A Thematic Review of Global Software Engineering Concerns: Results from an MLR

Harley Martin[1], Jim Redmond[1], John Charles Ronan Kehoe[1], Ruth Leavey[1]

[1]School of Computing, Dublin City University, Glasnevin, Dublin, Ireland
{harley.martin5, jim.redmond26, john.kehoe9, ruth.leavey2}
@mail.dcu.ie

Abstract. This paper investigates the concerns of communication, geography, culture, agile approaches, training and education experienced within global software engineering environments. Studies show that global software engineering is gaining more traction among organizations in the software engineering industry. Additionally, Global software engineering can aid these organizations by reducing costs of engineering software products. However, benefits of a distributed software engineering setting can be limited by the many challenges and concerns associated with the process. These concerns may be present in distributed settings because of superficial reasons attributable to differences in communication, coordination, development frameworks, knowledge skills and cultures.

Keywords: Global software engineering (GSE), Software Process Improvement (SPI), Communication, Software P, Agile programming

1 Introduction

Global Software Engineering has become an established paradigm of software development which involves collaboration on the same project by globally distributed teams [13]. The International Conference of Global Software Engineering (ICGSE) 2015 industry keynote noted that 30% of embedded software is developed in global or distributed environments [17]. A 2016 paper from the 11th ICGSE found that in the previous decade, global software engineering and IT outsourcing had grown at rates of 10-20% per year [23]. Its application rates are growing but it is not always successful. The same paper noted that across the world 20-25% of all outsourcing relationships fail within the first two years and 50% fail within five years [18, 23].

There are several benefits to GSE which make it a popular approach in modern software development. Reduced cost is the main trigger for organisations to distribute their development globally. The benefits of this are reduced by the well-known complications of the process which result in hidden costs and many additional overheads [17]. Distributed environments may result in a project taking 2.5 times more effort than one developed by collocated teams [16]. The aim of this paper is to analyse these complications and discuss the findings.

The first concern this paper discusses is worldwide communication. For software development projects to be successful, effective and efficient communication is of utmost importance. The different types of distance (geographical, temporal, and socio-cultural) that are inevitable in GSE make this very difficult [20]. Geographical distance in this context is not measured in kilometres or miles. It refers to the ease with which the participants in GSE can travel to each other's locations. Temporal distance refers to the time-zone differences between the numerous countries where the participants of GSE are located. The beginning of the workday in site A can happen several hours before the beginning of the workday in site B. Socio-cultural distance refers to the gap between two participants in knowledge and understanding of each other's cultures, values, and practice [13].

The second concern this paper discusses is the compatibility of GSE and agile programming. Agile programming has been a popular method of software development since the 1990s [h14]. According to the Agile Manifesto, "Business people and developers must work together daily throughout the project" [16]. The adoption of methods like scrum and sprints fulfil this need. IT infrastructure, and geographical, temporal, and socio-cultural distances pose obstacles to these and other agile methods because communication plays a primary role in agile development and these factors impact communication [16]. These obstacles have been reported as early as 2003 but as our research into recent studies has confirmed, they still have not been resolved as they are very difficult to overcome [16]. Optimal technologies and tools for efficient communication have been explored in several studies but the problem is still very much evident.

The final concern discussed in this paper is the education and training needed for GSE participants. Soft-skills are needed for collaboration with teammates and other teams. This is already problematic in collocated environments, so distributing participants only worsens the problem. Since GSE is a relatively new paradigm, many university courses do not specifically teach it [13]. When graduates enter the workforce, they do not have the required education and training to work in a GSE environment and usually face many challenges. Development companies teach this to new employees but it has become increasingly important for educational providers to account for this lack of knowledge and skill in their students in order to prepare them for employment [13].

2 Literature Review Methodology

2.1 Methodology

The method of research undertaken for the completion of this report was to identify, evaluate, and interpret the available papers and articles published in relation to our research topic. The research was carried out by a team of 4 as part of a computer science undergraduate degree. The aim of this MLR was to identify themes of global software engineering concerns.

In order to carry out this research, Google Scholar was utilised to search for literature on the topic. The search string used during the initial researching phase was "global software engineering". After the original search string had been entered, papers of interest were identified and gathered. The contents of these papers were examined with the aim of identifying themes that were suspected of being concerns in global software engineering. The most common concerns were found to be Geographical, temporal, linguistic, cultural concerns, technology, IT tools, agile approach to development concerns, communication, collaboration, teamwork concerns, education, training, skills concerns, project management, requirement engineering concerns, trust, and employee relationship concerns (see figure 1 in appendix).

For each theme identified in the initial search, further research was carried out and discussed to determine which concerns would be chosen for this specific research paper. It was discovered during this step that many of the topics were closely related and could be associated under four main headings which were decided as our initially chosen themes for this review; communication, geography and culture, agile approach to development and training and education (see figure 2).

These themes were recursively examined for sub-topics which were used in further reforms of the search string until adequate information was found. Search strings used throughout the process of this review are represented in figure 3.

It is important to note that for the purpose of this review the theme of communication as well as geography and culture were merged to facilitate commonalities relating to both.

2.2 Inclusion/Exclusion Criteria

Upon searching the search engine, the search results time range that was considered was from 2015 to the present year 2021. However, exceptions were made to include papers from outside of the set time range given that they provided a significant reinforcement of arguments. Only the top results from the first 3 pages returned were viewed. The main papers that were considered were from publishers of high ranking and mainstream Software Engineering journals and conferences, including IEEE, Wiley, Springer, ACM, Elsevier, ScienceDirect. With these guidelines in mind the inclusion criteria was one of relevance and reliability with which the framework of this paper was based on.

3 Analysis

3.1 Communication, Coordination and Control in GSE

In traditional software development settings, it is widely accepted that communication, coordination and control are integral factors of the software design process [5]. Communication is defined as the exchange of information between two or more actors where a common understanding is found. Coordination can be described as the integration of each task into a unit of an organisation. These units then work together to achieve a unified goal or end product. Control is the process whereby parties follow goals, policies and standards set by the organisation. These factors are integral to successful collaboration within a team [5]. The significance of these factors is amplified when working in a GSE environment [4]. Global Software Engineering struggles to achieve successful collaboration among team members, between different teams, between management and teams, and between developers and users [15, 6, 7].

Communication is an issue in software development as a whole, but it is even more challenging when all of the people involved are spread across the globe. In these cases, the selected medium of communication is of particular significance [20]. Teams may communicate verbally with each other through the use of phone calls, text messages and video calls [4, 16, 21]. Non-verbal communication between teams happens through the use of schedules, various collaboration tools and online wikis [17, 4].

3.2 Factors Affecting Communication, Coordination & Control

Four of the most commonly noted hindrances to effective communication, coordination and control in GSE are geographical, temporal and socio-cultural distance, and communication technologies. GSE by nature generates geographical, temporal and socio-cultural distances in the working environment due to the fact that teams are operating in a distributed manner. While technical aspects of the software development process are often considered to be the key elements of a product’s success, the more nuanced aspects must also be considered [1]. When organisations turn to a dispersed working environment, which is often due to the many economic benefits associated with GSE, issues can often arise due to geographical, temporal and socio-cultural challenges [2-3, 7]. The impact each of these factors have on communication, control and coordination are detailed here [6].

Geographical Distance. Geographical distance measures the effort required for one actor to physically visit another [4]. Distance is not necessarily measured in kilometres - two locations separated by a large physical distance can be said to be close if there are regular transport options (for example flights) available, whereas closer locations with poor transport links could be considered to have a greater geographical distance between them [12]. Geographical distance affects the amount of opportunities for social interaction in the workplace [10]. It is a challenge to build strong employee relationships with a sense of community and teamwork among GSE teams. Teamwork is a highly sought out soft skill that will contribute immensely to the success of powerful and efficient collaboration. [17] It has also been found to encourage knowledge sharing [17]. Frequent social contact is essential when building trust within a team [10]. It provides an appropriate platform for negotiations and discussions [20]. The absence of opportunities for informal conversation prevents collaborators from forming a personal bond and "getting to know each other". In their 2005 case study, Kotlarsky and Oshri attempt to show the importance of human interaction among teams and its importance to collaborative work in a GSE context. They carried out research in collaboration with SAP and Teledyne LeCroy, both of which have teams located across multiple sites. Through this research they showed that if social ties can be made within a globally distributed team, collaboration and communication increase among the group. They identified the difficulties of building trust and rapport among the team members. This in turn had a negative impact on the transactive memory of the distributed team which hindered the collaborative nature important to SPI [9]. The study displayed the difficulties faced by GSE teams due to scarceness of informal communication between remote colleagues and unfamiliarity with each other's culture. Lack of a team essence leads to "them-and-us culture" [15]. This may be exacerbated by the fact that onshore team members are not interested in sharing their tacit knowledge with the low salaried competitive offshore counterparts, for fear of losing their job to offshore members who are less expensive to the company [53].

Temporal Distance. Temporal distance measures the difference in work patterns among different actors within an organisation. This distance is often caused by time zone differences [4]. Temporal distance is caused by the large physical distance among the individual actors trying to communicate. Communication is often disturbed due to time zone differences within a distributed team as there may be limited cross-over in their working days [12]. This makes it difficult to achieve synchronous communication where all members can gain unanimous understanding of the current status of a project at the same time, as it is hard to find a convenient time for everybody. [15, 5, 54] Updates on emerging and changing issues or requirements may not reach all participants in a timely fashion, which leads to misunderstandings, conflicts and wasted time working on changed features [15,24]. The administration, planning, preparing, documenting, and reporting of meetings involve a lot of overheads, which reduces time spent on the project at hand [15, 16].

A 2017 study undertaken by Hoda *et al.* attempts to better understand challenges faced by software engineering students who had undertaken cross-university collaborative courses to gain GSE skills in a practical manner. The courses involved typical GSE practices such as video calls, email and instant messaging technologies, and involved ten universities across eight countries. It was found that expectations of time and timekeeping varied across different cultures. Asian participants, for example, were found to be late for meetings more often than their European and American counterparts. There were also expectations on some participants to be available outside of working hours and weekends which led to difficulties in finding common times to communicate [13]. Other studies have found that this leads to offshore employees experiencing frustration and burn out as a result of working late hours [15].

Socio-Cultural Distance. Socio-cultural distance measures the extent to which one actor understands another actor's cultural and social norms [4]. Socio-cultural distance can in some cases have a huge effect on how people perceive and respond to certain work environments. Holmstrom *et al.* describe culture as being a “complex dimension, involving organisational culture, national culture and language, politics, and individual motivations and work ethics” [11]. There is a close relationship between culture and communication, and language goes hand in hand with that [15]. Misinterpretation can result from the variety of ways in which different cultures use speech, text, body language and behaviour to communicate their intentions. This is made all the more complicated due to the use of communication technologies [20, 21, 54]. Different styles of communication have different meanings across the globe, for example; formal/informal, direct/indirect speech, deference to hierarchy, and not being able to say ‘no’ [53]. Research examining cultural gaps finds fewer negative effects of the cultural gaps among participants in differing Western countries than between Eastern and Western countries. [17]. A typical measure taken to reduce language barriers is to have a GSE team that shares a common language. An example of this is seen with cross-Atlantic software development between the USA and Irish developers [6].

However, this is not always the case as even when there is a common understanding of the shared language within the team there can be issues with interpretation because of accents and terminologies. Expected differences and actual differences between cultures can also make it difficult to build and maintain employee relationships. Differences in ways to express dissatisfaction, perceptions regarding punctuality, scheduling, hierarchy, urgency, or risks, often leads to misunderstandings among participants [15]. Stereotypes, lack of understanding and assumptions about national culture, values, and practices influence a person’s judgement of their overseas colleagues [12, 13, 54]. One culture’s intentions may be misinterpreted if their style of communication involves some type of mannerism that is deemed inappropriate by another culture. This can reduce trust among groups [15, 20]. Trust is one of the key factors in successful teamwork and collaboration on a project [20]. The Stereotype Content Model is a theoretical framework for the conceptualisation and operationalisation of stereotypes. It is two dimensional, with warmth referring to how welcoming and friendly a country’s stereotype is perceived to be, and competence referring to how capable and efficient a country’s stereotype is believed to be. It appears the higher warmth and competence a country is associated with, the more likely they are to be trusted by others [19]. Published in 2017, an empirical study involving multiple organisations with participants ranging from CEO’s to software developers identified socio-cultural distance as a major barrier to SPI in these GSE environments. The same study highlights that language differences are a contributing factor to this breakdown in communication [8].

It is useful to look at how these factors can influence GSE as a collective. In their 2015 pilot study, Khan *et al.* investigated how geographical, temporal and socio-cultural factors had an impact on communication and therefore on the collaborative nature of software development. A survey questionnaire conducted focused on two areas; demographic profile and effect of geographical, socio-cultural and temporal distance on GSE process. The study found that there is a negative relationship between these factors and communication within the GSE environment [12].

Communication Technology Issues. In person communication is the best way to resolve issues and misunderstandings. [20] However, this does not exist in GSE so communication technologies must be used to simulate reality as much as possible. Distributed teams, especially in agile GSE, depend heavily on these technologies [16]. The quality and usage habits of these tools affects communication among GSE organisations. The wide range of communication tools available can allow for several types of communication; directional/bidirectional, synchronous/asynchronous, one to one/one to many, and rich/not rich. Rich communication is defined as two-way interaction involving more than one sensory channel [2]. Existing technologies such as e-mail lists, telephone calls, teleconferences, instant messaging, wikis, screen sharing, and video conferencing are a best effort at mitigating communication issues but they do not entirely solve the problem. [15]

Synchronous tools may not be made use often because of the struggle to find an appropriate time for everybody involved. Therefore, asynchronous tools tend to be utilized more frequently, although they slow down the processes of decision making, issue clarification, etc. and thus delay the completion of the project. [15]

The mismatch of tools used between sites also impacts communication and coordination because it can create loss of data transfer, creating an incomplete understanding on the other side. [15] Inadequate hardware configuration, mainly connection speed at remote sites creates a lag in communication too.

3.3 Geographical, Temporal & Socio-Cultural Distance, & Communication Technologies Issues Affect on GSE Lifecycle

The imperfect communication caused by geographical, temporal and socio-cultural distance, along with communication tools and technologies, negatively impacts each phase in the lifecycle of project development individually, and the quality of each phase affects the quality of the following phase [15, 20].

The communication and collaboration intensive nature of the requirement engineering phase make it the most challenging phase of global software development. Differences in terminologies and in level of detail when describing user needs during this phase can lead to an incorrect understanding of user needs and an inappropriate design of product [15]. A shared vernacular leads to a good understanding of requirements, but common vernacular is rare among native speakers of a language and even more rare when non-native speakers are involved. People belonging to different cultures have different explanations and translations of requirements [20].

Poor involvement from the client and insufficiently detailed requirement specifications result in incomplete knowledge. When this happens, some of the doubts often remain unresolved, meaning developers must work based on their best assumptions. Incorrect assumptions are practically inevitable and this results in requirement change which increases delays [15]. Requirement change management is difficult because it needs proper negotiations and discussion which are difficult without face to face meetings, so the same issues are likely to arise [20].

A common understanding of customer requirements and architecture is very difficult to achieve among different teams composed of large numbers of people who all come from different backgrounds [15]. Variations of understanding of requirements impact collaboration between teams as they are not working towards a common goal, so queries, discussions, and negotiations arise which slow down development [20].

The rest of the phases are mainly impacted by the requirements elicitation phase but they may pose their own problems too. In the development phase, as participants from different organizations collaborate on code, they sometimes follow different processes which can cause trouble when their individual work must be combined. Integration problems can cause delays or even failure which leaves customers unsatisfied [15].

During testing, when the completion deadline is closest, time-zone differences disrupt the back and forth communication and cause delays. Testers may have to resort to quick fixes on bugs rather than systematically fixing them, in order to get the product out before service level agreements are breached [15].

Time and effort required for integration is often underestimated. Implementing different code branches at each development site can lead to merging problems which may not be exposed until integration where they can become very costly. When integration fails, the same issues mentioned previously like time-zone and cultural differences, communication tools and lack of feeling of a team delay the fixing of these issues [15].

3.4 Agile Processes in GSE Context

Agile practices have been the recipient of attention from the software engineering industry as an alternative to plan-driven and linear software engineering approaches. Agile practices can encourage smaller, self-organized, collocated teams, whereas GSE involves distribution across cultural, temporal, and geographical boundaries. Therefore, combining them together is speculated to create challenges [24]. A further examination into some of the agile practices can highlight the possible concerns that may be experienced in a GSE environment where the agile practices are established.

Scrum. Among all of the various forms of agile practices, the Scrum method is considered to be the most frequently used agile practice in the modern software engineering era. The Scrum method is also increasingly being incorporated into distributed software engineering environments [25]. This trend seemingly creates a contradiction as the Scrum methods require a solid and close form of communication which would normally be experienced in a connected setting but may not be experienced in a distributed setting [25]. This suggests that some of the Scrum tactics could obtain causes for concern when being incorporated into a GSE environment.

Scrum meetings are short meetings (usually 15 minutes long) among team members where each member must answer three distinctive questions based on their previous meeting, what they plan to do in the future and also the obstructions that they may face [26]. These meetings can occur on a planned or unplanned basis. This Scrum method is integrated into software engineering as a way to devise a plan for all team members to approach their daily challenges and to understand each member's input towards the project at hand. In order for Scrum meetings to be used effectively in GSE these types of meetings must be held frequently. It seems that there is a concern present with this method in conjunction with GSE as non-overlapping time frames between group members means that unplanned Scrum meetings can occur too infrequently and out of proportion to planned meetings. Thus, this can hinder coordination between group members [26].

An alternative Scrum method used in software engineering are Sprints. Sprints are applied to software engineering teams to fulfill certain requirements before a specified deadline (usually 30 days) which are listed in a backlog (a list of functional requirements in a scale of preference) [26]. In a normal software engineering environment, the benefits of the Sprint approach are that the client can see the product and share any relevant feedback on it, which will direct the software development team into considering changes that are essential in meeting the client’s expectations as well as the client being able to get a sense of direction for the future of the product [28]. A possible concern for using the Sprint method in a GSE setting would be that there is a potential difference in knowledge between different group members [29]. This suggests that some group members may have to input more of a workload into meeting requirements which may lead to deadlines being missed and the product vision being unclear.

Lean development. Lean development practices initially originated in a manufacturing unit of Toyota, where it provided an efficient system of continuous flow for production [30]. Lean approaches are a form of a subset of Agile approaches that focus on disposing waste. Lean software development starts with value orientation, then reducing unnecessary features, improving the interfaces, empowering the software engineers, and continuously advancing the solutions [31]. Many organizations which have implemented lean development methods have reported the practices to be successful. However, they still face several challenges daily due to several loopholes in the structure when used in a distributed software engineering environment [30].

Kanban is a software development methodology which applies Lean principles. Kanban is becoming increasingly popular and is being used to better Scrum and other Agile methodologies. Although Kanban's popularity is rising, many questions with regard to its implementation in GSE are still not answered. Software engineers face serious concerns while implementing Kanban as clear definitions of its practices, principles, techniques, and tools are lacking [31]. An apparent main concern of adopting the Kanban approach in GSE is that it can clash with different team members' organizational culture. In essence, the implementation of Lean practices can often require a shift in organizational culture and processes, this may not be easily achieved in a distributed setting where group members may be used to operating with different types of organizational processes [31].

Extreme Programming. Extreme programming was designed for use with 10–12 co-located, software developers. Extreme programming is an iterative development methodology with a short planning time scale (1-2 week iterations). In extreme programming, team members work towards a release which is a deliverable version of a product that can be used by customers by a given date (3 month long release). Iterations are shorter increments of development where individual tasks are assigned to the software developers and a working prototype of the system is evolved and regularly evaluated by the project's stakeholders [32]. Consequently, this advocates that extreme programming is dependent mainly on a solid form of communication between the stakeholders and tight feedback loops to specify feature utilization and to adapt to change. This means that extreme programming potentially poses concerns towards GSE teams.

Since communication is considered as one of the most important factors when applying extreme programming in software development, it is treated as being a success factor in allowing for extreme programming to be undertaken efficiently [33]. It can be derived from this information that extreme programming's reliance on informal communication, as well as the use of an unfamiliar development methodology, can be presented as a concern to GSE groups [32].

3.5. Training and Education for GSE Industry

This section details GSE concerns in relation to training and education. Do employees lack skills and knowledge necessary for work in the global environment? What is the importance of corporate training and how crucial is training to work practices and project success within GSE? Would teaching third level students prepare professionals better to work in GSE and does the lack of a university teaching standard for GSE add to concerns expressed about training in this industry? These are questions which will be analysed in this section.

Software engineering success factors are related primarily to human elements, skill being one of them [34]. A great number of potential challenges can arise in GSE projects. Many of the factors which lead to these challenges relate directly to competencies that are amenable to training and education. Some of the human factors of important concern in GSE include communication skills, collaboration skills, knowledge of a common language and socio-cultural awareness. Each reflects an area where preparing and training professionals with specific skills could prevent problems from occurring and/or recurring [35].

Pertaining to a failed agile software development project, a case study performed in 2019, detailed the human factors which contributed to the project's failure. The paper determined not only that team-related skill deficits was one of the top two challenges regularly encountered in agile software development, but also that insufficient skill came as a result of a dearth in both education and training among team members [34]. Team-related problems can derive from members having varied soci-ocultural backgrounds, working in different geographic locations and across multiple time-zones. Appropriate training results in professionals being better prepared to avoid allowing these issues to create conflicts or difficulty [36]. Socio-cultural training among GSE teams is identified as effective in developing understanding among workers and crucially to contribute to successful project experiences and outcomes [37].

In contrast, insufficient training regarding human factors increases the chance of facing them repeatedly. A research paper studying communication in global software engineering among virtual teams deemed that lack of appropriate training was the largest determinant of poor communication within a team. This point was agreed by 66% of the survey respondents [38]. Insufficient training in potential sociocultural and geographic issues was similarly signposted as an important issue in a paper which suggested many adaptations in project management that could help avoid failures in global software development [39].

An important onus falls on the project manager to help alleviate the many difficulties which may arise within a GSE project as a result of team member knowledge base and skill-set [35 7]. Therefore managerial competence in understanding team members training and educational backgrounds and requirements is very important. A project's success and failure are often associated with project management competence and specifically the approaches taken to assessing and providing training and education at all levels [35 7].

A specific concern in GSE relates to high staff turnover [41]. It results in the loss of skilled professionals who have valuable experience with the project and corporate practices. This in turn intensifies challenges in terms of corporate training and education with a constant need to focus on the new staff engaging with the project [40]. How staff turnover is planned for and managed can have a significant impact on a GSE project's success [41]. An ongoing focus on continuous training, although this incurs a cost, can prevent problems from reoccurring.

In the last decade (2010-2020) GSE has grown rapidly, with 10-20% growth rates per year reported in GSE among IT outsourcing as well as business process outsourcing [42]. During the same period of time only 9.5% of research in the field of software engineering-education has been related to GSE [43]. This discordance illustrates a major concern that there is a need for third level institutions to enhance their focus on preparing students for work in GSE.

Studies indicate that existing education practices in software engineering have become "outdated and lack authenticity". It is believed that software engineering students should be introduced to the challenges of GSE to help prepare for their future careers [44]. A number of approaches have been employed within universities, however there remains debate as to which is most effective [45]. A potentially attractive approach is a distributed, cross-university one, which involves project collaboration between students from several different institutions. It strives for improvement in cultural awareness and understanding, global collaboration and communication as well as providing experience with certain complexities that arise when working in this type of environment [45, 46]. However, this educational approach does not come without creating some issues. Pedagogy-related challenges such as "educator overheads" and variation regarding "enrolment systems, grading systems and course volatility" can all occur and have deterred institutional engagement [44, 47, 48]. A different practice that has been tried is the assignment of open-source projects. Although avoiding some of the challenges existing in multi-university GSE education, it may not provide sufficient experience and learning, particularly regarding soft skills, required for working in the GSE industry [45, 46]. The adoption of simulated games is yet another approach to overcoming some of the pedagogical challenges. It aims to provide student experience within communicative, collaborative and socio-cultural contexts. However, the success of this approach is based on the standard of the simulated game being used [46, 49, 50].

In the growing field of GSE, many skills both technical and non-technical are needed for success. Experience with these skills is imperative and lack of training presents concern as a contributor to failure. Another important concern in GSE lies with slow educational development forcing only corporate training to target competencies in these skills. More academic research and creative realignment of educational processes will be required if professionals are to be better prepared in the future for work in GSE.

5 Future Direction for Research

Over the past 10 years GSE has been on an upward trajectory as more organisations look to offshoring due its many economic benefits [7, 22]. Now, even more so we expect to see a greater shift towards distributed teams. In their 2020 study, McCarthy *et al.* of NUIG conducted a study in two phases during the COVID19 pandemic showed that an increase of people surveyed wished to work remotely some or all of the time post crisis. The increase showed 83% in April 2020 which then increased to 94% in October 2020 of people wishing to work remotely some or all of the time after the pandemic [51]. This desire for a distributed work environment among workers is expected to drive new and interesting research in the area.

6 Conclusion

The literature reviewed in this paper covers a broad range of aspects involved in the GSE industry. The aim of the research was to detail the concerns faced by the organisations which conduct work in a distributed environment. These concerns ranged from nuanced areas relating to social and cultural issues to more practical issues such as agile processes in GSE context. The importance of training and education was also highlighted as a common trend detailed throughout the studies was issues caused by inexperience in GSE environments [22].

The research process of the team is detailed in the literature review methodology section of the paper. The methods used to search for the papers and inclusion/exclusion criteria was documented in order to show transparency of the research process. A table was conducted based on an excel spreadsheet used during the research stage to display the major themes and corresponding documents.

The importance of communication, coordination and control within an organization operating in a traditional software development environment was detailed [5]. However, in the context of GSE these concepts face ampilied challenges [4]. These challenges mainly related to distance; geographical, temporal and socio-cultural distances. The infrequencies of social interactions between members of the team, varying work patterns and interpretation issues relating to language were just some of challenges faced by GSE teams.

Agile processes in the GSE context faced several issues due the nature of the environment [24]. Scrum was identified as the most frequently used agile practice software development [25]. However, lean development and extreme programming were also highlighted as agile practices used in GSE environments also. These practices were all affected by the poor organization and communication caused by challenges detailed relating to geographical, temporal and socio-cultural factors.

Education and training are important to GSE as they provide the skill set to future/inexperienced software engineers to tackle the challenges faced in distributed environments. This paper investigates studies that highlight the importance of these factors.

7 Appendix

Concern	Frequency of Themes within Pools (%)
Geographical	22%
Temporal	13%
Language	13%
Culture	14%
Technology	1%
IT tools	1%
Agile	26%
Communication	26%
Teamwork	25%
Education	18%
Training	14%
Project management	16%
Project Lifecycle	0.5%
Requirement Engineering	13%
Trust	8%
Employee relationships	7%

Fig.1. Identified concerns and their percentage frequency of occurrence in the initial research pool using string "global software engineering"

Theme	Concerns
Communication	Linguistic, collaboration, coordination, teamwork, employee relationships
Geography and culture	Geographical, temporal, cultural, socio-culture
Agile development approach	Agile development, scrum
Training and Education	Education, training, skills, project management

Fig. 2. Themes to be used throughout this review and the concerns which are associated with them

Theme	Search Strings

Communication	“Global software engineering” “communication”, “trust”, “coordination”, “relationships”, “teams”
Geography and culture	“Global software engineering” “geography”, “geographical distance”, “temporal”, “temporal distance”, “culture”, “cultural”, “socio-culture”
Agile	“Global software engineering” “agile”, “agile software”, “scrum”, “lean development”, “kanban”, “extreme programming”
Training and Education	“Global software engineering” “skill”, “lack of skill”, “knowledge”, “training”, “staff training”, “experience”, “education”, “multi-university education”, “simulated games”

Fig. 3. Search strings used throughout the research for this review

8 References

1. Richardson, I., Casey, V., McCaffery, F., Burton, J., Beecham, S.: A Process Framework for Global Software Engineering Teams. Information and Software Technology 54(11), 1175-1191 (2012).
2. Vizcaíno, A., García, F., Piattini, M., Beecham, S.: A validated ontology for global software development. Computer Standards & Interfaces 46, 66-78 (2016)
3. Hsieh, Y,: Culture and Shared Understanding in Distributed Requirements Engineering. In: 2006 IEEE International Conference on Global Software Engineering (ICGSE'06), pp. 101-108, Florianopolis, Brazil (2006).
4. Ågerfalk, P., Fitzgerald, B., Holmström, H., Lings, B., Lundell, B., Ó'Conchúir, E.: A Framework For Considering Opportunities And Threats In Distributed Software Development. In: Proceedings of the International Workshop on Distributed Software Development, Paris, France (2005).
5. Curtis, B., Krasner, H., Iscoe, N..: A Field Study Of The Software Design Process For Large Systems. Communications of the ACM 31(11), 1268–1287 (1988).
6. Carmel, E., Agarwal, R.: Tactical approaches for alleviating distance in global software development. IEEE Software 18(2), 22-29 (2001).
7. Ó'Conchúir, E., Ågerfalk, P. J., Olsson, H. H., Fitzgerald.: Global software development: where are the benefits? Communications of the ACM 52(8), 127–131 (2009).
8. Khan, A. A., Keung J., Niazi, M., Hussain, S., Ahmad, A.: Systematic literature review and empirical investigation of barriers to process improvement in global software development: Client–vendor perspective. Information and Software Technology 87, 180-205 (2017).
9. Kotlarsky, J,. Oshri, I.: Social ties, knowledge sharing and successful collaboration in globally distributed system development projects. European Journal of Information Systems, 14(1), 37 (2005).
10. Child, J.: Trust - The fundamental bond in global collaboration. Organizational Dynamics, 29(4), 274–288 (2001).
11. Holmstrom, H., O'Conchuir, E., Agerfalk P. J., Fitzgerald, B.: Global Software Development Challenges: A Case Study on Temporal, Geographical and Socio-Cultural Distance. In: Proceedings of 2006 IEEE International Conference on Global Software Engineering (ICGSE'06), Florianopolis, Brazil, pp. 3-11 (2006).
12. Khan, A. A., Keung, J., Hussain, S., Bennin, K. E.: 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), pp. 159-168. Barcelona, Spain (2015).
13. Hoda, R., Babar, M. A., Shastri, Y., Yaqoob, H.: Socio-Cultural Challenges in Global Software Engineering Education. IEEE Transactions on Education 60(3), 173-182 (2017).
14. Beecham S., Noll J.: What Motivates Software Engineers Working in Global Software Development?. In: Proceedings of the 16th International Conference on Product-Focused Software Process Improvement, pp. 193–209. Bolzano, Italy (2015).
15. Inayat, I., Salim, S., Marczak, S., Daneva, M., Shamshirband, S.: A systematic literature review on agile requirements engineering practices and challenges. Computers in Human Behavior. 51, Part B, 915-929 (2015).
16. Yagüe, A., Garbajosa, J., Díaz, J., González, E.: An exploratory study in communication in Agile Global Software Development. Computer Standards & Interfaces. 48, 184-197 (2016).

17. Licorish, S., MacDonell, S.: Communication and personality profiles of global software developers. Information and Software Technology. 64, 113-131 (2015).
18. Ebert, C., Kuhrmann, M., Prikladnicki, R.: Global Software Engineering: An Industry Perspective. IEEE Software. 33, 1, 105-108 (2016).
19. Wang, Y., Zhang, M.: Country Stereotypes, Initial Trust, and Cooperation in Global Software Development Teams. Proceedings of the 2019 ACM/IEEE 14th International Conference on Global Software Engineering (ICGSE). (2019). Montreal, QC, Canada.
20. Yaseen, M., Baseer, S., Sherin, S.: Critical challenges for requirement implementation in context of global software development: A systematic literature review. Proceedings of the 2015 International Conference on Open Source Systems & Technologies (ICOSST). (2015). Lahore, Pakistan.
21. Iftikhar, A., Alam, M., Musa, S., Su'ud, M.: Trust Development in virtual teams to implement global software development (GSD): A structured approach to overcome communication barriers. Proceedings of the 2017 IEEE 3rd International Conference on Engineering Technologies and Social Sciences (ICETSS). (2017). Bangkok, Thailand.
22. Ebert, C., Kuhrmann, M., Prikladnicki, R.: Global Software Engineering: Evolution and Trends. Proceedings of the 2016 IEEE 11th International Conference on Global Software Engineering (ICGSE). (2016). Orange County, CA, USA.
23. Li, Y., Krusche, S., Lescher, C., Bruegge, B.: Teaching Global Software Engineering by Simulating a Global Project in the Classroom. Proceedings of the 47th ACM Technical Symposium on Computing Science Education. (2016). Memphis, Tennessee, USA.
24. Jalali, S., Wohlin, C.: Global software engineering and agile practices: a systematic review, Journal of Software: Evolution and Process, 24(6), 643-659 (2011).
25. Lous, P., Kuhrmann, M., Tell, P.: Is Scrum Fit for Global Software Engineering?. In: 2017 IEEE 12th International Conference on Global Software Engineering (ICGSE). pp. 1-10, Buenos Aires (2017).
26. Faniran, V., Badru, A., Ajayi, N.: Adopting Scrum as an Agile approach in distributed software development: A review of literature. 2017 1st International Conference on Next Generation Computing Applications (NextComp). pp. 36-40, Mauritius (2017).
27. Stray, V., Moe, N.: Understanding coordination in global software engineering: A mixed-methods study on the use of meetings and Slack. Journal of Systems and Software. 170, 110717 (2020).
28. Scrum: An Agile Process Reengineering in Software Engineering. International Journal of Innovative Technology and Exploring Engineering. 9(3), 840-848 (2020).
29. Arumugam, C., Vaidayanthan, S., Karuppuchamy, H.: Global Software Development: Key Performance Measures of Team in a SCRUM Based Agile Environment. In: Computational Science and Its Applications – ICCSA 2018. pp 672-682, Melbourne, VIC, Australia, (2018).
30. Gupta, P.: Applying Agile Lean to Global Software Development, (2017).
31. Tanner, M., Edgar Dauane, M.: The Use of Kanban to Alleviate Collaboration and Communication Challenges of Global Software Development. Issues in Informing Science and Information Technology. 14, 177-197 (2017).
32. Layman, L., Williams, L., Damian, D., Bures, H.: Essential communication practices for Extreme Programming in a global software development team. Information and Software Technology. 48(9), 781-794 (2006).
33. Shameem, M., Kumar, R., Nadeem, M., Khan, A.: Taxonomical classification of barriers for scaling agile methods in global software development environment using fuzzy analytic hierarchy process. Applied Soft Computing. 90, 106122 (2020).

34. Nisyak, A., Rizkiyah, K., Raharjo, T.: Human Related Challenges in Agile Software Development of Government Outsourcing Project. In: 2020 7th International Conference on Electrical Engineering, Computer Sciences and Informatics (EECSI), pp. 222-229, Yogyakarta, Indonesia (2020)
35. Márquez, R., Vizcaíno, A., García, F., Manjavacas, A.: GLOBAL-MANAGER: a serious game for providing training in project manager skills. In: Proceedings of the 15th International Conference on Global Software Engineering, pp. 127–131, Association for Computing Machinery, New York, NY, USA (2020)
36. Hoda, R., Babar, M., Shastri, Y., Yaqoob, H.: Socio-Cultural Challenges in Global Software Engineering Education. IEEE Transactions on Education. 60(3), 173-182 (2017).
37. Marinho, M., Luna, A., Beecham, S.: Global Software Development: Practices for Cultural Differences. Product-Focused Software Process Improvement. 299-317 (2018).
38. Jusoh, Y., Haizan Nor, R., Mahmood, B., Wafeeq, M., Ali, M., Baihaqi Jusoh, M.: Communication Management in Global Software Development Projects. In: 2018 Fourth International Conference on Information Retrieval and Knowledge Management (CAMP), pp. 1-7, Kota Kinabalu, Malaysia (2018)
39. Jain, R., Suman, U.: A Project Management Framework for Global Software Development. ACM SIGSOFT Software Engineering Notes. 43(1), 1-10 (2018).
40. Khan, A., Keung, J., Hussain, S., Niazi, M., Tamimy, M.: Understanding software process improvement in global software development. ACM SIGAPP Applied Computing Review. 17(2), 5-15 (2017).
41. M. Bass, B. Sarah, M. A. Razzak and J. Noll.: Employee Retention and Turnover in Global Software Development: Comparing In-House Offshoring and Offshore Outsourcing, In: 2018 IEEE/ACM 13th International Conference on Global Software Engineering (ICGSE), pp. 77-86, Gothenburg, Sweden (2018)
42. Ebert, C., Kuhrmann, M., Prikladnicki, R.: Global Software Engineering: Evolution and Trends. In: 2016 IEEE 11th International Conference on Global Software Engineering (ICGSE), pp. 144-153, Orange County, CA, USA (2016).
43. Cico, O., Jaccheri, L., Nguyen-Duc, A., Zhang, H.: Exploring the intersection between software industry and Software Engineering education - A systematic mapping of Software Engineering Trends. Journal of Systems and Software. 172, 110736 (2021).
44. Clear, T., Beecham, S.: Global Software Engineering Education Practice Continuum Special Issue of the ACM Transactions on Computing Education. ACM Transactions on Computing Education. 19(2), 1-8 (2019).
45. Beecham, S., Clear, T., Damian, D., Barr, J., Noll, J., Scacchi, W.: How Best to Teach Global Software Engineering? Educators Are Divided. IEEE Software. 34(1), 16-19 (2017).
46. Ouhbi, S., Pombo, N.: Software Engineering Education: Challenges and Perspectives. 2020 IEEE Global Engineering Education Conference (EDUCON). 202-209 (2020).
47. Beecham, S., Clear, T., Barr, J., Daniels, M., Oudshoorn, M., Noll, J.: Preparing Tomorrow's Software Engineers for Work in a Global Environment. IEEE Software. 34(1), 9-12 (2017).
48. Clear, T., Beecham, S., Barr, J., Daniels, M., Oudshoorn, M., Noll, J.: Developments in Global Software Engineering Education. 2016 IEEE Frontiers in Education Conference (FIE). 1-4 (2016).
49. Vizcaíno, A., García, F., Guzmán, I., Moraga, M.: Evaluating GSD-Aware: A Serious Game for Discovering Global Software Development Challenges. ACM Transactions on Computing Education. 19(2), 1-23 (2019).

50. Bosnić, I., Ciccozzi, F., Crnković, I., Čavrak, I., Nitto, E., Mirandola, R., Žagar, M.: Managing Diversity in Distributed Software Development Education—A Longitudinal Case Study. ACM Transactions on Computing Education. 19(2), 1-23 (2019).
51. McCarthy, A., Ahearne, A., Bohle-Carbonell, K., Ó'Síocháin, T., Frost, D.: Remote working during COVID-19: Ireland's national survey initial report (2020)
52. Clarke, P.: AgileOriginsRADManifestoNotes. DCU, Dublin, Ireland (2021).
53. Clear, T., Beecham, S., Barr, J., Daniels, M., McDermott, R., Oudshoorn, M., Savickaite, A., Noll, J.: Challenges and Recommendations for the Design and Conduct of Global Software Engineering Courses. Proceedings of the 2015 ITiCSE on Working Group Reports. (2015). Vilnius, Lithuania.
54. Niazi, M., Mahmood, S., Alshayeb, M., Riaz, M., Faisal, K., Cerpa, N., Khan, S., Richardson, I.: Challenges of project management in global software development: A client-vendor analysis. Information and Software Technology. 80, 1-19 (2016).