Research Article

Saba Siddique, Muhammad Naveed, Atif Ali, Ismail Keshta, Muhammad Islam Satti, Azeem Irshad*, Zakaria Alomari, Onome Christopher Edo, and Oladapo Ayodeji Diekola

An effective framework to improve the managerial activities in global software development

https://doi.org/10.1515/nleng-2022-0312
received May 19, 2023; accepted August 1, 2023

Abstract: Global Software Development (GSD) is a contemporary approach to software development that offers numerous advantages, including enhanced cost-effectiveness and timely delivery. It enables access to a vast pool of skilled developers and facilitates the exchange of best practices and innovative ideas within the software industry. However, effective project management plays a vital role in ensuring successful product development. Organizations that achieve project success consistently adhere to well-defined project management methodologies, resulting in desired outcomes within predefined time frames and allocated resources. The success rate of software projects significantly increases with diligent software management efforts. Nevertheless, the distributed nature of GSD presents significant challenges related to collaboration, information dissemination, and process control, which ultimately impede effective development and compromise software quality. In this study, we identify various challenges associated with the GSD process and propose strategies to overcome obstacles to effective project management. Additionally, we introduce a comprehensive framework designed to enhance managerial activities in GSD.

keywords: SDLC, global software development, project management, quality control

1 Introduction

Global Software Development (GSD) is the process in which a software system is developed across two or more nationally and continentally separated locations. The Software Development Life Cycle (SDLC) phases are also geographically distributed, and the stakeholders involved are socio-culturally different [1,2]. The interesting GSD is remarkably increasing for its marketable benefits. Many industries have incorporated this technique into their software development practices. As the process of GSD takes place at different geographical locations, the components of the software are also distributed across different locations depending upon the size of the project. The distributed phenomenon reduces the development time and cost of the product. Customers get the deliverables on time, increasing their trust pool. Modularization also reduces the complexity of the software system which helps the customers in better understanding of the system. There are minimum chances of error propagation. GSD prevails over the sense of responsibility for development among various stakeholders. Large and highly skilled labor can be accessed, reducing labor costs. Creative and innovative ideas can be exchanged. However, the management area of GSD still lacks standards and strict planning for the successful completion of software projects.

It is inappropriate or lacks coordination in the development stages which is caused by communication problems, different cultures around the globe, work distribution tasks,
uncertainty in organizational process understanding, requirement and design, planning of the project, and continuation. Due to a lack of visualization and less communication, the requirements gathered are incorrect and insufficient. This poor management problem poses many other serious challenges among which the main issue faced in GSD is product quality. It becomes very expensive to remove big errors when the distributed modules are integrated, or the product comes into operation. It is very difficult to trace back the phases of the SDLC to find the problem area at which the problem occurred and rework it. For instance, the product may lack any feature or not behave the way that the customer wants when tested.

This problem can be generated due to any missing, incorrect, or insufficient requirement. There is also a possibility that the developer may have perceived the requirement in some other sense due to socio-cultural differences, lack of visualization, and communication. Hence, this problem will require more time and cost to be solved. The development process will be decelerated as it will have to backpropagate, and the trust pool of the customer will also decrease. Therefore, to avoid such issues, the GSD process requires such techniques through which it can be properly managed. Different strategies have been adopted so far to overcome these challenges but still, there is a dire need to address these issues comprehensively. Successful management will ultimately lead to the successful development of the software product.

In the GSD environment, the system development has made information and knowledge sharing an important element among team members that must be well organized. Many organizations have implemented GSD environment that faces many management issues. In this study, we conducted a literature survey from different but relevant research papers based on the following three formalized questions:

RQ1: What are the different challenges related to GSD?
RQ2: What are the possible solutions for acquiring better project management?
RQ3: How can we improve the managerial activities during SDLC in GSD?

2 Literature review

Most of the project's failure depends on unsuitable project management practices at team, company, and country levels. The methods and processes considered by developed countries differ from those of developing countries. The difference between vendor and client companies' perspectives is another issue. One focuses on skilled workers and saves costs; however, the latter wants a lucrative business and achieving projects [3]. The business model framework can facilitate the software companies to improve the project management and development processes [2]. Knowledge management deals with the communication and synchronization management of information regarding a project which affects the progress of the development of the project. There is very rare coverage on joined retrieval of knowledge, the ability to get new sources of knowledge, and how to handle tacit knowledge.

Noor and Rana [4] focused on using a common repository of information sharing which abridges the communication and coordination gaps. To improve the quality of products in GSD, Sanjay Misra et al. [5] highlighted the importance of quality assurance. Quality can be managed and enhanced by tracing the change occurring and controlling them through the management of risks and then properly reporting them. Proper planning for the development of projects near the customers will decrease the differences in culture and language. The active and thorough working relationship between the development team members during the project reduces the issues of communication and coordination. Strategic choices should be adopted to reduce the issues, for instance, outsourcing the parts of a project should only be considered for the non-culturally sensitive ones. Team members need to be skilled in other languages to reduce the communication gap. The exchange of team members across different sites for short intervals of time can improve the working environment.

Organizations are trying to adopt different strategies and methods for combining component-based software and a global-based environment. The challenges [2,4–9] that were identified during the management of globally distributed component-based software i.e., lack of understanding of ownership, decency of components on each other, insufficient process for the changes in the requirement, the difference between the cultures of team members working in an organization, and tracking of tasks and bugs are some of the challenges that we found in component-based GSD. The identified challenges can be used as guidance while developing strategies for component-based software related to GSD.

Component software development is a convenient and easy way of developing the system from the existing components, instead of using it from scratch. With the risks of GSD development, there is also some risk associated with component-based software development. Insufficient knowledge of components and interdependency between them may give rise to other project management challenges [7]. Jain and Suman [6] proposed a framework that combines the Project Management Book of Knowledge areas
for great management of globally distributed software, such as feasibility management for risk, scope, resource, virtual, and integration management are also included. Some of the factors that can help in completion of a project successfully are the involvement of customers, and project management techniques that will increase reliability, and reduce the development cost and time. Managing globally distributed projects is very complicated. Lack of planning, irresponsibility, delay in timing, and inexperience results in overall project failure.

The Four C [7,8] problems are identified related to the GSD challenges such as Communication, Coordination, Collaboration, and Control problems. Successful virtual management is based on four elements that are cohesion between teams, configuration, allocation of tasks between team members, and sometimes conflicts arise between team members which need to be resolved. A knowledge management framework for GSD may be comprised of a system school, cartographic school, engineering school, organization school, spatial school, and strategic school. However, it is an inefficient approach to cost management that may result in low quality output.

A virtual team can be established to carry out the project management life cycle (PMLC) activities in globally distributed projects [8]. Communication in project management can be improved if there is a limited time constraint between team members of an organization and the sociocultural and geographical differences among the team members are properly understood. Good interpersonal sensitivity skills will result in high-quality projects. Acceptance of others’ opinions and expressing what the members really want are important factors for achieving a high-quality product. The uncertainty of project completion is high in a global software environment. Due to the lack of methods and techniques, many risks associated with practitioners can result in project failures.

Arunumg et al. [10] proposed a framework, multi-agent simulation model (MASM), from which two agents called Global practitioner and Organizational agent can assess the risk associated with the global practitioners. MASM can be applied to an environment where the overall cost of developing the model is relatively high. Knowledge management issues can be generated due to a lack of communication, coordination, collaboration, and poor documentation. People usually do not have working experience in distributed environments [11]. The management of change is difficult in distributed environments. Data stores such as cloud computing can help to manage change [12]. Medium and large-scale organizations have closely related challenges of the requirement change management in GSD [13]. Less attention has been given to successfully executing the process improvement activities in the GSD environment.

Information sharing, building strong relationships, more staff involvement, a sense of responsibility, and skilled human resources can help to incorporate Software Process Improvement (SPI) techniques [14,15]. Because of integration complexities and incompatibilities, GSD integration issue arises, which in the end results in delay, loss of quality, chaos, and extra cost for the project [16]. The basic reason behind the issue of GSD integration is that the remote teams do not understand the working environments of distributed teams [17].

The members of teams at different locations and belonging to different cultures of the world follow different quality models, processes, and development standards. However, integration can be achieved by staying organized, keeping track of the processes, identifying the problems or risk a bit earlier by using good risk management techniques, and identifying failure factors like the class related to the single cause. Requirements are essential to successfully implement software projects [12,18–22]. Any change in the requirement during the development stage is unpreventable.

However, the change in requirement in the co-location is quite easy whereas in different locations, where the stakeholders are distributed across multiple locations, is difficult. This led to many challenges i.e., communication, collaboration, control, managing the central repository, and change in an efficient and effective manner [23,24]. Therefore, cloud computing can be applied to mitigate the challenges. The research [25,26] presents an improved framework called Reliability-centered maintenance (RCM) in GSD. The goal is to manage the requirement change in an efficient and effective manner. The proposed framework focuses on the processes of RCM and concerns of GSD are minimized.

During the literature analysis, the existing strategies to enhance project management activities in GSD were found to contain certain loopholes which need to be filled with further research in this domain [18]. Researchers and practitioners were invited to provide a more comprehensive solution to cover all the aspects of management.

3 Proposed framework

Many tools and techniques are accessible in the present world which addresses one or more than one problem but not all. As a solution to this, we have proposed a comprehensive framework that can help the teams to work more efficiently in a GSD environment. It can help to properly manage the managerial activities during the software development lifecycle and thus, reduce the failure rate.
which can be very expensive to afford if occurs at the end of SDLC. The proposed framework, as illustrated in Figure 1, has integrated multiple modules along with tools to provide a more comprehensive solution. There are six major modules, each dealing with a specific problem of managing activities in GSD. Each module contains important key elements which must be taken into account by the project management team. We have also associated the quality control module with project management modules to keep track of the quality of the product. For more accurate and efficient data management, all these modules are linked to a central data store.

3.1 Communication and collaboration module

The problems of the communication and collaboration module can be minimized by having a defined model to guide and support communication. Strong relationships must be established among the staff members to foster team spirit. Social activities must be promoted in face-to-face visits among the team members to build trust and personal relationships. Informal meetings should also be encouraged. Weekly Skype and Video conferencing will enable us to exchange information. Other collaborative tools must be adopted for communication like Instant messaging (IM), Short messaging service, Email, and Telephone.

1) Knowledge management (KM): A company's knowledge can be created, understood, shared, and distributed through KM. This knowledge can be used and reused at all levels of the organization. KM contributes to the growth and utilization of knowledge effectively by providing an accurate and complete understanding of the needs. When KM is incorporated into SDLC, it speeds up the development process, reduces the cost and produces a software with good quality. The defect density can also be reduced. KM can be enhanced by the usage of central data repository or virtual machines.

2) Content versioning system (CVS): It provides the same and latest piece of code or data to the team members working on the same project and keeps the data consistent. CVS acquires a lock when one member is working on a piece of code. The lock is only released when the member is done with working so that the latest version of the code is accessible to all.

3) Discussion board: Discussion board will incorporate all the opinions and questions made by the members regarding the working projects. In this way, highly experienced members' knowledge can be transferred to low experienced members and the whole team can be made powerful.

4) Language translator: Coordination and communication can be made effective by using language translator to help the team members in understanding different languages.

5) Cultural ambassador: Cultural ambassador can help to develop a common understanding among the team members coming from different backgrounds and culture.

6) Buddy system: Buddy system will assist in frequent knowledge sharing among the team members. New

![Figure 1: The proposed framework to improve the managerial activities in GSD.](image-url)
ideas can be exchanged while maintaining a friendly environment. It will also help the new employees to understand their roles and responsibilities.

7) Flow-mapping: Communication can be incorporated at all stages of the project through flow-mapping [27,28]. It helps to plan and manage communication easily in team-building.

### 3.2 Coordination module

The problems related to coordination can be minimized through proper planning and scheduling of all project management activities in GSD. Standardized work guidelines, deadlines, and commitments must be defined to manage the coordination in distributed projects. There must be certain rules for all the members of the project to work together. Frequent report sharing and deliveries will assist in managing up-to-date visibility of the project status. Tasks can be assigned and managed appropriately by adopting collaborative tools. Timely feedback can be obtained by shifting working hours across the sites in different time-zones. Here are some coordination models that play a significant role.

1) **Agile-based management**: Agile-based management facilitates the project team to coordinate more closely with each other and allocate tasks. Progress of the project is transparent to all the team members and can easily be tracked.

2) **Information quality management** (IQM): Within the GSD project management, IQM will help to assess and improve the data or information quality.

3) **DevOps**: The productivity of software projects can be increased with significant improvement in communication rate and accurate knowledge transfer. This can be achieved through substantial automation of the project-related tasks of software development. The reliance on automated systems has brought the operations and development teams into closer collaboration. The operational and development teams can have better communication, understanding, integration, and associations among them with the help of DevOps. Such project management teams encourage the construction of a system rather than just a software. Problems can be identified earlier which reduces the complexity of the problem and the time spent on error fixing. The changes can easily be handled and the productivity can be increased. Moreover, DevOps reduces the time of development and deliverables can be released earlier thus, return on investment and customer satisfaction will ultimately be increased.

4) **Near-shoring**: As people differ from place to place, lack of trust is common issue in cross-cultural environment. Therefore, the development should be made close to the customers and parent company. The cultural and language issues can be minimized this way. Requirements can easily be understood and gathered because of common or similar language and adjacent time zone. The developers can frequently interact with the customers. The sites which have same time zone are better options. The working hours can be extended at different sites.

5) **Unified access knowledge**: Knowledge sharing and knowledge management activities can be improved through unified access of knowledge [29]. This knowledge can be provided through any data store or cloud-based platform. New knowledge sources and any change made across any site can easily be notified.

6) **Strategic-choice**: Non-culturally sensitive components can be outsourced because it minimizes the issues of cross-cultural problems. For example, middleware and operating system are not sensitive to surrounding environment. They are independent software. So, they can be outsourced to decrease the development time.

7) **Synchronization**: It is an effective technique to improve coordination and collaboration in an organization.

### 3.3 Quality control module

Quality management is very crucial in order to meet the customers’ satisfaction and build trust relationships.

1) **Timeboxing**: Timeboxing each phase of PMLC and SDLC will help to obtain the product within the defined time frame. Often, schedule slippages lead to quality degradation. Timeboxing will also prevail the sense of responsibility among the team members of the project.

2) **Parallel verification and validation**: Any deviation from the conformance of requirements can easily be checked by parallel verification and validation. Time and cost can be utilized in an efficient manner as non-conformities are detected earlier rather than at the end of the PMLC and SDLC.

3) **Test plans**: Test plans help to schedule testing activities and execute them in an organized way avoiding duplicates. Problematic areas can be identified on a timely manner and the bugs can be eliminated.

4) **Essential documentation**: Essential documentation includes all the functional requirements, non-functional requirements, requirement rationale, and other necessary details. All the assumptions must be recorded.

5) **Quality audits**: Quality audits must be performed by the internal and external parties to ensure the quality
system [27,30]. Deficiencies and potential serious issues can be located and mitigated by using different measures. Potential opportunities can be identified while eliminating the waste. Quality audits also optimize the quality management process and reduces the cost of quality.

6) **SPI techniques:** By using SPI techniques, the goals can be achieved with higher development speed. Product quality can be enhanced and additional costs can be reduced.

7) **Transfer of knowledge and people:** The quality can be degraded by unknowledgeable or inexperienced staff at remote sites. The development team members should develop deep working relationships. Cultural negotiation can reduce the adverse effects of intercultural factors. By training staff members at different sites and exchanging them over short periods of time, the working environment can be improved [31].

### 3.4 Project planning module

In the beginning of the project, this module will estimate the cost, time, and effort in order to develop the software according to the given constraints.

1) **Cost estimation:** Before estimating the cost for the project, the size of the project must be measured while considering timing and technological constraints. Cash-flow plans must be formulated in order to forecast the cash inflows and outflows and keep better track of revenue. There might be few sites which require more budget due to high inflation rate. Therefore, Cost inflation must be analyzed across different sites along with market analysis. Additional cost for any delay or risk factor must also be taken into account. Cost estimation models like COCOMO and business metrics must be used for better cost estimation results. Projects’ performance and progress can be measured by earned value management. Important participants to administer the project contracts can be identified. Less traveling and face-to-face meetings will help to save cost.

2) **Effort estimation:** Before estimating effort, the size of the project must be computed along with the deadline provided by the client. Different effort estimation models and techniques, cost drivers, past experiences, available knowledge, and opinions of the experts can help to get more accurate effort estimation results. All the constraints and the risk that can generate during development, implementation, or deployment must be considered adding contingency hours. Work breakdown structure can also help to estimate effort for each individual task in more appropriate way.

### 3.5 Requirement change management module

Requirement change management in GSD is difficult due to lack of communication, coordination, and control. However, this management can be improved by using previous and expert knowledge. The knowledge along with requirement rationale can be stored in central data store and all the sites must have an access to use that knowledge to develop a project in distributed environment. Design protocol outlines the rules of requirement change management. The identified changes at any site must be reported to the change control board (CCB). The CCB must analyze the effect of these changes across all the sites along with risk analysis. Cost, schedule, and resources should be re-estimated. After evaluating the change, the decision should be made on the basis of expert reviews and voting system. All the changes must be documented with change rationale. Any change made across any site must be updated in data store as well to ensure that all sites are aware of the changes.

### 3.6 Management tools module

Different management tools are required to manage all the activities of PMLC.

1) **Virtual team manager:** Virtual team manager provides a set of effective guidelines and activities for training, developing, and managing. It can be adapted to different circumstances that usually arise in global software projects. It also assists in gathering and analyzing data in the time needed for preparation and launch.

2) **Resource manager:** Available resources including human, hardware, and software must be checked by the resource manager.

3) **Progress manager:** Task tracking which includes the completion status must be checked and managed by the progress manager. He will be responsible to generate progress charts and use progress meter to show the status.

4) **Document manager:** To maintain and share the documents during each phase of the project is the job of document manager.

5) **Report generator:** The data will be taken from the central database by the report generator. It will create a document that meets up to the requirements of a
particular stakeholder such as team members, project manager, or the clients and satisfy them.

6) **Task scheduler and updater:** Task scheduler and updater will provide the latest information about the tasks, schedules, updates of the project and status to the members. It will increase the productivity of work and reduce the irrelevant communication.

7) **Activity analyzer:** In order to keep track on team members, a screen shot is taken after small intervals by the activity analyzer. It will help a manager to keep track on members.

8) **Query tool:** This tool will enable the manager to effectively and efficiently access the database with location transparency [32].

9) **Bug meter:** Bug meter will help to keep track of work. The uncompleted tasks can be indicated in a project using this tool so that missing work can be completed.

### 4 Evaluation of the proposed framework

To evaluate our proposed framework, we conducted an online survey from different IT-based companies across the world belonging to two categories i.e., GSD companies and local development (LD) companies. We received 48 responses. 62.5% of the companies belonged to GSD category involving near shore outsourcing and offshore outsourcing whereas 29.9% of companies were developing software locally involving onshore outsourcing. About 81.3% of the companies were multi-sited. The percentage of the companies that participated in GSD projects was 75% of all respondents. The outsourcing or multi-sourcing partner countries for the abovementioned categories are shown in Figure 2.

![Figure 2: The outsourcing or multi-sourcing partner countries for GSD and LD companies.](image)

Figure 3: Conceptual model of GSD.

![Figure 3: Conceptual model of GSD.](image)

![Figure 4: Challenges faced by GSD companies GSD environment.](image)
Most of the companies were following Agile process model because Agile methodologies are more flexible than traditional ones in terms of change management. Agile helps in good management, rapid development, and accepts change in requirements at any stage of SDLC. Also, developers and testers can constantly interact with each other. A reliable software system can be developed in a shorter time. Few companies were following Scrum-based Agile methodology due to distributed chunks of functionalities among different teams. Scrum is the fast way of development when working on big projects and having to deliver on short time. Working in sprints, continuous meetings identify changes more easily which can be catered in future sprints. V-model was also adopted in few companies as testing across each phase of SDLC can identify errors early which saves time and cost.

The conceptual Model of GSD is shown in Figure 3.

As the development processes, developers, and information and technology are distributed, in order to identify certain challenges that are faced in such distributed project management, a checklist of possible challenges was provided in the survey form [33,34]. The results are shown in Figure 4.

Most of the issues that were found during the survey are related to coordination, project planning, and communication and collaboration. Incorporating testing in each phase of SDLC is considered to be expensive and time consuming, but if it is incorporated, it has more benefits such that the end product will be more reliable. As there will be less failures in real-time, company will gain people’s trust. Fixing errors at the end is more expensive and time consuming than identifying and fixing them at an early stage with less efforts.

About 97.9% of the respondents, as illustrated in Figure 5, considered incorporating testing at each phase of SDLC as an effective approach. For instance, unit testing is mandatory to satisfy use cases. Therefore, we have proposed an idea of using hybrid model (combination of Agile and V&V). This hybrid approach can be effective for GSD. Having centralized data stores will enforce unified access knowledge and effective management.

About 91.7% of respondents, as illustrated in Figure 6, considered our idea of having centralized data stores as an effective approach. A few respondents were of the view that data centralization can be dangerous and vulnerable to threats such as security breaches. However, data stores...
Figure 8: Management tools for effective management process.

Figure 9: Key elements that companies incorporate for effective communication and collaboration.

Figure 10: Key elements that companies incorporate for effective coordination.
Figure 11: Key-elements that companies incorporate for effective change control management.

Figure 12: Key-elements that companies incorporate for effective cost estimation.

Figure 13: Key-elements that companies incorporate for effective effort estimate.
can be protected by incorporating the best possible security measures.

The results of assessing the importance of other incorporated key elements in our proposed framework are shown in Figures 7–13.

5 Conclusion and future work

GSD is widely employed in various organizations where specific expertise is essential to address specialized software development and support needs. The GSD industry represents a distinctive type of business that has exerted significant influence over the past several years, necessitating access to expert resources and prompt availability. The increasing reliance on software engineering has led to a transformation in contemporary software development practices.

In the realm of software development, project teams are composed of individuals from diverse geographical locations and cultural backgrounds. This setup offers numerous advantages, including continuous development operations, cost-effectiveness, and round-the-clock progress. The pivotal technologies in GSD are communication and collaboration tools, which play a central role in the functioning of distributed teams. However, these tools come with inherent challenges that need to be explored further. Notably, these technologies must effectively manage requirements, track project progress, and ensure that milestones are comprehended and agreed upon by all members of the GSD team. Eliminating irrelevant project activities that hinder project development progress is crucial.

This research proposes a framework designed to address management issues in GSD comprehensively. The objective is to offer a more holistic and robust solution to the challenges faced in this domain.

In the future, the focus of the researchers will be to bridge any gaps that exist in the current study. They aim to implement the proposed framework in various IT companies to attain more accurate results, utilizing data mining techniques. By doing so, they seek to enhance the effectiveness and efficiency of GSD in real-world scenarios, fostering better collaboration, streamlined processes, and successful project outcomes.

Acknowledgements: The authors would like to express their gratitude to all reviewers who provided constructive feedback that helped to improve the manuscript, and to the Department of Computing and IT (DOCIT), TMUC, for support and motivation to make this research more effective.

Funding information: There was no explicit funding for this work from any public, commercial, or non-profit funding bodies.

Author contributions: Conceptualization: S.S. and M.N.; methodology: S.S.; framework: S.S. and A.A.; formal analysis: M.N., M.I.S., and Z.A.; investigation: M.I.S., and Z.A.; resources: A.I., I.K., and A.A.; data curation: M.I.S., I.K., and O.C.E.; project administration: M.I.S. and O.A.D. All authors have read and agreed to the published version of the manuscript.

Conflict of interest: The authors declare no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Data availability statement: The datasets generated and/or analyzed during the current studies are available from the corresponding author upon reasonable request.

References


Appendix

Literature review

**Table A1: GSD current issues and existing solutions**

<table>
<thead>
<tr>
<th>Reference</th>
<th>Identified problems</th>
<th>Proposed solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Saleem et al. [2]</td>
<td>Impact of critical success factors (CSF) at country, team, and company level and how CSF should be classified</td>
<td>Techniques for project management should be prioritized, good cooperation between team members</td>
</tr>
<tr>
<td>Noor and Rana [4]</td>
<td>Lack of tools for exchanging knowledge</td>
<td>Middleware repository for knowledge sharing</td>
</tr>
<tr>
<td>Misra et al. [5]</td>
<td>Lack of collaboration, requirement, and degradation of quality</td>
<td>Quality control, managing timely cultural sensitivity must be avoided, knowledge must be exchanged between team members and staff</td>
</tr>
<tr>
<td>Mahmood et al. [7]</td>
<td>Lack of problem understanding, cultural dissimilarity, task associations between team members</td>
<td>Not defined</td>
</tr>
<tr>
<td>Jusoh et al. [8]</td>
<td>Lack of virtual team communication</td>
<td>Shifting work-hours, understanding cultural dissimilarity, and open mindedness to others’ opinions</td>
</tr>
<tr>
<td>Vizcaíno et al. [9]</td>
<td>Lack of understanding the GSD model between computer science engineers and software engineers.</td>
<td>GSD-awareness game to educate people about GSD challenges</td>
</tr>
<tr>
<td>Arumugam et al. [10]</td>
<td>Lack of risk strategies and evaluation methods, uncertainties</td>
<td>MASM</td>
</tr>
<tr>
<td>Akbar et al. [13]</td>
<td>Lack of organizational support, politics in organization and budget problems</td>
<td>Not defined</td>
</tr>
<tr>
<td>Khan et al. [14]</td>
<td>Lack of SPI techniques</td>
<td>Not defined</td>
</tr>
<tr>
<td>Zafar et al. [16] and Khan et al. [15]</td>
<td>Failure of integration</td>
<td>Not defined</td>
</tr>
<tr>
<td>Gomes et al. [17]</td>
<td>Lack of trust, experience, team spirit; background and culture background are not same; absence of informal communication, lack of knowledge sharing, reduction in quality of communication</td>
<td>Cultural ambassador, communication infrastructure, buddy system, and communication model</td>
</tr>
<tr>
<td>Bibi et al. [35]</td>
<td>Lack of communication, coordination, control, central repository management, and change management.</td>
<td>Cloud computing, constant integration and maintaining repository</td>
</tr>
<tr>
<td>Minhas et al. [36]</td>
<td>Lack of communication and coordination strategies among stakeholders</td>
<td>Change in control board system, central repository, and voting</td>
</tr>
<tr>
<td>Akbar et al. [19]</td>
<td>Gap in requirements and existing requirement change model</td>
<td>Requirement change management</td>
</tr>
<tr>
<td>Lane et al. [24]</td>
<td>Lack of requirement and poor coordination, software quality degradation</td>
<td>Timely delivery, reporting, quality control; quality management, exchange of knowledge between the development team and staff on remote sites, avoid cultural sensitive components</td>
</tr>
<tr>
<td>Tariq et al. [25]</td>
<td>Lack of project management, planning, communication, and coordination</td>
<td>Content versioning system, telephone, IM, video conferencing and email, cost estimation, time estimation, and effort estimation tool</td>
</tr>
<tr>
<td>Lane et al. [37]</td>
<td>Distance and process challenges</td>
<td>Mechanism for communication and tracking communication</td>
</tr>
<tr>
<td>Rafael et al. [20]</td>
<td>Lack of communication and coordination strategies among stakeholders</td>
<td>Mapping and change of control board system, central repository, and voting</td>
</tr>
</tbody>
</table>