and communication in globally distributed software development," IEEE Trans. Softw. Eng., vol. 51, no. 3, pp. 833–835, 2025.

[16]. Z. Liu, et al., "Towards a comprehensive framework for verifying open-source software license compatibility," Empir. Softw. Eng., vol. 31, no. 1, p. 15, 2026.

[17]. N. B. Moe and D. Šmite, "Understanding coordination in global software engineering," Empirical Softw. Eng., vol. 23, no. 5, pp. 2550–2590, 2018.

[18]. R. Prikladnicki et al., "Distributed software development: Practices and challenges," in Proc. IEEE Int. Conf. Global Softw. Eng. (ICGSE), 2011.

[19]. T. Jaanu et al., "Near-synchronicity and instant messaging in global software development," in Proc. IEEE ICGSE, 2012.

[20].V. Gomes and S. Marczak, "Problems and solutions in distributed software development," in Proc. IEEE ICGSE, 2012.

[21]. F. Calefato et al., "Global software engineering: Challenges and solutions," Journal of Systems and Software, vol. 174, p. 110887, 2021.

[22].A. Mishra and D. Mishra, "Cultural issues in distributed software development: A review," J. Softw., vol. 9, no. 9, pp. 2374–2381, 2014.

[23].M. Shameem et al., "Communication issues in global software development," in Proc. Int.
Conf. Softw. Knowl. Inf. Manag. Appl. (SKIMA), 2015.

[24].R. A. Khan et al., "Communication challenges and mitigation in offshore software development," Int. J. Adv. Comput. Sci., vol. 6, no. 5, pp. 45–52, 2015.

[25].H. Khalid and K. K. Farhat-Ul-Ain, "Root causes for the failure of communication in GSD," J. Inf. Technol. Softw. Eng., vol. 7,

no. 2, 2017.
[26].M. Yaseen et al., "Communication practices in global software development," Int. J. Comput. Sci. Issues, vol. 16, no. 2, pp. 55–63, 2019.
[27].U. I. Janjua et al., "Impact of communication issues in global software development," IEEE Access, vol. 7, pp. 123456–123470, 2019.
[28].M. Yaseen et al., "Coordination practices in global software development," J. Softw.
Eng., vol. 13, no. 1, pp. 22–30, 2019.
[29]. M. Yaseen et al., "Practices for effective software project management in global software development: A systematic literature review," International Journal of Computer Application, vol. 177, no. 36, pp. 1–6, 2020.
[30]. M. Yaseen et al., "Infrastructure and organizational improvement practices in global software development: A systematic literature review," i-Manager's Journal on Software Engineering, vol. 15, no. 1, p. 13, 2020.
[31]. A. A. Khan et al., "Effects of geographical, socio-cultural and temporal distances on communication in global software development during requirements change management: A pilot study," in 2015 International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE), 2015.
[32]. R. A. Khan and S. U. Khan, "Empirical exploration of communication and coordination practices in offshore software development outsourcing," Proceedings of the Pakistan Academy of Sciences: A. Physical and Computational Sciences, vol. 54, no. 1, pp. 41–54, 2017.
[33].I. Nurdiani et al., "Risk identification and risk mitigation instruments for global software development: Systematic review and survey results," in 2011 IEEE Sixth International Conference on Global Software Engineering Workshop, 2011.
[34]. A. Mathrani and S. Mathrani, "Test strategies in distributed software development environments," Computers in Industry, vol. 64, no. 1, pp. 1–9, 2013.
[35]. M. Shameem et al., "A systematic literature review to identify human related challenges in globally distributed agile software development: Towards a hypothetical model for scaling agile methodologies," in 2018 4th International Conference on Computing Communication and Automation (ICCCA), 2018.
[36]. S. Anwer et al., "Comparative analysis of requirement change management challenges between in-house and global software development: Findings of literature and industry survey," IEEE Access, vol. 7, pp. 116585–116611, 2019.
[37]. M. Paasivaara, C. Lassenius, and V. T. Heikkilä, "Inter-team coordination in large-scale globally distributed scrum: Do scrum-of-scrums really work?" in Proc. ACM-IEEE Int. Symp. Empirical Software Engineering and Measurement, 2012.
[38]. N. B. Moe et al., "From offshore outsourcing to insourcing and partnerships: Four failed outsourcing attempts," Empirical Software Engineering, vol. 19, no. 5, pp. 1225–1258, 2014.
[39]. I. Javed et al., "The impact of mitigation strategies for socio-cultural distance issues in GSD: An empirical study," IEEE Access, vol. 11, pp. 99499–99518, 2023.
[40]. A. R. Abd Shaheen and L. M. I. Alzubaidy, "Challenges performance facing global virtual teams in global software development: Literature review," in Proc. 2022 8th Int. Conf. Contemporary Information Technology and Mathematics (ICCITM), 2022.
[41]. J. Alqahtani et al., "Evaluating success factors of software project management in global software development," IEEE Access, vol. 12, pp. 22345–22358, 2024.
[42]. M. R. E. Indrajit, "The ten critical success factors in software development projects," Jurnal Teknik Informatika (JUTIF), vol. 5, no. 6, pp. 1633–1639, 2024.
[43]. J. Noll, S. Beecham, and I. Richardson, "Global software development and collaboration: Barriers and solutions," ACM

Inroads, vol. 1, no. 3, pp. 66–78, 2011.
[44]. A. Yagüe, et al., "An exploratory study in communication in Agile Global Software Development," Comput. Stand. Interfaces, vol. 48, pp. 184–197, 2016.
[45]. R. Qureshi, M. Basheri, and A. A. Alzahrani, "Novel framework to improve communication and coordination among distributed agile teams," International Journal of Information Engineering and Electronic Business, vol. 10, no. 4, p. 16, 2018.
[46]. R. Britto et al., "Effort estimation in global software development: A systematic literature review," in Proc. 2014 IEEE 9th Int. Conf. Global Software Engineering, 2014.
[47]. A. Reich and N. Reich, "Scrum in global software development: Challenges, risks, and mitigation strategies for effective project management," Journal of Policy Options, vol. 8, no. 1, pp. 43–50, 2025.
[48]. U. I. Janjua, A. Ahmed, and T. M. Madani, "Temporal distance issues and their mitigation strategies in GSD: An empirical study," KIET Journal of Computing and Information Sciences, vol. 6, no. 1, pp. 1–27, 2023.
[49]. G. Jean, "Cross-border e-supply chain coordination: Communication barriers and solutions," 2024.
[50]. A. A. Khan, S. Basri, and P. D. D. Dominc, "A proposed framework for communication risks during RCM in GSD," Procedia Soc. Behav. Sci., vol. 129, pp. 496–503, 2014.