Challenges and Recommendations for the Design and Conduct of Global Software Engineering Courses: A Systematic Review

Tony Clear
School of Engineering, Computer & Mathematical Sciences
Auckland University of Technology
Private Bag 92006, Auckland, NZ
+64 9 921 9999
tony.clear@aut.ac.nz

Sarah Beecham
Lero – The Irish Software Research Centre,
University of Limerick
Ireland
+353 61 233769
sarah.beecham@lero.ie

John Barr
Ithaca College
Ithaca, New York,
USA
+ 1 607 274 3011
barr@ithaca.edu

Mats Daniels
Uppsala University
Box 337 751 05 Uppsala
+46 18 471 3167
matsd@it.uu.se

Roger McDermott
Dept. of Computer Science
Robert Gordon University
Aberdeen, UK
+44 1224 262000
roger.mcdermott@rgu.ac.uk

Michael Oudshoorn
Dept of Computer Science and Computer Networking
Wentworth Institute of Technology
Boston, USA
+1 617 989 4275
oudshoornm@wit.edu

Airina Savickaite
Dept. of Comp. Sci. and Engineering
Vilnius University
Lithuania
+1 (860) 486-5232
airina.savickaite@gmail.com

John Noll
Lero – The Irish Software Research Centre,
University of Limerick
Ireland
+353 61 233769
john.noll@lero.ie

ABSTRACT

Context: Global Software Engineering (GSE) has become the predominant form of software development for global companies and has given rise to a demand for students trained in GSE. In response, universities are developing courses and curricula around GSE and researchers have begun to disseminate studies of these new approaches.

Problem: GSE differs from most other computer science fields, however, in that practice is inseparable from theory. As a result, educators looking to create GSE courses face a daunting task: integrating global practice into the local classroom.

Aim: This study aims to ameliorate the very difficult task of teaching GSE by delineating the challenges and providing some recommendations for overcoming them.

Method: To meet our aims we pose two research questions ("When teaching GSE to students in Higher Education, what are the (a) challenges, and (b) recommendations for addressing them") and then conduct a systematic literature review (SLR) to determine the answers to these questions. Our SLR follows a carefully designed and validated protocol.

Results: We found 82 papers that addressed our research questions. Our findings indicate that in addition to the challenges posed by GSE in general, particular problems arise in educational situations. The majority of these challenges fall into the "global distance" category, though teamwork challenges and people issues (such as trust) also commonly arise. Organizational differences between institutions, differing skill sets between students in different locations, and varying cultural work norms, for example, all operate within educational settings in quite different ways than in professional development teams. Integrating cultural training, conducting teamwork exercises to build trust, and instructor monitoring of team communication are all examples of techniques that have been used successfully by educators according to our review

Conclusion: Despite the severity of the challenges in GSE education, many institutions have successfully developed courses and curricula targeting GSE. Indeed, for each of the challenges we have identified in the literature there are numerous recommendations for overcoming them. Instructors can use the recommendations given in this study as a starting point to running successful GSE courses.

ITICSE-WGR'15, July 04-08, 2015, Vilnius, Lithuania
DOI: http://dx.doi.org/10.1145/2858796.2858797

CCS Concepts

• Social and professional topics → Computing education **• Software and its engineering.**

Keywords

Global Software Engineering, International Collaboration, Open Ended Group Project, Capstone, Teaching and Learning, Global Software Development, Systematic Literature Review.

1. INTRODUCTION

In this working group we have examined the many issues facing Computer Science (CS) educators teaching CS courses involving global collaboration, and the options available to them in responding to the issues arising. The available literature proved to be voluminous and wide ranging, but was rather disparate in nature, reporting local experiences and often failed to build upon prior work.

In this paper we present recommendations from our comprehensive systematic literature review on Global Software Engineering (GSE) Education. Since Monasor *et al.*'s literature review in 2010 [47], no comprehensive review of the research has been published to provide a complete instructor-focussed picture of the available material on GSE education. While Monasor [47] presented the state of the art of GSE education and training in both university and industrial settings, there were no clear guidelines in terms of barriers and recommendations. Fortaleza [30], conducted a mapping study in 2012, which reviewed who was doing GSE Education; although identifying some 19 GSE courses, they noted a contradiction between the espoused need for such education and the reality, concluding there were "a small number of institutions that in fact teach it in their programs" [30]. By focussing more on practical guidelines and recommendations (see Section 6.1), we aim to help CS course instructors and interested researchers to determine the current state of research in GSE Education, and how they might use that research to inform their practice. Our systematic approach to analyzing published studies enables us to identify reliably where the literature has recurring themes, where it presents conflicting findings, and where are there gaps in the existing body of knowledge.

GSE is fast becoming standard practice [4; 30; 48; 57], and today's software engineering students are very likely to become tomorrow's global software engineer. Nearly ten years ago now, the report of the ACM Job Migration Task Force on the Globalization and Offshoring of Software was introduced with these statements by John White, then Chief Executive Officer of ACM:

…the field of computing and information technology has experienced a dramatic shift in the past five years to a truly global industry.

The forces that have driven and shaped this change are still at play and will continue.

The educational systems that underpin our profession will need to change. [4]

We consider the wider implications of preparing students for the complex world they are likely to enter, where they will not only be confronted with difficult technical problems, but also with how to work successfully in multi-site teams.

Global software engineering, or global software development (GSD), courses have increasingly been offered as a way to afford students authentic learning experiences [23; 43; 56] of global collaboration. Given the importance of educating students with the required skills for developing software in multi-site teams we look to the literature to answer two research questions (RQs):

RQ1: What are the challenges in delivering GSE courses to Software Engineering students?

RQ2: What are the recommendations for delivering GSE courses to Software Engineering students?

The aim is to produce a broad ranging resource for global software engineering educators, which will support efforts to design and conduct successful courses between globally dispersed institutions and student teams.

This paper is organised as follows: in the next section, we begin with a brief background on Global Software Engineering Education, which also outlines a rationale for the definition we have adopted when scoping this study. In Section Three we summarise our systematic method for conducting the review which involves following rules set out in our protocol. Section Four presents an overview of the surveyed literature, including geographical spread, temporal aspects, and publication details. Section Five reports the results of our synthesis of identified themes based on our two research questions. In Section Six we discuss our key findings, as well as some limitations of this study; finally, in Section Seven we present our conclusions.

2. BACKGROUND

2.1 Global Software Engineering Education

This systematic literature review is concerned with a crucial but neglected area of software engineer education and training: – *how to teach global software engineering methods to students before they enter the workplace?* There is increasing recognition that GSE requires special treatment, and that students entering the workplace are likely to find themselves working in distributed teams. In investigating the topic of GSE for this study, the available literature proved to be voluminous and wide ranging, but was rather disparate in nature, reporting many local experiences and often failing to develop upon prior work. As noted in our introduction, two reviews have been conducted on this topic that address aspects of the area [30; 47]. Yet no review of this scope, with the purpose of bringing together the combined knowledge on the topic with specific recommendations for GSE educators, has been found in the GSE education literature. Therefore, since much of the prior work has been descriptive in nature, our goal in this paper, in common with the call to offer 'solutions' in Ali-Babar and Lescher [1], is to provide some guidance for CS educators that not only identifies issues and pitfalls, but is of a more prescriptive and directly applicable nature for those planning and offering GSE courses.

Since GSE is increasingly cited as becoming the norm [4; 30; 48; 57], students studying SE are very likely to find themselves working in multi-site teams on graduation. Yet GSE projects often fail to realise hoped-for advantages such as higher productivity through hiring highly skilled engineers from countries with competitive labour rates [21]. The challenge of developing software across Global Distance (temporal, geographic and cultural), is complex. Many organisations are realising that they need to invest in cultural training to improve team collaboration [45; 46]. If educators of the future workforce can pre-empt this need, the new tranche of software engineers will be better equipped for the unique challenges imposed on them by working in multi-site teams.

The studies in this area suggest that conventional approaches to teaching SE are increasingly outdated and lack authenticity. For instance, as observed in Matthes [43]:

When considering the personal requirement today's software engineers are facing in their daily work life, it is surprising to see that teaching GSE at universities is still in its infancy.

The literature is presenting mixed messages. The balance between developing students with strong technical skills, and augmenting those with a broader set of professional capabilities, has long been a source of tension in the academy [43; 56]. Traditionally these challenges in computer science and software engineering programmes have been addressed through capstone courses and internships [18; 20].

However with the rise of globalisation and the concomitant changes in the working environment for professional software engineers [4], new approaches are needed, and a number of collaborative software engineering programmes have arisen in response [3; 4; 13; 24; 25; 28; 31; 61]. These initiatives have mostly been pioneering and relatively discrete, and have represented non-trivial commitments for the participating institutions. Some of the collaborations however have been long lived e.g. [19; 23; 28; 55]. One encouraging report has observed that *students in international teams can benefit from learning by osmosis and can perform as well as the students in local teams despite the extra effort usually required for GSE projects* [41].

Underlying the need for extra effort in courses of this nature are a number of issues which inevitably arise from the challenges of the distances posed by time, space, organisational, linguistic and cultural boundaries [15; 16; 19; 26; 27; 32]. Confusions, technology breakdowns [34], issues relating to trust development [38], collaboration readiness [52], cultural challenges [14; 16], student motivation [11; 56] and uncertainties in communication [22; 52], are all inevitably part of the experience. Consequently, the ability to manage ambiguity and uncertainty are key capabilities that students must develop if they are to have an education that endures [19; 24; 27; 45; 56]. Since we do not have all the answers for doing this well, it is therefore necessary to continue to develop models, practices and strategies that will serve both students and educators, as well as the profession. A starting point for capturing these methods is to identify key lessons from what has worked well in GSE teaching and learning as reported in the literature.

2.2 Defining GSE/GSD

In this working group we had to wrestle with scoping our study and its boundaries. We use the terms Global Software Engineering (GSE) and Global Software Development (GSD) interchangeably, but with education added as a modifier settled on the abbreviation “GSE-Ed” for this study. A working definition for GSD/GSE is given below:

In GSD, stakeholders from different national and organizational cultures and time zones are involved in developing software…and tasks at various stages of the software lifecycle may be separated and implemented at different geographic locations coordinated through the use of information and communication technologies…[36]

2.3 Defining GSE-Ed

GSE-Ed can be considered as an extended case of Software Engineering Education. The starting point therefore for a definition will be the definition of standard co-located models for SE Education. Yet as is obvious from the quotes below that is no simple task, as there has been argument over the definition of software engineering itself for decades, for instance.

There is no universally accepted definition of software engineering. For some, software engineering is just a glorified name for programming [44].

The fact that the literature contains many different definitions of software engineering implies that a concise and complete definition of software engineering is difficult to formulate [42, p.11].

Essentially therefore, software engineering practices are largely concerned with managing relevant processes and with design activities, and these can appear in a range of guises. Most of the activities involved in software development and evolution tend to use team-based processes that embody some form of design element…Each of these adds yet another layer of complication: teams must be organized with regard to aspects such as communication, coordination, and management and design activities are nondeterministic…processes that lead to solutions that are rarely right or wrong [42, p.12].

Therefore SE Education needs to address these aspects of theory and professional practice and the SE 2014 report on Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering [42] identifies three guiding principles:

The first is the desired outcomes for a student who has studied an undergraduate curriculum in software engineering. The second is a set of foundational ideas and beliefs about the nature and form of software engineering. The third concerns the goals for the curriculum guidelines [42, p.20].

GSE education amplifies the inherent challenges in SE education noted above in the SE2014 report. Team based processes in GSE now incorporate various forms of distance, which further add to the *yet another layer of complication* problem. So strategies to manage and limit these complexities become important, with theoretical courses and simulation approaches complementing full inter-institutional collaborations.

Given these definitional problems and inconsistencies we have settled on a working definition of GSE-Ed for the purposes of this paper:

> “GSE-Ed represents a combination of learning and teaching strategies that prepare students for GSE/GSD”

where GSD adopts the definition from Holmström [36] as stated in Section 2.2.

In a systematic literature review [17; 48] it is critical to be comprehensive, yet ensure that a manageable number of papers are retained for the analysis; therefore, defining terms is important. In this study we are interested in the differences in learning and teaching in globally distributed models. The research questions posed in the study then, are based upon this definition of GSE-Ed.

Similarities can also be noted with other forms of authentic educational experiences [cf.18]. GSE Education courses can be placed within a continuum model, with multiple dimensions. The model from [18] is repeated in Figure 1 to indicate the aspects for consideration in a co-located setting.

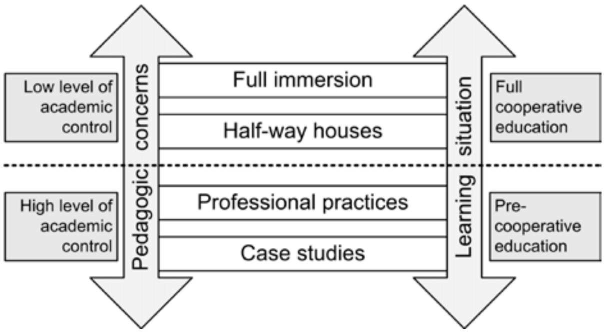

Fig. 1 The co-operative education continuum [18]

Helping to further focus this study, we started with the notions of distance common in the GSE literature [15; 16; 19; 26; 27; 32], and used those to determine whether a paper was in scope. We required that a focus on educational courses/initiatives addressing these dimensions was present in a study that would fit our inclusion/exclusion criteria. Figure 2 presents these distances, collapsed into three primary aspects, on the basis that *linguistic* and *institutional* distance could be viewed within a broader *cultural distance* dimension.

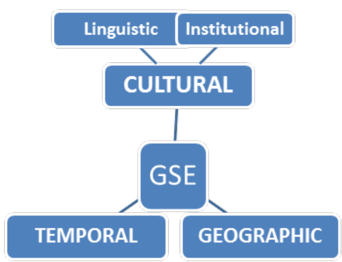

Fig. 2 Distances in GSE-Ed

These distinctions helped us frame the criteria by which to include or exclude studies as elaborated in the next section.

3. METHOD - Systematic Literature Review (SLR) Procedure

We used the SLR procedure defined by Kitchenham and Charters [39] to identify, evaluate and interpret the available published studies relating to our research questions.

The aim of this SLR is twofold: first, to identify the challenges that educators experience when teaching global software engineering, and second, to propose a set of recommendations that will facilitate and ease the teaching process. The population of interest comprises software engineering students and their instructors, and the topic of interest is teaching and learning in Global Software Engineering. We look for answers to our research questions through an investigation of primary studies found in selected sources.

In accordance with systematic review guidelines [39] we take the following steps:

1. Identify the need for a systematic literature review
2. Formulate review research question(s)
3. Source selection – record sources used to search for primary studies
4. Document selection: Classify data needed to answer the research question(s) (inclusion exclusion criteria).
5. Extract data from each included study (data extraction)
6. Summarise and synthesise study results (meta-analysis)
7. Assess and record the quality of included studies
8. Interpret results to determine their applicability (see discussion section of this paper) – we describe in this section how we validated our results.
9. Write-up study as a report (as evidenced in this paper)

These steps are detailed in our protocol [6], which is based on the process used by Beecham and colleagues [5]. We developed our protocol by piloting the process with three researchers who performed searches based on rules given in the protocol.

The remainder of this methodology section summarises the process presented in our protocol. Where more information is required please refer to Beecham *et al.* [6].

3.1 Identify the need for a review

The proposed systematic literature review is concerned with an important area in software engineer education and training: how to prepare students for global software engineering. While there is increasing recognition that GSE requires special treatment, and that students entering the workplace are likely to find themselves working in distributed teams, no deep review has been undertaken to bring together the combined knowledge on the topic.

3.2 Formulate review research questions

We considered whether our general research question, "What are the key issues in and approaches to designing and conducting GSE courses?" is suitable for investigation by systematic review. *Prima facie* this question does not closely match the type suggested by Kitchenham and Charters [39], where the emphasis is on assessing how technology is adopted in or affects software engineering. Our work perhaps relates more closely to the root of the guidelines provided by the medical literature. We can adapt a medical theme, "Assessing the economic value of an intervention or procedure", to "Assessing the [economic] value of applying recommended design approaches to global software engineering courses". In our case we can interpret "economic" in terms of a student's readiness to work in GSE.

Initial research shows very little work in the area of the economics of education in global software engineering. Therefore, to answer our key research question in terms of the value GSE courses bring to the student and the workplace we pose two sub-questions:

RQ1: What are the challenges in delivering GSE courses to Software Engineering students?

RQ2: What are the recommendations for delivering GSE courses to Software Engineering students?

We need to address both these questions as there may be barriers (RQ1) to implementing certain recommended practices (RQ2). Recommendations (RQ2) need to be in context with any known constraints (RQ1). The context of the education setting is Higher/Third tier level. The recipients of these courses can therefore be full time students (with no industrial experience), or software engineers (professionals) participating or collaborating in university based training.

3.3 Source selection

Papers were selected using two different methods. The principle technique was a keyword search in the major on-line digital libraries. Key words and synonyms were drawn up for each research question and then the following databases were searched using Boolean conditional key word search strings:

- IEEE Digital Library (www.computer.org)
- ACM Digital Library (http://portal.acm.org/dl.cfm)
- Scopus (http://www.elsevier.com/solutions/scopus)

The keywords used in each of these searches are provided in Appendix C, where different combinations of this string are applied:

((((software OR "information technology" OR "information system" OR comput* OR programming) AND (student OR trainee OR learner)) AND ("distributed software" OR "global software") AND (educat* OR train* OR course)))*

These initial searches resulted in 545, 57, and 160 papers respectively, with a total of 762 as shown in Table 1.

To ensure we did not overlook any important material, additional searches were performed directly on key conference proceedings producing a further 23 studies:

International Conference on Global Software Engineering (ICGSE)

International Conference on Innovation & Technology in Computer Science Education (ITiCSE)

Collaborative Teaching of Globally Distributed Software Development Workshop (CTGDSD)

These specific conferences were chosen because of their focus on either Global Software Development or Computer Science Education (or both). These direct searches involved an examination of the table of contents from the last ten years of ITiCSE and ICGSE, and the three years that the CTGDSD was held. Papers from these sources were selected according to our inclusion and exclusion criteria (see Section 3.4 below) and our two research questions (see Section 3.2 above). These additional searches resulted in 23 additional papers (see Table 1).

Table 1 shows the number of papers selected from our sources described in Section 3.3. Having removed duplicates across databases, we were left with 649 unique papers to consider including in our study. The table shows the several filtering phases used to establish our final set of 82 papers.

Table 1: Paper selection process

Selection Process	# papers	Validation process
Database papers found	762	Check for known papers
Duplicates removed (-136)	626	Agreement across researchers
Direct searches (+23)	649	n/a
Sift based on Title and Abstract (478 rejected)	171	All 649 papers assessed by 2 researchers
Full papers reviewed applying inclusion and exclusion criteria (63 rejected)	108	9 papers reviewed by 2 researchers to check agreement
Repeated studies identified (26 removed) to produce final set of 82 papers (excl criteria 2)	82	8 papers discussed by group to agree which paper to retain

3.4 Document selection

To be included in this study a paper had to meet the inclusion criteria and not fall into the exclusion criteria. The criteria used to scope the study are given in Tables 2 and 3:

Table 2: Paper Inclusion Criteria

1. must address global software development/engineering (GSD/GSE), defined as collaboration across various Global Distances: cultural/linguistic, temporal, geographic.
2. must be either a theoretical study or an empirical study (where we define 'empirical' as in Hirschheim [35] *(i.e., cases, [including experience reports] multiple observations as in surveys, statistical samples, anything coming from the five senses)*
3. must have been published in the years 2000-date; (publications found in a secondary searches can be any date).
4. must be peer reviewed.
5. must directly answer one or more of our RQs.
6. must be a primary study (reporting ideas or experiences directly).

Table 3: Paper Exclusion Criteria

1. books, presentations, opinion pieces, posters, very short papers (less than 2 pages), or proposals.
2. repeated studies that did not provide significant new insight.
3. studies focused primarily on open source development rather than global software engineering
4. complete proceedings, not individual papers.
5. where the focus is primarily on E-learning, remote learning, or cloud computing as opposed to GSD/GSE.
6. where the primary concern is on hardware/distributed systems (where distributed relates to the system, rather than team).
7. where the focus is on collaborative software development which is not *globally* dispersed.
8. where there is no focus on (at least) parts of the life cycle development process across collaborative groups/parties.
9. SLRs or tertiary studies. These would reflect previously discovered insights, duplicate findings in our primary studies.

3.5 Data extraction

The 171 full papers (accepted based on reading the title and abstract – see Table 1) were divided between the authors for further review. Each researcher extracted data from their set of papers according to:

Our Research Questions
Exclusion Criteria
Inclusion Criteria
Quality Criteria (Valentine's taxonomy Appx B)

Data was extracted in two phases. Phase 1 required the researcher to provide context information and assess whether the study met our inclusion and exclusion criteria (as a result of this process we excluded a further 63 studies). The remaining 108 studies were checked for repetition (where similar study results are reported across different publications), resulting in a further 26 studies being eliminated. The final set of 82 studies were closely examined and recorded in Phase 2 (qualitative analysis) where text snippets that addressed our research questions were extracted directly into our data extraction form (see Appendix F all categories included in the data extraction form). The form also prompted the researcher to record any themes that emerged as part of the data extraction process for further analysis as explained in the next section. *For practical purposes all results, including quality assessment are combined into one document/excel spreadsheet.*

3.6 Summarise and synthesise study results

We synthesised our text snippets into themes using content analysis, a qualitative analysis technique often used to analyze unstructured text, such as focus group data. The process of data analysis as described by Krippendorff [40] is similar to the grounded theory method, where replicable and valid inferences are made from the data to their context. Where content analysis differs from grounded theory is that it is largely numeric and therefore includes a quantitative form of research. Content analysis produces results such as, "46% of challenges recorded in the GSE Education Literature relate to Global Distance". Content analysis involves assigning a type or code to excerpts of text, which captures or classifies the meaning of the excerpt. Traditionally, qualitative analysis is performed on large documents such as interview transcripts; individual sentences or fragments within the larger document are coded according to their meaning or intent. In the context of our literature review analysis these "documents" are the published studies, and the fragments are short text extracts from papers that are deemed to answer our research questions.

Consequently, each artefact can be considered as a whole, with a type assigned to convey the meaning of the entire artifact. Using content analysis, a researcher can assign a type code to each text snippet, classifying it, for example, as a Teamwork issue, or a Global Distance issue, where the resulting coded sample set gives a picture of the types of issues instructors experience when conducting courses in GSE.

3.6.1 Developing a Coding Scheme

Content analysis aims to identify the meaning of text by assigning a code that conveys that meaning. Coding allows researchers to ask quantitative questions about qualitative data, such as, how often do the studies on Education in GSE mention issues commonly associated with Global Distance as opposed to technical issues associated with how to learn to program? As such, it is essential that the coding scheme used to convey meaning is accurate. Also, it must be repeatable: different researchers should assign the same code to a given text fragment, and the same researcher should assign the same code to a given fragment when analyzed a week or a month later.

A good coding scheme is not only accurate and repeatable; if the number of codes is small, and their definitions are clear, the coding process becomes straightforward and can be completed easily and quickly. Our coding method is adapted from Noll [49-51] and comprises the following steps:

1) Create Initial Type Set: The first step is to select a representative sample of text fragments, and from these, create an initial set of codes that capture their meanings.

This is an inductive approach, in which the researcher reads a fragment and invents a code (word or phrase) that captures the meaning conveyed by the fragment. The list of codes grows and evolves as more fragments are read, and in the end may have many codes.

In this study, an initial set of codes was derived from a trial examination of several hundred text snippets extracted from 30 reviewed papers. These snippets were divided among six researchers, who individually created major categories and minor codes to describe their snippets.

This initial code set, shown in Appendix D, (Table D1), attempted to capture the wide variety of meanings, and comprised a total of 110 minor codes that were grouped in 18 major categories that reflected both research questions.

2) Aggregate into Type Categories: Next, the list of codes is examined to discover broader categories. Codes with similar meaning are grouped together, and coalesced into a single category. The goal is to refine the list into a handful of categories with distinct meanings, so that it is easy to decide to which category a given text fragment belongs. The categories are given names which become the codes that are assigned to text fragments.

The six researchers met as a group to compare their individual codes, and agree on an aggregated set of major categories and minor codes. In this way, the initial set of 18 major codes was reduced to seven categories, shown in Table 4, that capture meaning appropriate to the research questions for this study.

For example, the initial types 'language differences', 'cultural differences', 'time', and 'geographic distance' went into the single category 'global'; the minor codes under this major category capture the differences among types of Global Distance. Where issues or recommendations spanned across all Global Distances, we used the catch-all 'increased complexity' minor code.

Table 4: Final set of 7 codes and associated classifications

1. GLOBAL DISTANCE Increased complexity Cultural Temporal Linguistic General Organisational Skills	4. INFRASTRUCTURE Tools Technical issues Version Control
2.STAKEHOLDER/ ROLE Client Instructor Student University representative Role conflict	5. DEVELOPMENT PROCESS S/w Development Process Requirements Design Coding Testing System/code integration
3.PEOPLE/SOFT ISSUES Motivation Trust Stress Self-awareness	6. CURRICULUM/PEDAGOGY Course design Learning Outcomes 7. TEAMWORK/TEAM CREATION Synergy Task allocation

The results section discusses the frequencies of these codes, and summarises the findings according to each major category. The authors of this paper used the fully defined version of these codes as presented in Appendix E (Table E1) to categorise findings from their data extractions.

3) Create Checklist: Creating the final set of categories and codes was an iterative process. The final set of codes continued to be refined throughout the data extraction and coding process. It is in this way that new codes evolved, where there were gaps in the initial coding scheme.

3.6.2 Validating Codings

Once text fragments from all accepted papers were extracted and coded, the codings were subjected to a validation process to ensure consistency among different researchers. This was achieved with a four step process:

1) Extract text fragments from the coding forms. We used a form (see Appendix F) to collect data from each paper; the form included fields for text fragments and codings. These text fragment fields were extracted into a data file for processing.

2) Normalize codings using final checklist. The initial set of text fragments used to develop the final checklist was automatically recoded by mapping original major and minor categories and codings to final categories and codings from the checklist.

3) Validate codings. Once each text fragment had a major category and minor code from the final checklist, each fragment was reviewed by two researchers to ensure the correct category and code were assigned. A total of 806 text fragments were divided into four groups of 403 fragments (first half, second half, even, and odd). Each of four researchers then examined each fragment in one of these groups to validate the assigned major category and minor code.

4) Resolve conflicts. Where two researchers disagreed on a category and/or code assignment, the differences were discussed among the group via email, and a final major category and minor code agreed.

3.7 Study Methodology Quality Assessment

Many of the studies we reviewed were descriptive, and case specific. In keeping with our observations, Valentine [63] identified that CS education publications often do not fit the standard research quality benchmarks. We did not therefore attempt to assess the quality of the studies in terms of sample sizes, sampling, response rates, questionnaire design, etc. However, as discussed in the results section, we applied a scheme developed specifically for CS Education according to Valentine's taxonomy [63].

First, each paper was classified as "Experimental", "Marco Polo", "Philosophy", "Tools", "Nifty", or "John Henry"; these categories are described in detail in Appendix B. Then, for studies classified as "Experimental", additional data about the context of the study (geographical area(s) involved, total number of sites, methodology, and analytical technique) were captured. Finally, for those studies classified as "Experimental" an assessment of the strength of the findings was made, based on the technique(s) used; this assessment ranged from "anecdotal" for studies that were essentially experience reports, to "valid" for studies that followed an explicit methodology, to "strong" for studies that presented statistical evidence to support the findings.

3.8 Validation

3.8.1 Validation 1 - Paper Selection based on Title and Abstract.

Our paper selection followed a repeatable, auditable and reliable process as outlined in our protocol [6]. The initial list of papers was derived from several sources (see Section 3.3). After eliminating papers that were duplicated across sources, 649 primary papers were identified as potential sources for this study (see Table 1).

Three authors performed the initial screening of this list of papers in three stages. The aim was to only include those papers that met our inclusion/exclusion criteria detailed in Section 3.4.

Stage 1: Three authors assessed the first 100 papers in our extracted papers list and by reading the title and abstract classified them according to the following scheme: "accept", "reject", "background", or "don't know".

Stage 2: Any disagreements between the three authors were resolved through discussion, and the inclusion/exclusion criteria modified in the protocol accordingly. For example, the filtering criteria was scoped to exclude papers that focused on e-learning or distance-learning where there was not clear evidence of global software development.

Stage 3: The remaining 549 papers were examined by the same three authors, who applied the refined criteria to classify each paper by the same four designations. All designations were verified by the other two authors and discrepancies were resolved through discussion.

This first filtering based on abstract and title, resulted in 171 accepted papers, to go to the next phase of analysis which was to read the full paper and complete data extraction forms.

3.8.2 Validation 2 - Paper selection (Full Paper)

A generic data extraction form was developed to record the context of the paper, and how each paper addressed our research questions. As a test of utility, three authors then independently used the form to extract and record data from nine randomly chosen papers; each of these papers was reviewed by two authors. The resulting 18 data extraction forms were compared and discussed, and the form was modified to better reflect the information required for the SLR. An example of this form is given in Appendix F.

The first part of the form addresses inclusion and exclusion criteria. After reading the full text, we rejected 62 more papers, as they failed to meet our inclusion/exclusion criteria. We also updated the protocol to reflect issues in the criteria. For example we realized that we should exclude secondary reviews from our study, since we had included most of the primary papers examined by these reviews; as such, the inclusion of these review papers would have resulted in duplicate findings.

3.8.3 Validation 3 -Data synthesis

In order to test our synthesised codes (method described in Section 3.6) four authors independently examined the mapping of text fragments for each selected paper to the codes, as described above in Section 3.6.2. In the end, every text snippet was thus coded or validated by at least two researchers.

4. RESULTS

4.1 Overview of Selected Studies

This section provides a brief overview of the studies profiling sources and dates, study methods and themes.

4.2 Study Sources

Table 5 gives a breakdown of where our 82 papers have been published. Papers were derived from a variety of sources, many of them high quality SE and CS-Ed conferences, (and some miscellaneous in the 'other' category). Not unsurprisingly the GSE focused venues provided the most papers for our study.

Table 5. Sources of Selected Papers

Conference/Workshop/Journal	# Papers
Int'l Workshop on Collaborative Teaching of Globally Distributed Software Development (CTGDSD)	13
Int'l Conf. on Global Software Engineering (ICGSE)	13
Int'l Conference on Software Engineering (ICSE)	10
Frontiers in Education Conference	6
Journal Articles	6
IEEE Conf. on SW Eng'ring Edu. & Training (CSEE&T),	5
Annual SIGCSE conference on Innovation and technology in computer science education (ITiCSE)	4
Int'l Conf. on Information Technology Based Higher Education and Training (ITHET)	3
Other: (conferences and workshops that occur twice or less)	22
Total	**82**

4.3 Study Methods

Here we categorise the types of studies that we have included in our selected papers. As these studies have been conducted with an educational focus we apply a taxonomy developed by Valentine and applicable to CS education publications [63]. The taxonomy consists of six categories, with the *experimental* category representing studies with some form of research rigour, through to *Marco Polo* descriptive studies as experience reports of one-off course iterations. The other categories are relatively self-explanatory, but are elaborated more fully in Appendix B.

Table 6. Profile of Selected Papers using Valentine Taxonomy

Category	# Papers	%
Experimental	42	51
John Henry	0	0
Marco Polo	24	29
Nifty	1	1
Philosophy	6	7
Tools	9	11
Totals	**82**	**100**

As can be seen from the classifications, approximately half of the studies apply some form of rigour in research design, which helps support the quality of our analysis, but nearly a third consist of local experience reports. A smaller number of papers adopt a more philosophical stance, and the remainder focus on tools to support GSE-Ed.

The more general classification, in Table 7 below, presents an alternative view. (Although classified as experimental within Valentine's classification, we excluded the literature reviews from the full analysis to avoid inflating our challenges and recommendations). However they have provided useful background information for the study.

Table 7. Profile of Selected Papers by Research Type

Category	# Papers	%
Empirical Research	43	52.5
Experience Report	34	41.5
Theoretical	5	6
Totals	**82**	**100**

As can be seen from the classifications, the papers were relatively evenly divided between empirical research studies, (studies applying quantitative and qualitative methods), and experience reports, which had limited research rigour. Research approaches and methods used covered a broad range from action research, descriptive and exploratory case studies, controlled experiment, student and instructor surveys, questionnaires and interviews, grounded theory, content analysis, log usage data analysis, statistical analysis. So the field can be considered both diverse and open in its choice of methods. A small number of theoretical papers were noted which presented frameworks or philosophical perspectives on aspects of GSE-Ed.

4.4 Geographical distribution of papers

The Figure 3 bar chart groups countries represented in our 82 studies by frequency of citation in the selected papers. In all we see that a rich selection of 39 countries is represented, which span the globe from east to west and north to south.

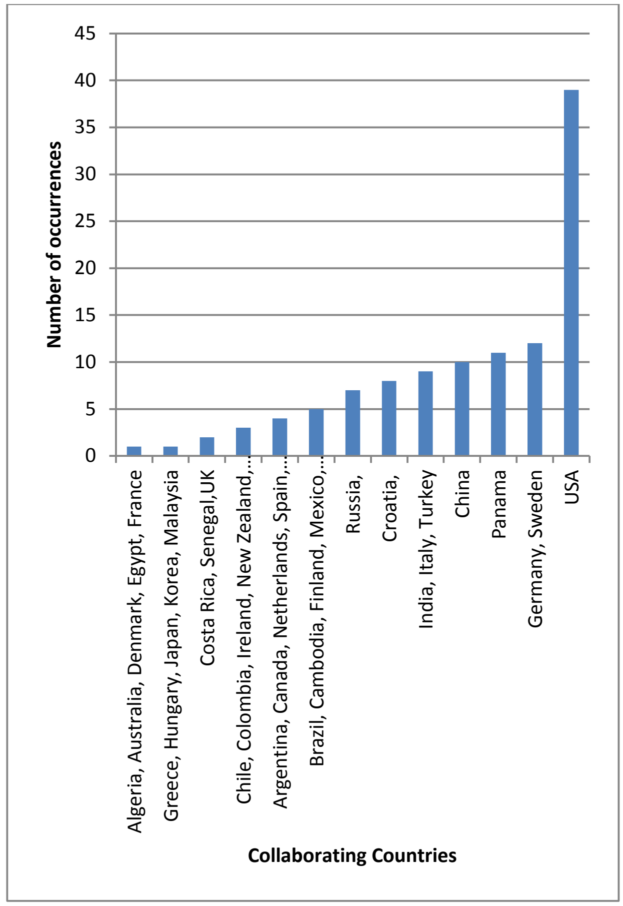

Fig 3: Countries represented in the empirical studies

The frequency groupings in Figure 3 indicate those more active countries from this study, with USA dominant overall, followed by Germany and Sweden as GSE sourcing countries, while the commonly regarded GSE providing countries of China and India are in a second grouping; Panama is represented perhaps as a nearshoring option. The subsequent groupings appear relatively mixed, representing a variety of collaborations between institutions across those countries. However, as can be seen from frequencies in the bar-chart, the studies covered in our review have a western /USA slant, given that they appear in the majority of our studies.

4.5 Study Dates

Figure 4 presents frequencies according to the year in which our 82 studies were published.

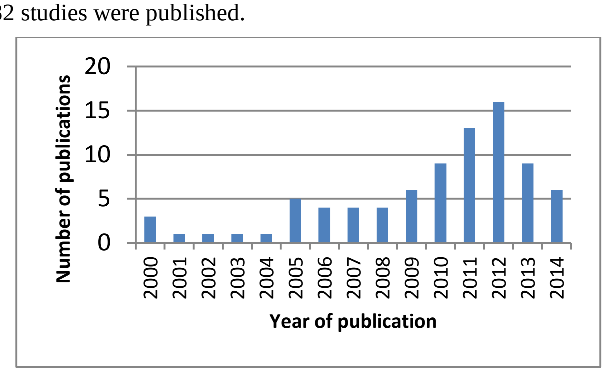

Fig 4: Number of papers included in review over time (no.82)

Table 8. Recommendations and Challenges Major Theme Breakdown

	Recommendations			Challenges			Total	
Theme	**Freq.**	**%**	**Rank**	**Freq**	**%**	**Rank**	**Freq**	**Freq %**
Global Distance	60	16%	4	189	46%	1	249	32%
Teamwork	80	22%	1	55	13%	2	135	17%
Curriculum/Pedagogy	71	19%	2	32	8%	6	103	13%
Stakeholder/Role	62	17%	3	36	9%	4	98	13%
Infrastructure	49	13%	5	33	8%	5	82	11%
People/Soft issues	28	8%	6	46	11%	3	74	10%
Development Process	16	4%	7	18	4%	7	34	4%
Totals	**366**			**409**			**775**	**100%**

As can be seen from Figure 4, there have been a fairly steady flow of papers over the last 15 years, from a slow start in the early 2000's and a picking up from 2005 to a peak in 2011 and 2012. While there appears to be a slight drop-off, papers are continuing to be published in the area. Although we conducted our searches to include papers published in the first half of 2015, no papers were identified that year. Many of the key conferences had not been held at mid June 2015, and there is often a lag until database publishing. We do however include commentary in the background and analysis sections (e.g. 5.3.2) based upon two specific ICGSE 2015 papers [7; 56], which had been published at time of writing and were relevant and known to the authors. Certainly GSE-Ed remains a rich area for enquiry and solutions are still being sought, as institutions grapple with preparing students for working globally, so we would expect GSE-Ed publications to continue for some time.

5. SUMMARY AND BREAKDOWN OF STUDY THEMES

The themes and subthemes identified in Section 3.6 above occurred with the frequencies shown in Table 8.

While Global Distance was a dominant category overall, in this as in other categories, the focus differed between challenges and recommendations. For the **challenges themes**, *global distance* was the dominant category, found in some 46% of the papers, with *teamwork* and a set of *people*/*soft issues* related concerns ranking next in approximately one third of the papers. Next were *Stakeholder roles*, *infrastructure* and *curriculum/pedagogy* in a grouping of similar weighting, with *development process* issues following at seventh place.

For the **recommendations themes**, the focus shifted to *teamwork* and *curriculum/pedagogy* as the areas into which interventions most often fell, with modifying *stakeholder roles* and addressing *global distance* not far behind, being found in approximately one third of the papers. Addressing issues to do with *infrastructure* ranked next with *people/soft issues* and *development process* having a lesser occurrence.

The following subsections present the results of our analysis, grouped by theme. We first present the theme's challenges (our RQ1) in order of frequency. Each challenge is followed by the recommendations for solving the issues (RQ2). Table 8 provides a full list of our themes along with their frequency and ranking.

5.1 Global Distance

Our Global Distance theme encompasses cultural, temporal, linguistic, geographic, and organisational distances. This distance tends to be expressed in terms of communication overhead due to increased complexity, and scaling of processes. For example tacit knowledge can remain hidden when working remotely, and temporal distances can lead to time delays, especially when distributed teams are sharing modules with high dependencies. In an educational setting, Global Distance extends to differences in institutional regulations (concerning synchronisation of semesters, assessments schemes, expectations and goals), and also differences in student skill levels. The following sub-sections discuss these distances more fully along with example quotes from related studies. In keeping with our research questions, we divide up our section in terms of GSE teaching and learning challenges (RQ1) – Section 5.1.1; and recommendations (RQ2) in section 5.1.2. Section 5.1.3 summarises the findings for the Global Distance theme.

5.1.1 Global Distance Challenges

Instructors reporting experiences of conducting GSE courses find Global Distance issues the most problematic, accounting for 46% of all our major identified themes (see Table 8). Looking at this large category in more detail we find that temporal issues (that encompass time related issues such as lack of time overlap, and delays), are the most prominent of all Global Distance challenges identified in our literature review on GSE Teaching and Learning. On the other hand, Table 9 shows that the recommendations for reducing the effects of Global Distance are most often found in the form of Organisational and institutional practices (accounting for 39% of Global Distance recommendations), where for example an effort should be made to synchronise course length, and start and finish times. The following 7 sections give an overview of the main challenges associated with Global Distance.

5.1.1.1 Cultural Distance Challenges

Cultural distance is defined in terms of differences in student communication styles (formal/informal; direct/indirect speech; deference to hierarchy, not being able to say 'no'), cultural norms, Ethnic and religious beliefs, and treatment of gender.

Table 9: Global Distance: Frequencies of Themes Identified in 82 Studies of GSE-Ed

Theme	Challenges			Recommendations		
	Freq	%	# Papers	Freq	%	# Papers
Cultural	39	21	27	6	10	6
Geographic	9	5	9	4	5	4
Increased complexity	13	7	12	6	10	6
Linguistic	22	12	22	6	10	6
Organisational and institutional	46	24	30	23	39	16
Skills (of student)	11	6	11	3	5	3
Temporal	49	26	35	12	20	9
Total	**189**	**100**	**N/A**	**60**	**100**	**N/A**

Cultural challenges in GSE education spanned from differences in 'student' culture [#76][1], work culture and work ethic [#38, #40], formality at work [#5], treatment of women [#23], adherence to rules [#23], to reactions to criticism [#23].

Cultural difference was a recurring theme across several studies [#28, #38, #54, #61]. These differences can be observed in styles of interaction [#33, #44] where mixed cultural backgrounds *caused problems with the interactions with extended team members* [#37].

Culture can have an effect on performance, where students with a strong hierarchical culture performed more poorly in collaborative projects [#75]. Non-reporting of project task overruns [#69], not giving an opinion to avoid confrontation [#19], and a reluctance to ask questions [#68] can all have an adverse effect on the project and are issues to be aware of in mixed cultural groups.

5.1.1.2 Geographic Distance Challenges

We view this minor theme as the impact geographic distance has on communication; where members of the team, for example, can no longer rely on informal type of communication to discuss issues and share knowledge.

Since remote team members are no longer visible, they may get forgotten or left out [#19], also it is difficult to interpret silence in a conversation when not meeting someone face to face [#82].

5.1.1.3 Challenges due to Increased Complexity

This theme was a catch-all in terms of covering issues that cut across several Global Distance minor themes, which we define as 'increased complexity' due to running large-scale projects that are distributed.

Some observations in this category were quite high level, and just noted that in GSE education communication processes can be challenging [#18, #13, #4] or need more communication than co-located student project groups [#49]. One group of researchers note that the preoccupation with communication and cultural issues limits the student's ability to explore other SE challenges [#60] perhaps underlining the finding that video-conferencing alone does not work [#11].

[1] Note: we prefix with # references from the secondary reference list of selected studies in Appendix A.

5.1.1.4 Linguistic Distance Challenges

Several studies reported issues relating to language, for example where students are forced to communicate in their second language.

Language was often viewed as a barrier to communication [#1, #12, #13, #40]; some students had problems in expressing themselves in English [#5], and in extreme cases a lack of language skills led to isolation [#7], also poor language skills made it more difficult to share a sense of humour [#82]. Language can also be a problem in development, for example when a group of Turkish students were asked to develop Graphical User-interfaces in Spanish [#72]. Often the Lingua Franca is English which in one case was the third common language across all the teams and introduced an additional translation filter in both directions [#23].

5.1.1.5 Organisational Distance Challenges

This minor theme caused the course organisers a great deal of anguish – and is perhaps the most difficult to control, with some advice given to select collaborating institutions with care given that many of these issues are beyond the control of any one location or institution. This section concerns problems with synchronisation or with infrastructures across collaborating institutions. For example, development process and communication tool mismatch, mismatch in course length, content, semester start and end dates, and different approaches to assessment, also differences in governance and management styles.

Management of the course has considerable overheads, and includes the additional coordination work of hosting institutions [#52, #55, #56]. Different school schedules is a problem [#51, #4, #13], along with conflict resolution [#76], dealing with continuous changes of rules at each site [#4], differing university regulations [#76], different views and expectations of teaching staff [#4], differences in admission requirements across institutions [#4], differences in examination requirements [#4], and incompatible holidays [#9, #23].

Bosnić *et al.* [#7] conducted many research studies in the field of GSE Education, and noted: *it can be a very challenging task for the project supervisor to monitor remote team and, at the end of the project work, evaluate their overall performance and individual contributions*, and GSE courses *require additional effort in technical preparation, involvement with students and interaction with host institutions* [#7].

5.1.1.6 Skills Imbalance Challenges

This set of observations refers to the distance between technical skill levels of participating student groups. It links closely to the Organisational challenges above. Here the focus is on the problem from the instructor's perspective in terms of trying to create a balanced team [#17], and managing different skill levels [#4], where in one extreme case students joined the distributed course with no training in technologies, tools or architectures [#67].

5.1.1.7 Temporal Distance Challenges

This sub-set of challenges proved the most prominent in the Global Distance theme. Forty-nine text fragments were identified in our 82 papers that dealt with some issues around time (see Table 9). We take a broad interpretation of temporal challenge, where any issue relating to time zone differences, time delays, issues with synchronous communication are included here, including time pressure due to over stretched team members.

Time-zone differences between locations made it difficult to schedule meetings [#1, #53, #67] and coordinate activities [#38, #40, #76, #78,] where meeting times were arranged that did not suit everyone [#68]. Time management is also a concern [#74, #4]. These difficulties point to the student project setting reflecting the kind of issues global teams in industry face daily; which is what the instructors aim to expose.

Time is important to communication, and has been studied as a specific line of research, in for example Swigger's 2012 study on the temporal communication behaviors of global software development student teams [#78]. Swigger *et al.* note: *communication behaviour seem to be linked to a team's timing, pacing, synchronization and, ultimately, performance. This tends to support the premise that global software teams, similar to teams everywhere, are temporally patterned in complex ways* [#78]. This pattern is perhaps explained by Swigger in an earlier study in 2009: *students' temporal patterns during projects also varies, with lots of communications at the beginning of each project and again soon after the mid-point of the project, and finally at the end of the project* [#80].

5.1.2 Global Distance Recommendations

For definitions of each of the seven minor themes in this category please refer to the challenges section above and Appendix E. The recommendations in this section may not apply in every case – they are taken from a range of papers with different contexts (see Section 4), however the interested reader can gain some useful insights into how other educators have overcome some of the challenges mentioned in the previous section. There is a stark contrast as well in terms of the number of recommendations (60) compared to the number of identified challenges (189) in Global Distance (see Table 8).

5.1.2.1 Reducing Cultural Distance

Understanding of different cultures can be taught in separate short courses [#27], and some say must be taught in terms of norms, beliefs and business ethos [#71]. Culture can be taught using practical or theoretical teaching methods [#54]. A tip given by Lago [#45] is to identify the cultural and educational differences between the students in the separate locations, and have students learn from each other with respect to culture [#36].

5.1.2.2 Reducing Geographic Distance

According to Richardson, *an initial face to face meeting with the central team proved to be extremely important* [#68]. Updating and revising the associated documentation is especially important when team-members are not all collocated [#2]. However, there was very little in the literature that directly addressed geographic distance issues. Geographic distance issues are perhaps helped by recommendations made in other major themes, such as infrastructure.

5.1.2.3 Reducing Increased Complexity due to Global Distance

Course leaders should encourage informal forms of communication [#64]. This can be achieved through a mix of communications styles (text based methods for complex details, and video for getting to know team) [#13]. Also communication should be frequent – and kept consistently high [#19, #64]. Communication protocols and strategies should be taught to the students [#71].

5.1.2.4 Reducing Linguistic Distance

Conversational English should be taught [#71], and some recommend that, in order to participate in the project, students must pass an English language test [#63]. Also given the different language skills in the teams, students should be given the opportunity to communicate through text; *email evened out the differences in English language abilities between teams, it was completely text based and allowed everyone to focus on what was being communicated* [#13].

5.1.2.5 Reducing Organisational/Institutional Distance

These recommendations are based around harmonizing processes across institutions and establishing a clear line of responsibility. For example, Favela [#28], recommend establishing responsibilities and power, whereas Clear suggests that project leaders agree to a course specific set of terms [#16], yet Bosnic [#4] state that a key success factor is flexibility in accepting different rules and habits. Students should be selected by the instructor based on a student profile to ensure a balanced team [#4], which requires a reliable and consistent student profiling process. The selection of the collaborating partner institution is critical; look for collaborators that are patient, reliable and supportive [#17], and ideally have evidence of pre-existing strong relationships [#35].

5.1.2.6 Redressing Skills Imbalance Between Locations

Support for this distance was sparse, other than ensuring that there is a clear understanding of knowledge prerequisites [#21], and perhaps as a warning Gotel notes, *it would be better if [students] had a software engineering class first to learn the skills and then be able to apply them in a global context* [#38]. Another way to close the skills gap is to *identify … educational differences between students … and then exploit those differences through knowledge transfer in the delivery of the course* [#45].

5.1.2.7 Reducing Temporal Distance

Temporal distance is eased by team members being flexible about meeting times [#48], and finding a common time for weekly development that all teams can participate in to keep track of progress and problems [#13]. Schedules should be shared to include working hours of the team [#35]. A mandatory project communication plan should be drawn up [#62], and teammates should be told if the student cannot attend a meeting or will be unable to answer emails [#48]. A mix of synchronous and asynchronous communication methods [#35] that are organised at regular intervals [#72] helps the team to keep in touch when not working in the same timezone. Communication might need to be started by brute force [#19], and text based communication may be the only option available at certain times [#13].

5.1.3 Global Distance Summary

Global Distance challenges and recommendations found in GSE educational settings are reflecting similar issues to those reported in the general GSE literature, where both observe issues relating to cultural, geographic, linguistic, and temporal distances [48; 58].

Table 8 shows the number of challenges observed in GSE Distance far outweighs the recommendations (with 189 observations made in challenges, and 60 recommendations). However, when considering recommendations made in our other themes, for example 'infrastructure' in section 5.5, these will have a positive impact on Global Distance issues. Therefore we advise anyone looking to conduct GSE courses to look across all our themes to gain a full and balanced picture of what is required.

5.2 Teamwork

Teamwork is an essential component of software engineering in general and it is not surprising that it is also a central problem in global software engineering, ranking as the second most addressed challenge in this study (see Table 10).

5.2.1 Teamwork Challenges

Teamwork in a global setting is certainly affected by issues such as culture and temporal differences that are also addressed in the *global distance* category. However, the teamwork category reflects the unique challenges imposed on students working together on a global project in an educational setting.

Table 10. Teamwork Challenges

Theme/Subtheme	Count	# Papers
Teamwork	**55**	**37**
Synergy	36	27
Task allocation	19	17

There are two minor categories designed to tease out the particular global problems in teamwork, 'synergy' and 'task allocation'. *Synergy* refers to issues affecting the cohesion, integration, and cooperation of global teams while *task allocation* includes the challenges of balancing skills, responsibilities, authority, accountability and management amongst a team. Project management issues in particular are included in the task allocation category.

5.2.1.1 Teamwork & Synergy Challenges

Comments in the *synergy* category reflected the difficulty student teams had forming effective working relationships with distant teams. Papers commenting on this issue indicate that a large amount of effort is required to engage distant teams and that local and distant team members sometimes do not commit sufficiently to a project or have different goals. These challenges often serve to limit engagement within and between local and distant teams and, as a result, reduce trust and cooperation between local and distant teams. Studies found, for example:

students working in the international teams were expected to put extra effort because of the communication, coordination, and collaboration issues that characterizes GSE projects [#46].

students are demotivated mostly due to lack of commitment by other [remote] *team members, differing goals or lack of motivation from the start of the project, and large differences in knowledge* [#7].

One aspect that had a negative effect on the motivation was one team member who repeatedly failed to deliver what he had promised [#29].

…*distance makes it difficult to establish trust and form relationships among distributed stakeholders* [#2].

…*global teams don't cooperate* [#11].

Another common cause of this lack of synergy is the mere fact that teams are geographically separated. Local teams tend to forget about their global teammates or misunderstand their intentions:

…*forget the other (global) team* [#19].

Since students do not see each other, they have difficulty to generate a picture about their partners in their minds [#72].

The participants in the international teams did not know each other. In some cases, this turned out to be problematic [#46].

5.2.1.2 Teamwork & Task Allocation Challenges

Comments in the task allocation minor category addressed the difficulties students and instructors had in creating teams with balanced skills and managing these teams in fair and effective ways. The studies in the papers described many different reasons for this imbalance. For example, when students themselves determine roles their inexperience or lack of knowledge of a remote team can lead to imbalances. As the papers point out:

Leadership is a critical management skill. In this study, in some projects, it was asked students to select their project manager (leader) by themselves. But this did not work [#63].

The sub standard group had never defined a team leader [#50].

students may not be able to delegate the roles and responsibilities within the project team so that the team can work successfully together [#69].

In addition, this immaturity in leadership often leads to project management mistakes that impact the project and the student learning experience:

student[s] do not estimate time to complete tasks well [#37].

views of perceived experts were given extra weight within the consensus-seeking procedure [#53].

5.2.2 Teamwork recommendations

Teamwork is the most addressed recommendation category in terms of both number of recommendations proposed and number of papers mentioning recommendations. Almost half of the papers provided at least one recommendation to the teamwork problem (see Table 11).

Table 11. Teamwork Recommendations

Theme/Subtheme	Count	# Papers
Teamwork	**80**	**37**
Synergy	49	26
Task allocation	31	21

Teamwork recommendations reflect the unique challenges found in GSD as detailed in the *Teamwork Challenges* section (see section 5.2.1). These recommendations detail methods that can be used to get local and remote teams to communicate and collaborate across time and distance so that trust can be built between the teams and work can truly be effectively shared.

As in the *Teamwork Challenges* section, recommendations are grouped into two minor categories, synergy and task allocation. Recommendations in the synergy category propose methods and mechanisms to help local and remote students communicate, understand, and even bond. Task allocation recommendations, on the other hand, are designed to assist and train students in the management of local and remote groups.

5.2.2.1 Teamwork & Building Synergy

A key part of synergy, according to the papers we examined, is to ensure clear and constant communication. Good communication *creates visibility, better decision-making and a common understanding of shared goals* (Feljan *et al.* [#29]). This communication must start early to have a positive effect on synergy; Cao *et al.* [#12], Matthes *et al.* [#52], Last [#48], and Gloor [#33] all suggest that it's important to *get the* [local and global] *students to be familiar with each other as soon as possible* (Filipovikj *et al.* #30]). Dastidar et.al even recommend that *at least one member from each team, preferably the liaison should travel to the other sites to know the other team members and the team psychology better* [#20].

Part of the initial meetings should be to help local and remote teams develop a shared vision. A shared vision is essential to synergy says Gotel *et al.* [#35] and Favela [#28] while Peña-Mora suggest that instructors *require students to articulate their own vision of the project they want to work on, and then asks them to collaborate with the other class members and the instructors to develop one shared project vision* [#65].

Once an initial understanding is established, studies recommend that instructors continue to be proactive to ensure that teams maintain their synergy. Filipovikj *et al.* [#30] state that instructors must ensure that *communication levels* [are] *consistently high*, Gotel *et al.* [#37] claim that instructors must *focus more than [they] expect upon social bonding activities*, and Nordio *et al.* [#62] recommend *optional group exercises emphasising communication skills*. Crnković *et al.* [#19], Filipovikj *et al.* [#30] and Ende *et al.* [#23] all stress that synergy is only possible if instructors *ensure that students keep the other site in mind* [#30]. The Filipovikj *et al.* paper in particular offers several tips that will keep students from forgetting about the remote site.

5.2.2.2 Teamwork & Handling Task Allocation

The second minor category under the teamwork major category was task allocation, including student team management. Most of the recommendations from the studies in this area concerned team organization. Several papers observed that small teams (both local and global) worked better: *Keeping teams small induces team members to equally contribute to project and allows tutor to monitor individual performances; also communication overhead is kept to a minimum* (Matthes *et al.* [#52]; see also Giraldo *et al.* [#32], and Gloor *et al.* [#33]).

Other authors suggested that self-organized teams worked well (see Fagerholm *et al.* [#26] and Gotel [#35]) though some authors preferred mapping students to particular skill positions (Feljan *et al.* [#29], Neto *et al.* [#58], and Deiters *et al.* [#21]). The trade-off between self-organizing teams and planned teams seemed to be that the former provides more buy-in and synergy from the students while the latter creates technically stronger teams: *balance the expertise within each group so that each group has a range of skills available to it*. (Neto *et al.* [#58]; see also Peña-Mora *et al.* [#65]). Lago *et al.* found that *Both VUA and UDA students found the assignment of roles and roles rotation a good idea to better organize and distribute the teamwork* [#46] and Feljan *et al.* note that *there should be a balance of nations involved in a team; otherwise a kind of favoritism can occur*. [#29]

Some studies recommended appointing team leaders and in-country champions to organize local and global teams [#41, #80]. Serce *et al.* claim that the *experience level of the leader is critical and it is better to assign a more experienced person to globally distributed students teams*. [#72]

A final note of interest is that several studies found that the distribution of responsibilities, as opposed to roles, was crucial. *Deliverables produced by the international teams must contribute more centrally, to the final product* claims Doerry *et al.* [#22]. One paper in particular, Peña-Mora *et al.* [#65], recommend *developing a responsibility chart comprised of the tasks students must complete to fulfill their role in the course is the next move instructors and students need to make*.

5.2.2.3 Teamwork Summary

For instructors, the take away on synergy from these papers is that local and global teams seldom create synergy on their own. It takes careful planning and mentoring from the instructor to develop and encourage this synergy.

Task allocation, in sum, is seen by many studies to be a crucial part of the success of global student projects. The studies seem to indicate that instructor involvement in team constitution and roles, and even in individual responsibilities, is necessary. Any instructor planning to teach a GSD course would do well to think carefully about the makeup of local and global teams and apply some of the principles recommended in this section to their teams.

5.3 People/Soft Issues

This theme was the third most dominant in the group of themes identified, indicating the important role that the socio-emotional dimension plays in globally distributed courses. For students expecting to repeat their often individual experiences from technically focused computer science or software engineering courses, dealing with the people issues which must be surmounted in GSE presents a major challenge [#5].

5.3.1 People/Soft Issues Challenges

As indicated in Table 12 below, challenges under this theme grouped into three primary subthemes: motivation, trust, and stress, with motivation and trust having the bulk of the focus.

Table 12. People/Soft Issues Challenges

Theme/Subtheme	Count	# Papers
People/Soft Issues	**46**	**29**
Motivation	23	17
Trust	18	16
Stress	3	2
Self-awareness	2	2

5.3.1.1 People/Soft Issues - Stress Challenges

As an example of the **stress** experienced by students in a GSD course [#47] report

this is often the first encounter with working in a bigger project group, and it is easy for them to get overwhelmed and lose focus of what we try to convey in the course.

5.3.1.2 People/Soft Issues - Motivation Challenges

The topic of student **motivation** was mentioned in several studies, with a wide range of contributing factors being identified, as noted in the excerpts below:

The topic of student **motivation** was mentioned in several studies, with a wide range of contributing factors being identified, including: *Communication issues; different perception of time, respect for deadlines and low quality work;* [#5] *differences in work culture leading to increased conflict and decreased trust; lack of commitment by other team members, differing goals or lack of motivation from the start of the project, and large differences in knowledge* [#7].

Lack of motivation resulted in *procrastination leading to insufficient time at the end of the project* [#63], and *poor performance: The two groups that developed a bad attitude toward the exercise never bought into the outsourcing idea* [#50].

Method of grading both individual work and group projects may negatively affect student participation and performance [#79], [#45]. *Fear of results may affect motivation* [#69], and *competition between teams sometimes is de-motivating* [#37]. *Division of less glamorous tasks may breed resentment and reduce buy-in* [#54].

As can be seen the range of **motivation** related concerns that arise in GSD courses is considerable, with a resultant need for instructors to actively manage student motivation. Indeed the issue has been strongly stated as a conclusion in a previous Review [47]:

Students involved in GSD training programs usually experience a lack of motivation.

5.3.1.3 People Issues - Self-Awareness Challenges

Cognate with motivation **Self-awareness** was a code relating to student specific attributes that can cause issues in a non face-to-face environment, for instance student concerns relating to *perceptions of self; of respect; and of temporal repercussions from misunderstandings; rendered collaboration less effective in nftf environment* [#82]. In addition work teams with low attention to future time *had greater odds of poor performance* [#75].

While these student degrees of **self-awareness** may not be easy for instructors to impact, it is necessary to be aware of them so that suitable designs or interventions can be considered.

5.3.1.4 People/Soft Issues - Trust Challenges

Issues associated with **trust** were apparent from several studies. The selected issues arising included:

Remoteness - since students do not see each other, they have difficulty to generate a picture about their partners in their minds [# 72], *so assumptions about team members materialise from day one* [#37]. S*tudents lack loyalty, team spirit and collective responsibility* [#19, *trust in the remote teammates*.[#46], and suffer from *Inability to find stakeholders quickly* [#8].

Poor performance: when programming difficulties resulted in some members underperforming, the team began losing trust in them on important tasks [#64], *Two of the teams had a bad attitude related to the use of an outsource programmer. They were unwilling to bring this third party into the team as a contributing partner* [#50].

Qualities important in teamwork, such as trust or cooperation [can be] *an additional challenge if various cultures are involved* [#15], *students are faced with different cultures for the first time, and they may have a hard time accepting behavior not similar to theirs* [#6].

5.3.2 People/soft issues recommendations

This theme in contrast to its third highest placing under challenges was the sixth most dominant in the group of recommendation themes identified, suggesting that the socio-emotional dimension as an area in itself was not given critical focus in devising recommendations. It is interesting to speculate on the reasons for that reduced emphasis, but perhaps some of this focus had already been taken up in recommendations regarding more active stakeholder roles and strategies addressing Global Distance. Nonetheless this remains an area which did attract several recommendations, and the need for motivation from both staff and students was noted in [#4] as a *key success factor*.

As indicated in Table 13 below, recommendations under this theme grouped into two primary subthemes: motivation and, trust echoing their primacy as subthemes in the accompanying challenges category.

Table 13. People/Soft Issues Recommendations

Theme/Subtheme	Count	# Papers
People/soft issues	**29**	**16**
Motivation	20	10
Trust	8	8

5.3.2.1 People/Soft Issues – Inspiring Motivation

For the topic of student **motivation** several concrete recommendations were made. For instance:

give students the choice of co-located or distributed project. [#62]

and

Enhance student motivation through mandatory participation [#32].

As can be seen these are directly contradictory recommendations, but each could play a role in encouraging student participation in learning. The first suggestion however presumably means that only students interested in GSD would participate, which seems undesirable from a GSD learning outcome perspective. Other recommendations can be classified as recommendations for course design or action before the course begins, actions to be taken at the start of the course and actions during the course. Key suggestions for motivational **designs** of the course relate to its being motivational for students through its authenticity or through a degree of competition, for instance:

Genuine global software engineering projects that engage students in activities and deliverables that are truly interdependent are more important to the success of a global software engineering course than anything else an instructor can do [#77].

Both [#4] and [#6] recommend competitions *which motivate students for developing software* [#4, #6], *where student teams work on a software engineering problem, defined by an external customer from a foreign university* [#4].

One recommendation for instructors from the **outset** is the importance of conveying the motivation for the GSE-Ed course [#29].

Peters *et al*..[56], reflecting on student motivational concerns, observe that the use of a learning agreement can be a conscious strategy to assist students to be conscious of the broader learning goals of a GSD course.

Recommendations for instructors **during** the course relate to consistently encouraging and rewarding student work, and active mentorship:

Yet not all concerns relating to student motivation may be readily addressed, as recognized in the frank acknowledgments by [#30]:

Tip 3: Keep communication levels consistently high - outcome communication flow good but some students passive, the lesson learned is that frequent communication is not enough, and that the students' engagement plays a crucial role [#30].

Tip 6: Remember: we are different - outcome positive: The implementation of the teachers' tips prevented issues related to cultural difference in the project. Teachers' view: However the differences in technical skills were significant and the overall motivation of some students was too low to cover for the deficiencies in their knowledge [#30].

In the latter case of course some task redesign and reallocation prior to the course, or implementation of technical mentoring strategies during the course may be helpful. These recommendations would tend to fit under the teamwork/task allocation, or stakeholder role themes.

5.3.2.2 People/Soft Issues – Building Trust

Issues associated with **trust** were apparent from several studies. Again we illustrate selected recommendations for course design or action before the course begins, actions to be taken at the start of the course and actions during the course below. An initial recommendation is that, *teams should travel to the other site* [#11], which is consistent with those from the GSD literature which recommend face to face meetings to alleviate forms of Global Distance [48]. While it is clearly desirable for the teams to meet face to face **before** or at the **outset** of the collaboration, it is not always practical with student teams due to cost or logistical issues. Related challenges and recommendations referring to the cost and sustainability of GSD courses are noted under the category of 'Global Distance: organisational'.

to make the first contact easier, students asked for some ice-breaking sessions, as well as proposed to have additional innovative and fun moments during the course, to break the "serious" course atmosphere [#29].

These recommendations on approaches to 'breaking the ice' need careful design, as cautioned by [#30] who observed the need to move beyond formal introductions to: *a deeper informal interaction between the team members.*

Augmenting the informal interaction may also take the form of culturally specific educational components in the course, which address cultural differences through assignments comparing cultures [#19].

5.3.2.3 People/Soft Issues – Summary

Processes and approaches that build trust will be important if teams are to function and perform effectively. The literature on trust in global teams is large, but we can see here some tensions between what may be assumed to be initial positive assumptions with notions like ***Dispositional trust*** [which] *refers to an individual's ability and willingness to form trust in general*, [62] (*and includes attributes such as openness to experience*), ***swift trust*** [38] (*assumptions of professional competence on the part of peers*) or ***referred trust*** [54] (*trusting behaviours based on instructor's assurances about the remote colleagues*). In reality, as the quotes above indicate, the situation is dynamic and fluid, as not only do the degrees of trusting behaviour that students bring to the course vary, but a form of ***situated trust*** [53] appears to be in operation (*where trust evolves based on the situation and the performances and the cues from team members)*. As observed in [62], interpersonal trust in virtual settings builds based upon the attributes of *Competence, Predictability, Benevolence, Integrity*. So when working in global teams with unfamiliar colleagues, failure to demonstrate competence and behave in predictable ways (especially on the part of remote team members) can be extremely damaging to the fragile initial dispositions and to further trust development, and poses a major challenge to effective student learning in GSD courses.

Instructors need to pay attention to motivational designs for a GSE-Ed course and the need to mentor and encourage students with key interventions at critical stages of a course. But the onus rests also with students themselves. As noted under motivation above, the differences in student skills and subsequent performance can damage trust [#30]. But a strategy of honesty is one that students can adopt to help enlist support and sustain trust:

Be honest about your own technical abilities [#48].

5.4 Stakeholder Role

The stakeholder category encompasses the various roles in GSE-Ed (instructor, student, client, university representative) and considers how their participation creates either challenges/barriers or recommendations to education. Since GSE-Ed is delivered by instructors to students in the context of a university, it not surprising that this category has received attention from many (about a fourth) of the studies.

5.4.1 Stakeholder Role Challenges

GSE-Ed courses have several stakeholder perspectives to manage as indicated in table 14.

Table 14. Stakeholder/Role Challenges

Theme/Subtheme	Count	# Papers
Stakeholder/Role	**36**	**20**
Instructor	14	8
Student	13	10
Client	6	4
Role conflict	2	2
University representative	1	1

5.4.1.1 Instructor Role Challenges

The instructor sub-category involved issues the course instructor(s) encountered with managing a globally dispersed set of students. As can be imagined, the most common challenge for the instructor is the time involved in such management. This idea was repeated in many studies [#37, #76, #7]. This increased workload can also apply to other associated staff [#4].

Another rather obvious challenge is the difficulty planning and coordinating a course that operates in two or more locations. Studies warn about the high degree of synchronization of objectives, classes, and project that must be done between locations [#67, #76, #37]. Similarly, instructors must be prepared to spend significant time coordinating schedules and resources across global classes, but these challenges are addressed in the infrastructure section.

Student communication across teams is one of the particular areas that needs special planning and time from the instructor to be effective. As one author said, *effective communication requires additional time and instructors required a communication plan. Students [are] sometimes unwilling to put the time into effective communication* [#63].

Finally, normal teaching activities like maintaining objectivity, continually auditing students, and keeping students focused become even more challenging in a distributed environment. Gotel *et al.*, for example, talk about the difficulty in maintaining objectivity with global teams [#37] and also about the challenges in properly auditing local and global teams [#37, #34]. And [#1], [#53] and [#67] describe the time and difficulty in managing student meetings.

Overall, running a GSD course requires more time, planning, and monitoring than a normal SE course because of the distributed nature of the course and instructors must be aware of this when teaching or planning such a course.

5.4.1.2 Student Role Challenges

The second stakeholder role we identified in these studies is that of the student. The student role includes issues of student management skills, preparation, and focus. Management skills are a particularly difficult challenge. Many of the courses presented in these studies had no software engineering experience as a prerequisite for the GSD course. As a result, students were learning both processes and management skills in a very difficult global situation. As Gotel *et al.* note, *Students are not prepared for managing resources at distance* [#35]. Problems students have in this area include a lack of attention to details about the distributed process [#53, #35], a frustration and inability to handle a lack of predictability related to global management [#76], an inability to manage deadlines between local and distant teams [#62, #21, #37] and an inability to get global and local teams to attend meetings [#72].

Students also lack a global awareness that is an important part of their role. For example, *global and local groups tend to use tools that they are familiar with but which are different from the other group; makes coordination between groups hard* says Petkovic *et al.* [#67], a challenge echoed by [#2] and [#35]. Students also have a tendency to forget the distant group [#33, #36, #19]. As Gloor *et al.* [#33] put it, *students must remember to inform the rest of the group about their other activities.*

When students do confront the complications of geographic and temporal distance they seem unprepared to handle them and surprised by the effort required. Paasivaara *et al*. [#64, #10] discuss the problems that temporal distance causes students, for example. Lago *et al.* [#46] note that *students working in the international teams were expected to put extra effort because of the communication, coordination, and collaboration issues that characterizes GSE projects* and that *participation in the international teams was perceived as effort consuming*. A result of this increased effort means that *communication issues are an expected demotivating factor* that instructors must manage says Bosnić *et al.* [#5].

The student role, then, sees many challenges that must be addressed by the instructor either through curricular design, course management, or mentoring. Student immaturity in global management and their unfamiliarity with overcoming geographic and temporal distance pose distinct challenges that must be addressed.

5.4.1.3 Client Role Challenges

Another role that poses challenges in GSE-Ed is the "client" role. In the GSD context a "client" could be external to the academic environment or a global student team that is serving as a client. In both cases, similar problems crop up. The most challenging of these is handling feature creep. Gotel *et al.* [#38] say, for example *unlike projects that students create for themselves, the US students were developing code for the Cambodian students (clients) and scope creep was a concern* while Neto *et al.* [#58] saw students having problems with *managing [external] customers and the development process; customers wanted additional functionality etc*. One result of this complication is that *dealing with problems at the customer site impacts on student progress and may cause work redistribution within the team* [#58].

On the other hand, Bruegge *et al.* [#11] note that sometimes the *client is not responsive* which causes acute problems in a course with semester time constraints. When an external client **is** responsive, it causes different problems: *an external customer from industry usually does not possess teaching experience, raising the risk of unsuitable course advising and project support* [#7]. Or, even if the client provides proper support, … *projects requiring an acquisition of highly domain specific knowledge, usually under supervision of the customer. Such knowledge tends not to be communicated to the remote team* [#7], and thus causes problems for the remote team.

Several studies noted that problems sometimes arise not out of a particular role, but from the fact that roles conflict. Clients are voluntary, for example, but students must meet class requirements. Clear *et al.* [#17] describe the problems created by the *tension between voluntary participation of subjects and [the] student role*. Lago *et al.* [#46] note that students themselves assume many roles on local and remote teams and *the use of many roles has been perceived as a limitation (or overload) in local teams*.

5.4.1.4 University Representative Role Challenges

The final role that was described in the set of studies that we considered was the 'university representative'. This role is separate from that of the instructor and includes the critical (but background) managerial, technical and administrative supporting roles, which sanction and enable a GSE-Ed course. A course must operate in the university environment with its structure and rules and Gotel *et al.* [#37] point out that *it is easy to overlook the costs in start-up, set-up and on-going management.* In other words the university role, as it represents the structure and rules of the university, presents a challenge that instructors must plan for and accommodate.

5.4.2 Stakeholder/Role Recommendations

The stakeholder recommendation category (instructor, student, client, university representative) provides ideas for overcoming the various challenges that these different roles face in GSE-Ed. As with the stakeholder challenges category, the stakeholder recommendation category was the second most targeted category for papers. Table 15 provides the statistics for this category.

Table 15. Stakeholder/Role Recommendations

Theme/Subtheme	Count	# Papers
Stakeholder	**62**	**33**
Instructor	48	25
Student	7	7
Client	5	5
University	2	2

5.4.2.1 Instructor Role Considerations

The majority of recommendations were for the instructor role. A reading of the comments indicate that the overall recommendation is that the instructor must be intimately involved with the teams and projects (both local and global) and must provide clear and predictable guidance. Cao *et al.*, for example, state that an *appropriate level of management and control by the instructor are needed in the [global] project* [#12] while Gotel *et al.* suggest that instructors *review planned tasks and help students improve their global estimating skills* and *monitor the health of all team members [local and global] regularly* and McDermott *et al.* claim that [distributed] *groups needed high levels of academic and IT support* [#53]. Junhua *et al.* [#42] also recommend a coordinator role between local and global teams for the instructor and Fagerholm *et al.* suggest that courses *use a resident coach to actively mentor the team and make sure that the project lives up to its expected outcomes for the customer* [#26]. Finally, we hear from Bosnić *et al.* that *regular and frequent team and student status reporting and monitoring; more intensive at start and early instructor intervention is needed* [#4].

Students are obviously immature global software engineers and several studies emphasize the guidance aspect of the instructor role. Serce *et al.* advise, for example, *be specific about deliverables, including who delivers them and when and where they are delivered* and *create the specific locations where the deliverables will be placed* [#72]. Gloor *et al.* echo this, saying [create] *good agendas for each meeting* [between local and global teams] *with exact presentation schedules and clear instructions for what to prepare is necessary to arrange effective virtual meetings* [#33] and Gotel *et al.* suggest *if it is important to do something, provide the students with guidelines* [#37]. Peña-Mora *et al.* [#65, #41] recommend that instructors actively assist students in developing project goals and creating architectural designs.

These recommendations that instructors guide students and provide explicit structure may make it sound like instructors should be fairly rigid in their approach to GSE-Ed courses. In fact, many studies indicate that the opposite is true; instructors must be flexible and anticipate change. Crnković *et al.* suggest that instructors *be flexible, overcome the differences* and *be alert; new problems can arise at any time* [#19]. Ende *et al.* [#23], Keenan *et al.* [#43], Petkovic *et al.* [#66], and Filipovikj *et al.* echo these suggestions, with Filipovikj adding *beat the administration* [#30]!

5.4.2.2 Student Role Considerations

Student roles received much less attention from these studies. The most common suggestion is that students have the appropriate background before entering the course. Students lacking this background can cause serious problems among teams; as Feljan *et al.* noted: *One aspect that had a negative effect on the motivation was one team member who repeatedly failed to deliver what he had promised*; [#29]. Matthes *et al.* had the most specific recommendation: *student level of expertise should be at least in 5th semester of a computer science study project (e.g. late bachelors or master student level)* [#52].

5.4.2.3 Client Role Considerations

Several alternative approaches were suggested for dealing with the client relationship. Bosnić *et al.* [#7] suggest using *student contests (especially software engineering contests)* as *an alternative form of external customer*. Also several papers considered using simulators in lieu of actual clients, with [#44] and [#54] both exploring this approach.

5.4.2.4 University Representative Considerations

A final role identified in these studies was that of the university representative. A university representative, in this context, may be a person outside the formal class structure (perhaps another faculty member) who provides an independent view of the course. Swigger *et al.*, for example, says *it would be helpful to have an independent faculty member have some oversight to keep the bigger picture in mind. It is too easy for the instructor to become focused in minute details*. Gotel *et al.* [#37] echo this sentiment that it is important to *establish independent third-party oversight* to ensure that projects do not get out of hand.

5.4.2.5 Stakeholder Role Summary

In addition to the instructor and student roles, studies mentioned two other common roles, that of the client and the university. The client is usually external to the course and will be remote from at least one of the teams. This poses challenges for students and instructor. Fagerholm *et al.*, [#26], for example, comment that *close customer participation is a critical success factor for producing a software product in seven weeks with a newly composed software team*. Bosnić *et al.*, [#6] further add *the customer involved should be company representative who should be willing to spend time with the students discussing the project proposal and status* an idea echoed in Paasivaara *et al.* [#64] that says that an instructor must *ensure frequent communication with the customer*. Overall, these studies reflect the necessity of close local team/global team/client relationships if a project is to succeed within the limited timeframe of a course.

In brief, we learn from these studies that instructors planning GSE-Ed courses must be aware of the importance of the instructor role in particular. Instructors must carefully consider how they will guide both local and global students through the software engineering process and must ensure that all teams have clear guidelines and structures. The studies mentioned above all give excellent guidance for establishing guidelines and providing guidance and these suggestions should be considered carefully when embarking on a GSE-Ed course.

5.5 Infrastructure

The infrastructure theme comprises challenges and recommendations related to development platforms and tools, communication infrastructure such as instant messaging and video conferencing, and source code control (SCCS) systems.

5.5.1 Infrastructure Challenges

Table 16. Infrastructure Challenges

Theme/Subtheme	Count	# Papers
Infrastructure	**33**	**23**
Tools	19	14
Technical issues	12	12
Version control	2	2

Issues related to infrastructure are another category of challenges that are faced by both global software development projects and global software development courses. And, like other challenges discussed previously, there are infrastructure challenges that are unique to the educational context.

5.5.1.1 Infrastructure Tools & Challenges

Infrastructure challenges were mentioned 33 times in 23 papers. Nearly a third of these concerned communication and collaboration technology, including both the difficulty in installing and learning how to use, such technology [#32, #46, #82], to the lack of reliability of some tools [#13, #23, #76].

Differences in available technology at different locations is also a problem [#12].

5.5.1.2 Infrastructure Version Control Challenges

Several studies pointed out the problems caused by lack of a shared Source Code Control System (SCCS). Lack of a shared SCCS not only makes it difficult to identify the latest version of an artifact [#8], but also makes inter-team coordination more difficult [#42.] This goes beyond simply sharing code among groups; as Berkling and colleagues observed, *in distributed projects, the physical location of information artifacts such as source code, task descriptions, or comments on changes, and the lack of 'global knowledge' about their existence make traceability and rationale management an especially hard task* [#2].

5.5.1.3 Infrastructure Technical Issues & Challenges

Differences in tools and environments across sites, was also a frequent problem. Part of this is simply due to the natural heterogeneity present when different teams from different institutions are involved. However, attempting to eliminate heterogeneity by imposing a common development environment or tool set across all teams brings its own problems. In addition to the extra effort required on the part of both instructor and students [#53], the chosen solution may not actually work in a distributed context [#42]. And even if it does work, a common tool set must overcome inertia on the part of students; as Petkovic observed, ... *global and local groups tend to use tools that they are familiar with but which are different from the other group...* [#67]; this inertia makes adopting a uniform environment and tool set difficult, even if institutional and administrative resistance can be overcome.

5.5.2 Infrastructure Recommendations

A total of 49 recommendations address some aspect of infrastructure.

5.5.2.1 Infrastructure Platforms & Tool Considerations

Of these recommendations, over a quarter specifically address communication, including video conferencing facilities [#8, #19, #29, #65]. Braun and colleagues suggest that both client reviews, and progress meetings with instructors or other teams, should be conducted via video conference [#8]. Feljan assert that it is important to have video conferencing for a "flawless" GSD course, despite the fact that *technical glitches occur from time to time* due to bandwidth limitations or other connection problems [#29].

Five studies suggest using groupware and similar collaboration tools to bridge the distance gap between teams [#8, #17, #64, #71, #76].

Finally, Junhua recommends desktop sharing software for both team meetings and meetings with the instructor [#41], while Gotel and colleagues suggest that communication tools need to be supplemented with *simple calendar, scheduling, and notification tools to assist meeting logistics* [#37].

Of the ten mentions of solutions to the challenge of heterogeneous infrastructure, most suggest using a common environment or set of tools across all sites [#4, #16, #27, #35, #41, #48, #61, #61, #65, #65]. But how should one achieve this? Gotel and colleagues recommend selecting a "minimal" tool set to be used at all sites [#35]. In contrast, Bosnić *et al.* suggest that students should be allowed to choose their tools, rather than having choices imposed by the instructor [#4]. To make deployment of such a common tool set easier, Junhua recommends using a virtual machine loaded with open source tools (to avoid licensing issues) [#41].

Five suggestions involve various kinds of knowledge management tools. Both Junhua [#41] and Nordio *et al.* [#61] suggest using a wiki for both distributing course information, and capturing project discussions. Junhua also suggests having each team deploy a web site to distribute plans and project status [#41]. Carlson and Nan observe that an SCCS provides knowledge sharing as well as archiving capabilities [#13]. Braun and colleagues recommend that communication tools used for informal communication should be able to record such conversations, as a way of capturing and sharing project knowledge [#8].

In addition to recommendations for different kinds of tools or infrastructure, a few recommendations address *how* to use these tools. For example, Monasor *et al.* describe an approach that uses chatbots to allow students to learn how to communicate with colleagues from different cultures [#54]. Romero and colleagues go so far as to assert that instructors *must teach communication groupwork tools* as part of a GSD course [#71], while Nordio *et al.* recommend that student teams be required to write a "communication plan" in order to encourage frequent, effective communication [#62].

5.5.2.2 Infrastructure Summary

Despite the obstacles posed by heterogeneity, appropriate tools are important to help bridge the communication gap opened by lack of informal face-to-face communication [#2]; this is especially true for projects employing Agile methods [#35].

In summary, while tools are an important part of any software development effort, additional tools, especially for communication, are required for global software development, and for global software development courses. Also, even conventional tools such as source code control take on an additional role in helping to bridge the Global Distance gap [#13, #41].

5.6 Curriculum/Pedagogy

This theme was the sixth most dominant in the group of themes identified, but its frequency fell within a secondary grouping of themes that included stakeholder and infrastructure issues, so was not insignificant.

5.6.1 Curriculum/Pedagogy Challenges

As indicated in Table 17, challenges under this theme grouped into three primary subthemes: course design, learning outcomes and pedagogy, with course design having the bulk of the focus.

Table 17. Curriculum/Pedagogy Challenges

Theme/Subtheme	Count	# Papers
Curriculum/pedagogy	**32**	**18**
Course design	27	17
Learning outcomes	5	4

5.6.1.1 Curriculum/Pedagogy & Course Design Challenges

As an example of the **Course design challenges** that arise in a GSD course, the curriculum needs to incorporate a focus on 'soft skills' in addition to the technical, for instance …*in the areas of cross-cultural communication and teamwork, with sensitivity to East-West differences in management* [#27].

But as Gotel *et al*. [#37] ironically observe, *learning soft skills is hard*.

Several student related challenges which have implications for **course design** arose. These included *students wanting freedom in choosing projects, and technologies* [#29]. Student concerns about *misbalance in knowledge levels of students enrolled to the course, ranging from students who have poor knowledge in programming and basic software engineering disciplines, to ones who have specific knowledge not required for their particular project* [#29]. Other concerns included: workload being too high [#29]*; the* granularity of task decomposition being too coarse [#35]*; just-in-time learning – [since] teaching content as the students need it made it difficult for students to plan far enough into the future* [#38]. Instructor concerns about *providing feedback regarding project grading too late in the course* [#7]*;* and *unequal grading and evaluation schemes across institutions* [#52].

Then there were the challenges related to providing realistic and "authentic learning" experiences [33], through **course designs** which accommodate the needs of the collaborating parties. As one report observed, *students must experience GSD to understand the challenges* [#70]. Challenges were noted in *setting up realistic settings that could allow the students to tackle representative problems* [#56]*;* with having *students actually participating in a real-life, multi-site, globally dispersed, industrial project and thus acquiring knowledge from experience,* [#81]*;* defining a project *at the same time attractive, of suitable complexity, easily modularized for distributed development, based on available technologies, and so on* [#7]. Cross site tensions were noted, *between running a successful global and collaborative project and accommodating the wider curriculum demands of each participating institution* [#37], and *preparing a common course and complying with local admin rules on each side [#4], as was the sparseness of curriculum materials* [#76].

More specific challenges related to the **design** of the ***course architecture***, which needs careful consideration. In one case the ***course architecture*** resulted in a communication bottleneck through the need for teams to communicate through a central team which became overloaded and *the level of detail that could be conveyed by this "middleman" was sometimes limited* [#68]. In another case cross-development phase collaboration issues made it *difficult for both sites to collaborate if they are in different phases, and thus, have a different focus on the project* [#11].

5.6.1.2 Curriculum/Pedagogy & Learning Outcomes Challenges

The topic of **learning outcomes** had a lesser focus but did raise some important challenges for learning and scaffolding strategies. For instance, the inability of students to look ahead and plan their work -*Consequently, they learn about requirements when they need to do requirements and they learn about testing when it comes to testing* [#37], and the tendency for *the students to get blinded by their particular role, thus not getting a holistic view* [#29], which leads to a strong *need to mentor students* [#34].

5.6.2 Curriculum Pedagogy Recommendations

This theme ranked second behind teamwork as the most prominent in the group of recommendation themes identified, in contrast to its lower placing under the challenges category This would suggest perhaps naturally that design of the student teaching and learning experience was seen as a key area for constructive intervention in GSE-Ed courses.

Table 18. Curriculum/Pedagogy Recommendations

Theme/Subtheme	Count	# Papers
Curriculum/pedagogy	**71**	**34**
Course design	64	31
Learning outcomes	7	7

As indicated in Table 18, recommendations under this theme echoed the challenges for the theme, and grouped into two primary subthemes: course design and learning outcomes with course design having the bulk of the focus.

5.6.2.1 Curriculum/Pedagogy & Course Design Solutions

Reflecting the complexity of the topic, a rich variety of strategies may be adopted as **Course design recommendations** for a GSD course. On reviewing the 64 recommendations arising from this review, these strategies in turn fell under several groupings:

> Course architecture; assessment; authentic professional experiences; collaboration; culture; curriculum development; evaluation; gamification; management skill development; Open Source Strategies; considerations at the outset of the course; considerations prior to the course; process related approaches; nature of the project; scaffolding learning; the course schedule; simulation strategies; task allocation and tools.

While space precludes an exhaustive enumeration here, since these are key recommendations for practitioners, selected examples of these strategies will be outlined below.

Under the category of ***course architecture***, considerations include: allocating different modules of a large system to distributed teams [#62, #20]; and in a more sophisticated, scaled and graduated approach Keenan *et al*. [#44] recommend applying four selected GSD teaching patterns: *remote testing*; *subordinate role*; *partitioning*; *continuous development*. The latter design is a strategy that incorporates a conscious approach to ***scaffolding*** student learning. A design that enforces choosing *a topic that naturally needs lots of cross team communication* [#52] is also an approach that can encourage collaboration.

Several recommendations address ***assessment***: it is important to design an assessment process tailored to GSD, with rules adapted from GSD practice [#57]; inform learners about assessment objectives [#57]; in grading: emphasize the entire software lifecycle [#67]; identify the learners' starting skill set by self-assessment, assess theoretical knowledge and interaction skills, final summative evaluation and learner self-assessment [#57]; define three deliverables for evaluation: initial presentation; final presentation and final report [#52]; and in a contrasting recommendation: *staff should make a thorough analysis and testing of the final product in the end and grades should be more influenced by the product quality. Students who gave their best should be awarded, with a greater distinction to the ones who invested less effort* [#29]. Approaches to ***evaluation*** were also recommended, one including evaluating teaching quality as well as student learning [#41] and the other recommending incorporating qualitative techniques [#15]. A further efficiency recommendation was that evaluation sheets take no *longer than 10 -15 minutes to complete for each project team and deliverable* [#52].

Learning by doing was advocated by several authors, with a goal of the course being seen to provide ***authentic professional experiences***, in which students are exposed to realistic experiences [#8] while still in the education process [#15] and exposure to software engineers from different cultures [#14].

Yet, in contrast to the above view, GSD courses were seen to be too difficult and complex for institutions to run, indeed Monasor further concluded, given the difficulties of covering the multiple aspects of GSD, that *any initiative should be focused on a specific field* [47], and so ***simulation*** strategies were one recommended option. These ranged from suggesting single site exercises run as simulations [#49], through developing courses with simulation scenarios including virtual meetings for cultural training [#54]; using a simulator for training in requirements elicitation [#70]; using a simulator for training in the decision making process and trade-offs in GSD [#59]. Augmenting simulation is the use of ***gamification*** as a learning strategy with games and contests being proposed [#60, #80].

Curriculum development is an important topic given as noted in [#76] the *sparseness of curriculum materials for GSD*. Topics to be taught include: supporting distributed groups and global distributed software development, cross-cultural communication, international ethics, problem solving [#27, #32, #76]. Courses recommended teaching modules or activities that addressed ***culture*** [#4, #14, #29]. As reported in [#14] *GSD modules with their particular emphasis on culture were a success. Modules include lectures, seminars, readings, and interviews of global software engineers*. Adding to the soft skills focus [#62] recommended that the course *have optional group exercises emphasising management skills*. One strategy recommended for curriculum development included using an ***open source*** community approach by *developing a core set of reusable instructional material and establishing a common web-based infrastructure supporting distributed collaboration* [#27].

Course sequencing was an important area, with establishing ***the course schedule*** and incorporating regular deadlines considered critical [#4]. Key activities needed to be conducted at distinct stages of the course. ***Prior to the course*** a set of pre-semester GSD training sessions [#62] or a crisp preparatory GSE overview with a project management focus [#52] were advocated. At the ***outset of the course*** it was recommended that instructors hold *lectures on past courses and typical challenges experienced [#4];* and to minimise student frustration, *should explain to the students the rationale behind vague requirements [#29]*. It was also considered important that *all sponsors and tutors consistently state the main objective right at the beginning [52]*, and keep a strong *focus upon the* ***process*** *before the project topics and tools [#37]* The nature of the ***project*** was considered of critical importance. Project feasibility including ability to be tested within the time allocated was the major consideration for some [#42], (e.g. *keep project scope to three months with prior defined outcomes* [#52]; *keep project simple* [#62]). Alignment with the sponsor's needs [#52] and delivering a complex software system for a real client [#8] were key elements of other recommendations. Two cautions were noted however; *do not try to run disparate projects with the latest technologies until the underlying process works* [#37]; and *it is not sufficient to assign a real-life project; it is also important to make it deployable and sustainable* [#37]. The latter of course has implications for hand-off processes and a clear definition of the scope of the project. Within the context of a project where student skills may be mixed a ***task allocation*** strategy may be needed, *allowing student teams to pay to outsource parts of their project to a global developer is an effective means of teaching GSD [#50]*. While this recommendation has a pedagogic focus, the topic of *task allocation* is treated more comprehensively under the *teamwork* theme.

On the topic of ***tools*** one recommendation suggested familiarising students with commonly used case tools in industry [#68]. While coded in this case under curriculum, the topic of tools is dealt with more fully under the *infrastructure* theme.

5.6.2.2 Curriculum/Pedagogy & Learning Outcomes Approaches

Again, as with the challenges for this theme, the topic of **learning outcomes** had a lesser focus but did propose some applicable approaches for learning. Key recommendations focused on the use of reflection to develop students insight into their own learning [#34] and by making the learning from the course more explicit, thereby inculcate the habits of a 'reflective practitioner' [60] through *a final phase of reflection about what the student has experienced* [#77].

Further recommendations had a course and assessment focus:

have optional group exercises emphasising communication skills [#62] and, *whole class project presentations and feedback* [#4].

Yet others focused on the mentor role of the instructor in guiding students towards achieving the learning outcomes [#34].

5.6.2.3 Curriculum/Pedagogy Summary

As can be seen there are a set of tensions between course design, learning outcomes and student inclinations. For instance, it may be logical to design a course of the inherent complexity that GSE presents, with a just-in-time learning philosophy [#7] to help scaffold student learning, but this works counter to the innate student inclination not to look ahead and plan their work [#7, #37]. Striking a balance in course design to meet learning outcomes, pace and complexity of material to be covered and mentoring students with differing skill sets is a challenge for instructors, who wish not to be dragged into the role of technical lead for their teams.

A concluding challenge for course design really serves to motivate the need for a GSE-Ed course and especially a front end lifecycle dimension to a GSE-Ed course:

professionals who have recently graduated from universities often lack the skills and abilities to do global requirements elicitation [#70].

The general strategy of 'learning by doing' advocated above, was also noted as a key conclusion in the review by Monasor *et al.*. *The teaching and training of GSD must be supported by practical experiences through which students can learn by doing* [47].

Final recommendations had a broader focus than the course itself, with the first recommendation noting the need for a build up to such a course within an educational programme. Similar points have been made in [25; 55] suggesting integrating the course with the *whole software engineering curriculum, so students will have necessary skills to complete a distributed project* [#41]. A further recommendation proposed constructing *a research linked model of learning and teaching* [#18], which relates to the recommendation in [#17] and reflects concerns also noted in [#27] namely *establish a sustained and adequately funded research project* as a strategy to fund and sustain a longer term GSE-Ed initiative. This recommendation also advocates for an extension of the curriculum and student learning towards the broader forms of scholarship in teaching and learning, integration and application advocated by Boyer [12], and towards the 'scholarship of engagement' in which the academy and society more closely interact [59].

5.7 Development Process

Development process challenges and recommendations concern different phases of the software lifecycle, from requirements to integration and testing. Consequently, one might expect that process issues, and recommendations, would feature highly when discussing global software development education.

However, as noted in Table 19, relatively few studies (11 of 82) present software process challenges, or recommendations (13 of 82) as presented in Table 20.

5.7.1 Development Process Challenges

Table 19. Development Process Challenges

Theme/Subtheme	Count	# Papers
Development process	17	11
System/code integration	6	4
S/w development process	4	4
Coding	2	1
Design	2	2
Requirements	2	2
Testing	1	1

The most frequently occurring process challenge is system integration, with seven challenges mentioned in four studies [#24, #44, #62, #63]. This is perhaps not surprising, since successful integration requires distributed teams to collaborate effectively, both in designing and adhering to effective interfaces among components, and in performing actual integration. The two main integration problems are integration failures before deadlines [#44, #62], and merge conflicts during integration [#24, #44].

General software development process problems received four mentions [#17, #18, #35, #68], ranging from poor quality resulting from lack of process [#17], to problems arising when processes are not followed [#18], to the effects of constant change on development [#35, #68].

Requirements issues mentioned feasibility [#36] and negotiation [#49]. Coding issues centered around difficulties understanding and modifying legacy systems [#44].

Of particular interest is that two studies [#41, #44] mentioned that students lack design experience, which resulted in too much time spent on design [#41], and designs that were not partitioned effectively into modules that could be developed independently. The latter is particularly important in a distributed context where Global Distance results in communication delays.

The one testing challenge mentioned stemmed from handing the testing task over to a remote team, a practice that is sometimes seen in outsourcing arrangements [48]; this practice was characterized as "difficult" and was seen to take longer than if the testing was done by the same team that developed the code [#44].

5.7.2 Development Process Recommendations

As table 20 indicates, despite being the most often mentioned challenge, only two studies [#24, #62] had recommendations related to system and code integration: "manage" merge conflicts when integrating software [#24], use design by contract to specify module interfaces using the Eiffel programming language [#62], and require mandatory code review of those interfaces before proceeding to implementation [#62].

Table 20. Development Process Recommendations

Theme/Subtheme	Count	# Papers
Development process	**16**	**13**
S/w development process	8	6
Design	3	3
System/code integration	3	2
Coding	1	1
Requirements	1	1

Design phase recommendations include partitioning for independent development [#52], and documenting design decisions and rationale to facilitate knowledge transfer to other teams [#46].

The one coding suggestion recommended forcing students to use a common programming language [#63], while involving students in requirements specification emphasized the importance of communication [#50].

For development process recommendations, two studies emphasized that teams should start design and implementation as early and possible [#13, #20], two recommended adapting Agile methods [#20] including pair programming [#58], and three advocated daily or weekly status meetings [#58], or summary reports [#23, #35], and one advocated prototyping as a way of exposing "emerging" requirements [#17].

5.7.2.1 Development Process Summary

Viewed as a whole, the key take-away in the process area for instructors planning a Global Software Development project course is to avoid integration problems by first ensuring modular designs, with modules that can be developed independently, and start implementing early. Some work has been undertaken in GSE to uncover the differences in GSE architectural knowledge management that shows the dependencies between the components and stakeholders [2; 8].

6. DISCUSSION

This working group has examined what turned out to be a voluminous body of literature on the topic of GSE-Ed. The scale of the undertaking has surprised, if not at times daunted us. Nevertheless, the richness of the material uncovered has allowed us to derive a solid set of results which answer both our research questions, relating to the challenges and recommendations for GSE-Ed. We also believe that this study adds considerably to our current stock of knowledge on how to better conduct such courses.

6.1 Recommendations

We have identified many challenges faced by educators who attempt to deliver a course on Global Software Development. These are due in large part to the nature of Global Software Development itself: geographic separation, cultural differences, and lack of timezone overlap present barriers to communication and collaboration, which in turn affect other aspects of the development experience. In addition, *teaching* Global Software Development brings its own unique challenges, especially when differences in curriculum among participating institutions is considered. Fortunately, we have identified numerous solutions to all of these challenges; the most frequently mentioned of these are presented in Table 21.

Global Distance is, not surprisingly, the most frequently mentioned challenge, and has been addressed in detail in Section 5.1. The most common solutions focus on improving communication, by building communication into the development process, and by improving cultural awareness to make communication more effective.

One of the unique challenges to GSE-Ed posed by Global Distance derives from the differences among participating *institutions* regarding schedules, policies, and expectations. The most common solutions for this challenge are to make the schedule and expectations (including deliverables) as uniform as possible across institutions, and to make sure both are well-defined at each location. While these recommendations focus on the student experience, a third counsels the instructor to choose counterparts wisely.

Teamwork is a challenge for any Software Development course, but is especially so for GSE courses, due to the aforementioned communication barriers. Recommendations for addressing teamwork focus on two areas: communication, where the advice is to encourage students to communicate early and often by including communication in the development process, and teaching communication skills as part of the course; and, the students themselves, where solutions include creating roles (such as local leaders or champions) that promote teamwork, matching skills to roles, and keeping teams small to encourage team bonding.

Some of the most interesting solutions address *People/Soft Issues*. The most commonly mentioned also apply to Global Distance and Teamwork: communicate early and often, and promote bonding through social interaction, games, and required participation. Instructors are advised to be exceptionally enthusiastic.

The most commonly mentioned solution for addressing *Stakeholder/Role* challenges for students is to make project requirements and roles, especially a group project manager, clear. The same advice applies to instructors: increased workload should be met by clearly defining roles and responsibilities at each location. Finally, several papers suggested using simulated rather than real clients.

Recommendations to address *Infrastructure* problems most frequently involve communication technology to overcome the lack of face-to-face encounters. These include collaboration tools such as groupware and wikis, and video conferencing. A unique problem for GSE-Ed is tool and environment heterogeneity; most commonly this is addressed by providing a common environment across all sites.

Curriculum/Pedagogy challenges are unique to GSE-Ed (as compared to GSE in general); the main solution is to scaffold learning, through mentoring as well as course design. As elsewhere, soft skills (such as communication and cultural awareness) are also frequently mentioned. Also recognized is the need to be realistic, as reflected in achievable learning objectives and assessment tailored to GSE.

Finally, a few recommendations address the *Development Process.* Regular meetings are frequently mentioned; also mentioned is documenting designs that are partitioned for independent development.

To summarize, communication, in the form of scheduled, frequent meetings that start early in the course, as well as course content focused on learning how to communicate, is a recurring recommendation across challenge categories. This is not surprising, as many of the challenges arise from barriers to communication that students do not face in their conventional classes. Consequently, they may not have the skills or experience to overcome these barriers on their own.

6.2 Comparison to Previous Work

It is informative to compare this study with the results gained from four prior studies. The first of these is the study by Noll and colleagues [48] into the same concerns for GSD practitioners. The second, by Crnkovic *et al.* [23], presents “ten tips” for GSD educators based on the authors’ experience. The third is the set of conclusions from the review of GSE-Ed by Monasor [47], and the fourth is a paper by Damian [24] (cited in [47]), in which a framework for conducting their GSE-Ed course is presented. As can be seen in Tables 22 and 23, no single study covers all of the categories our study has identified in GSE-Ed, in addition to the issues facing GSD practitioners.

Table 21. Top GSE-Ed Challenges and Recommendations for Educators, Synthesized From Our 82 Papers

Challenges	Recommendations
Global Distance	
Limited time overlap between sites inhibits communication and causes delays [#1, #4, #7, #8, #10, #12, #13, #19, #20, #22, #26, #31, #35, #37, #38, #40, #46, #49, #51, #52, #53, #54, #55, #61, #63, #64, #66, #67, #68, #69, #73, #74, #76, #78, #80]	1. Schedule regular meetings in advance [#13, #35, #72, #48] 2. Make teams communicate more often than they normally would [#19, #48, #62] 3. Use synchronous and asynchronous media [#49, #55]
Participating institutions have different term schedules, expectations, and regulations [#4, #5, #7, #8, #9, #12, #13, #16, #17, #19, #22, #23, #25, #26, #27, #41, #43, #44, #51, #52, #54, #55, #56, #62, #63, #67, #68, #74, #76, #80]	1. "*Harmonize deliverables, time schedules and evaluation schemes across all participating universities*" (but be flexible) [#4, #23, #41, #23] 2. Ensure roles are well-defined at each institution [#15, #16, #28] 3. "*find a supportive, reliable and patient collaborating partner*" [#17, #35]
Students don’t know how to work with people from different cultural backgrounds [#4, #5, #6, #15, #19, #23, #27, #28, #33, #37, #38, #40, #43, #44, #45, #49, #54, #60, #61, #64, #66, #68, #69, #75, #76, #80, #82]	1. Include cultural awareness topics in course content [#27, #71, #54] 2. Provide opportunities for students to learn about cultural difference from each other [#36, #45]
Teamwork	
Large effort and commitment is required by students to engage global teams [#2, #7, #11, #29, #33, #46, #73]	1. Create communication exercises to teach students how to use communication tools [#6, #37, #60, #62] 2. Instructors must monitor communication levels to ensure that they remain high [#30, #35]

Challenges	**Recommendations**
Local teams tend to forget about their global teammates or misunderstand their intentions [#2, #19, #33, #37, #46, #72, #78]	1. Start communications early to form relationships [#12, #20, #30, #33, #48, #52, #66] 2. Create a shared vision between local and global teams [#28, #35, #65] 3. Create bonding exercises [#23, #30, #37]
Difficulties for both students and instructors creating teams with balanced skills and managing these teams in fair and effective ways [#7, #23, #37, #44, #46, #50, #53, #63, #69, #72]	1. Instructor should map students to particular skill positions [#21, #29, 46, #58, #65] 2. Appoint team leaders and in-country champions [#41, #72, #80] 3. Keep teams small [#32, #33, #52] 4. Use self-organizing teams [#25, #35]
People/Soft Issues	
Lack of student motivation [#5, #7, #19, #29, #34, #37, #39, #44, #45, #46, #50, #53, #54, #63, #69, #77, #79]	1. Use contests and games to boost student motivation [#4, #6, #19, #81] 2. Be exceptionally enthusiastic as an instructor [#4, #19, #29, #65] 3. Require participation to maintain student engagement [#30, #32]
Lack of trust between teams [#6, #8, #15, #19, #33, #37, #44, #46, #50, #62, #64, #66, #67, #72, #76, #82]	1. "*focus more than you expect upon social bonding activities and communication protocols, and from day one*" [#11, #19, #23, #29, #35, #37] 2. Students to be honest about their own technical abilities [#48].
Stakeholder/Role	
Student lack of distributed project management experience [#2, #10, #12, #20, #21, #35, #38, #53, #62, #72]	1. Clearly define project organization and requirements [#19, #22, #23, #33, #37, #41, #61, #62, #65, #72, #80] 2. Require a designated group project manager [#15, #41, #58, #62, #72]
Increased workload for instructor [#4, #7, #18, #37, #76]	1. Create a complete class structure so that instructor responsibilities are clearly delineated [#19, #37, #65, #72] 2. Use a coordinator or coach to help teams with the project [#11, #26, #42] 3. Establish support systems for both students and instructors [#37] 4. Scale down the project if you change the course in any way [#37]
Client lack of education experience [#7, #11, #38, #58]	1. Use simulators in lieu of real clients [#44, #45, #54, #55, #59, #70] 2. Instructors should ensure that a close relationship is built with the client [#6, #26, #64] 3. Use programming contests in lieu of clients [#6, #7]
Infrastructure	
Communication and collaboration technology challenges [#13, #23, #32, #46, #76, #82]	1. Use groupware and similar collaboration tools to bridge the gap between teams [#8, #17, #64, #71, #76] 2. Use wikis and other knowledge management tools [#8, #13, #41, #61] 3. Use video conferencing facilities [#8, #19, #29, #65]
Tool and environment heterogeneity [#42, #53, #67]	1. Use a common environment or set of tools across all sites [#4, #16, #27, #35, #41, #48, #61, #61, #65, #65]
Lack of shared SCCS [#2, #42]	1. Use Git or similar SCCS [#13, #41]
Curriculum/Pedagogy	
Tensions between course design, learning outcomes and student inclinations [#37,#27,#15, #29, #70, #56, #38, #76, #52, #68, #7]	1. Scaffold student learning, through course design & mentoring [#62, #67, #37, #42, #81,#76, #81, #16, #27, #46] 2. Explicitly teach soft skills [#54, #76, #27 #14, #29, #60, #62, #4] 3. Encourage reflective attitudes in students [#34, #18, #77]
Difficulty of real-life, globally dispersed projects [#81,#37,#87,#15,#7,#37, #56, #70]	1. Design achievable and authentic learning experiences [#15, #67, #62, #42, #8] 2. Design an assessment process tailored to GSD[#57,#15,#52, #29,#41, #67]
Development Process	
System/code integration failures [#24, #44, #62, #63]	1. "Manage" merge conflicts when integrating software [#24] 2. Use design by contract to specify module interfaces using the Eiffel programming language, and require mandatory code review of those interfaces before proceeding to implementation [#62]
Requirements, design, testing failures [#36, #41, #44, #49]	1. Start design and implementation early [#13, #20] 2. Partition designs for independent development [#52] 3. Document design decisions and rationale [#46]
Other process failures [#17, #18, #35, #68]	1. Require regular status meetings and/or reports [#20, #23, #35, #58] 2. Force students to use a common programming language [#63]

Table 22. GSE-Ed Challenges — Comparison with Other Frameworks

GSE-Ed Challenge (Current study)	**GSD Theme** (Noll *et al.*, 2010)[48]	**Ten Tips** (Crnkovic 2012)[23]	**Student Preparation for GSD** (Monasor *et al.*, 2010)[47]	**GSD Instructional Design Framework** (Damian *et al.*, 2006)[24]
Global Distance	Language and cultural distance Temporal distance Geographic distance	Inflexible sets of rules from different institutions brings unsolvable situations Some students had more flexible interpretation of time Some more open and direct in conversation, some avoid confrontation Language differences cause difficulties	Schedule problems , communication difficulties - greater with cultural and language differences Different timetables of students make it difficult to coordinate projects	
Teamwork	Management	Different understandings of teamwork Students lack loyalty, team spirit and collective responsibility		International teamwork
Curriculum/ pedagogy		Technical capabilities differ between students. Causes problems in coordinating development		
Stakeholder/role	Organization		Simulating complexity of real environments difficult for universities	Distributed Project Management
Infrastructure	Infrastructure		Specific tools required for communication, collaboration and document management.	CMC
People/soft issues	Fear and Trust	Students with different backgrounds have different sources of motivation	Students in GSD training usually experience a lack of motivation	Ambiguity/uncertainty
Development process	Process Product architecture			Iterative development

As noted in section 5.1.3, Global Distance challenges and recommendations found in GSE educational settings are reflections of similar issues reported in the general GSE literature, such as issues relating to cultural, geographic, linguistic, and temporal distance [48; 58]. In addition to Global Distance, Noll *et al.* [48] identified five other categories of general GSD challenges: process and management issues; fear and trust; infrastructure; organization; and product architecture. As shown in Table 21 these have corresponding themes in the present study.

However, there are peculiarities of the educational setting that pose different challenges from industrial practice. For example, the challenge of dealing with different schedules across institutions, and allocating tasks to students with very different backgrounds and skill levels, are unique; these are captured by the Curriculum/pedagogy theme. This theme was also recognized as a challenge by Crnkovic *et al.*, but they offered no specific solution. On the other hand, neither Monasor *et al.* nor Damian *et al.* explicitly identified Curriculum/pedagogy as a challenge, but both offered some solutions in this category.

The mapping in Table 23 focuses on recommendations, where again differences and gaps which distinguish our findings from earlier studies can be noted. The ten tips recommendations in Crnkovic *et al.* [23] (also [#19] in our reviewed papers), address most of the categories apart from development process, but offer a single point of view and are more general than the broader set of concrete recommendations we have elaborated in this study.

As can be seen, Damian *et al.'s* [24] framework does not address all of the categories, and is also a very context specific presentation. Taken as a whole, these four studies (columns 2-5) suggest the validity of the categories we have identified, while the gaps confirm the need for a comprehensive review like ours. The differences between our study and Monasor *et al.*'s [47] conclusions are evident, reflecting the directive and recommendation focused nature of this study, as opposed to their survey of the state of the art. As such, our study has made a contribution, through its detailed mapping of challenges, and through a comprehensive set of recommendation for practitioners; the most frequently mentioned of these challenges and recommendations have been presented in Table 21.

As Table 22 and 23 illustrate, and as noted in Section 5.1.3 above, our theme boundaries, while useful in terms of illustrating the many different areas to consider when developing and conducting GSE educational courses, are not rigidly fixed. A cross theme view must be taken in order to gain a holistic picture of a course.

When considering recommendations made in our other themes, for example 'Infrastructure', these will have a positive impact on our 'Global Distance' issues. Therefore we advise anyone applying these recommendations to look across all our themes to gain a full and balanced picture of what is needed to conduct a GSE-Ed course. Should an even finer grained view of process and practice be required than presented in this systematic literature review (SLR), we suggest that the reader goes directly to the underlying studies that are grouped in terms of each issue they address.

6.3 Limitations

This study is very broad in two dimensions, the number of papers considered and the topic itself. In terms of the number of papers, 649 unique papers were considered and 82 were ultimately examined. The screening process was rigorous (see Section 3.8, Validation), yet it is quite possible that relevant papers were passed over. The considerable scope of the topics covered within GSE-Ed and the many headings and site/project specific terms used, together with a variable focus on the educational aspect of the publication, mean that GSE-Ed is inherently a challenging candidate for a systematic review. The quantity of examined papers and the rigor of the examination, however, give us confidence that the major challenges and recommendations of GSE-Ed were uncovered.

Table 23. GSE-Ed Recommendations — Comparison with Other Frameworks

GSE-Ed Solution (Current study)	**GSD Theme** (Noll 2010)[48]	**Ten Tips** (Crnkovic 2012)[23]	**Student Preparation for GSD** (Monasor 2010)[47]	**GSD Instructional Design Framework** (Damian 2006)[24]
Global Distance	Language and cultural distance Temporal distance Geographic distance	Start communication by brute force (Tip 1) Keep communication levels consistently high. (Tip 3)		
Teamwork	Management	Ensure that students keep the other site in mind (Tip 4) Be flexible – overcome the differences (Tip 7) Be alert (Tip 9)		Cross universities whole team
Curriculum/ pedagogy			Teaching of GSD must be supported by practical experiences through which students can learn by doing	Strategies for assessment of learning
Stakeholder/role	Organization	Be flexible – beat the administration (Tip 8)	Not possible for instructors to cover all stages and problems of GSD so any initiative should be focused on a specific field	Self-managed team negotiate scope
Infrastructure	Infrastructure		Appropriate selection of tools a key aspect	Wide range of tools
People/soft issues	Fear and Trust	Get the students to be familiar with each other as soon as possible (Tip 2) Keep the students highly motivated (Tip 5) Remember: we are different (Tip 6) Be enthusiastic (Tip 10)		Initial problem definition by client progressive clarification during lifecycle
Development process	Process Product architecture			Students alternate client & development role Project scope negotiated with client through iterations

The search itself posed several limitations. The initial search strings produced too many papers and false positives to be useful. The first IEEExplore database search, for example, produced over 40,000 hits. As a result, we had to narrow the search criteria and, may as a consequence have missed some relevant papers.

The inclusion/exclusion criteria that were used to filter papers pose another potential limit. Given the number of initial papers, the inclusion/exclusion filter had to be made quite tight to produce a manageable number of papers (see Section 3.8, Study Validation). In particular, studies centering on e-learning, studies concerning commercial GSD, and studies in books were not considered. These exclusions could have missed some important challenges or recommendations. The e-learning studies, for example, may have considered students in geographically dispersed locations working on projects in teams. Though not directly GSD, they could have produced useful data.

The extraction process also creates some limitations to this SLR. After verifying the extraction process itself (see Sections 3.5, 3.6 Data extraction and synthesis), data was extracted from each source by a single researcher. Again, given the quantity of papers it is possible that relevant data were missed in these extractions. As observed by Jalali and Wohlin [37] *We do not claim to have collected all relevant studies, but we included as many studies as possible. It should also be noted that although some studies may have been missed, there is no reason to believe that they would be distributed differently across the classifications than the papers included in the systematic review presented*.

Following the extraction of data, both challenges and recommendations were categorized into major and minor categories. The categories themselves were derived through the efforts of all the researchers and were filtered through four of the researchers. The categorization of data itself was examined by four researchers. As a result, we are quite confident that the categories and categorization are accurate and appropriate but, of course, a process this extensive leaves room for error.

Given the rigor and redundancy in our methods, we are confident that this study has produced comprehensive and accurate challenges and recommendations for GSE-Ed. Notwithstanding the many limitations detailed in this section, we have made efforts to provide enough detail in the form of derived themes and results, to allow other researchers and instructors to build on our findings. This is aided by our repeatable review process recorded in our protocol [6].

7. CONCLUSION

In this paper we set out through a comprehensive systematic literature review to answer two research questions:

RQ1: What are the challenges in delivering GSE courses to Software Engineering Students?

RQ2: What are the recommendations for delivering GSE courses to Software Engineering Students?

We identified seven major themes around which challenges and recommendations were grouped, namely: Global Distance; Teamwork; Curriculum/Pedagogy; Stakeholder/role;

Infrastructure; People/Soft issues; Development Process. A comprehensive and detailed set of challenges and associated recommendations have been outlined in this report.

Following a rigorous SLR process (described in Section 3) allowed us to thoroughly examine the issues and options available in the surprisingly rich GSD-Ed literature. We were able to draw on the wisdom of previous researchers as expressed in the 82 papers examined.

An overview summary of the key challenges and recommendations derived from our study was presented above in Table 21. The numbering in the table reflects the order of frequency – where those recommendations mentioned by the most studies come first. While this order does not necessarily reflect impact of the challenges or effectiveness of the solutions, it does suggest that the frequently noted challenges are likely to be faced by an instructor conducting a GSD-Ed course, and the frequently noted recommendations have worked across a range of contexts.

Teaching software engineering is difficult in a co-located setting, and as this review shows, teaching GSE-Ed courses comes with a considerable overhead, mainly due to distance issues. Teaching GSE-Ed in university settings is not for the faint hearted; yet, as the 82 studies in this SLR testify; many universities are doing just that. Universities in 39 different countries are collaborating across shores, in an effort to respond to the growing imperative of preparing their students for multi-site distributed development. These distributed development courses aim to give students first-hand experience of GSE in the hope to enhance their technical abilities, and at the same time teach them the importance of soft skills and teamwork.

This study has added to prior work in the area by consolidating insights from the diverse set of independent studies in GSE Education. It adds to knowledge gained through individual studies, by synthesizing a detailed set of identified challenges, accompanied by an actionable set of recommendations to address them. We hope it will prove a valuable reference source for educators seeking to enhance the quality of software engineering education through the design and implementation of successful GSE courses in fruitful global partnerships.

As Table 21 shows, each of the challenges raised in the studies we examined have clear recommendations that will help course designers prepare for GSE-Ed. If we were to distill these down to a single “key takeaway”, the message is: start preparing early for the course, know the level and experience of your cohort of students, and plan the development and tasks accordingly.

7.1 Future Work

While this study represents the results of an extensive review of the literature, there remain many unanswered questions. For example there appears to be little work on the individual student role in terms of recommendations as a stakeholder in the process, since most of the reviews focus on the instructor role, or how the student must interact as part of a team, or issues students have, with the exception of [29]; also there is not a great deal of work describing the client or customer role in GSE-Ed. Perhaps this is calling for more collaboration between universities and organizations, a familiar research question that has yet to be answered [9; 10]. From an external, cross-institution and intra-institutional linking aspect, the University representative role is underexplored. Also, although some work has started to assess a team’s general readiness for conducting GSE [7], an assessment for university’s readiness is also needed to take into account the differences identified in this review. Finally, research into strategies for successfully sustaining such courses on a long term basis is sorely needed.

ACKNOWLEDGEMENTS

We would like to thank the working group leaders for their support during this challenging process, and to the anonymous reviewers for giving their time and expertise to help us improve the final report.

This work was supported, in part, by Science Foundation Ireland grants 10/CE/I1855 and 13/RC/2094 to Lero - the Irish Software Research Centre (www.lero.ie), and by contract CF 2014 4348 from the European Regional Development Fund and Enterprise Ireland.

REFERENCES

[1] Ali-Babar, M. and Lescher, C., 2014. Editorial: Global software engineering: Identifying challenges is important and providing solutions is even better. *Information and Software Technology 56*, 1, 1-5.

[2] Ali, N., Beecham, S., and Mistrík, I., 2010. Architectural knowledge management in global software development: a review. In *IEEE International Conference on Global Software Engineering Workshops (ICGSEW)*. 347-352.

[3] Aspray, W., Mayadas, A.F., Vardi, M.Y., and Zweben, S.H., 2006. Educational Response to Offshore Outsourcing. In *Proceedings of the 37th SIGCSE technical symposium on Computer science education* ACM, Houston, Texas, USA, 330-331. DOI= http://dx.doi.org/10.1145/1121341.1121443.

[4] Aspray, W., Mayadas, F., and Vardi, M., 2006. *Globalization and Offshoring of Software - A Report of the ACM Job Migration task Force.* ACM.

[5] Beecham, S., Baddoo, N., Hall, T., Robinson, H., and Sharp, H., 2006. *Protocol for a Systematic Literature review of Motivation in Software Engineering*. Technical Report UH-CS-TR-453 Report.

[6] Beecham, S., Clear, T., Barr, J., and Noll, J., 2015. *Protocol for Challenges and Recommendations for the Design and Conduct of Global Software Engineering Courses: A Systematic Review*. (ITiCSE Working Group One: Technical Report No. Lero_TR_2015_01) Report.

[7] Beecham, S., Noll, J., and Richardson, I., 2015. Assessing the Strength of Global Teaming practices: A pilot study. In *Proceedings of the 10th IEEE International Conference on Global Software Engineering (ICGSE'15)* IEEE Computer Society, Castilla-la Mancha, Spain.

[8] Beecham, S., Noll, J., Richardson, I., and Ali, N., 2010. Crafting a global teaming model for architectural knowledge management. In *Proceedings of the 5th IEEE International Conference on Global Software Engineering (ICGSE'10*), Princeton, New Jersey, USA.

[9] Beecham, S., O'leary, P., Baker, S., Richardson, I., and Noll, J., 2014. *Making Software Engineering Research Relevant*. Computer 47, 4, 80-83. DOI= http://dx.doi.org/10.1109/mc.2014.92.

[10] Beecham, S., Oleary, P., Richardson, I., Baker, S., and Noll, J., 2013. Who are we doing Global Software Engineering research for? In *Proceedings of the 8th International Conference on Global Software Engineering (ICGSE'13)*, IEEE, 41-50.

[11] Bosnić, I., Čavrak, I., Orlić, M., Žagar, M., and Crnković, I., 2011. Student Motivation in Distributed Software Development Projects. In *Proceedings of the 2011 Community Building Workshop on Collaborative Teaching of Globally Distributed Software Development* ACM, 31-35.

[12] Boyer, E., 1990. *Scholarship Reconsidered: Priorities of the Professoriate*. Carnegie Foundation Special Report. Princeton University Press.

[13] Bruegge, B., Dutoit, A.H., Kobylinski, R., and Teubner, G., 2000. Transatlantic Project Courses in a University Environment. In *Seventh Asia-Pacific Software Engineering Conference, 2000. APSEC*, 30-37. DOI= http://dx.doi.org/10.1109/APSEC.2000.896680.

[14] Casey, V., 2010. Imparting the Importance of Culture to Global Software Development. *ACM Inroads* 1, 3, 51-57.

[15] Clear, T., 2002. E-Learning: A Vehicle for Transformation or Trojan Horse for Enterprise? - Revisiting the role of Public Higher Education Institutions. *International Journal on E-Learning* 1, 4 (October-December), 15 - 21.

[16] Clear, T., 2010. Exploring the Notion of 'Cultural Fit' in Global Virtual Collaborations. *ACM Inroads* 1, 3 (Sept), 58-65.

[17] Clear, T., 2012. Systematic Literature Reviews and Undergraduate Research. *ACM Inroads* 3, 4 (Dec), 10-11. DOI= http://dx.doi.org/10.1145/2381083.2381087.

[18] Clear, T., Claxton, G., Thompson, S., and Fincher, S., 2011. Cooperative and Work-Integrated Education in Information Technology. In *International Handbook for Cooperative & Work-Integrated Education*, R. Coll and K. Zegwaard Eds. World Association for Cooperative Education Inc, Lowell, MA, 141-150.

[19] Clear, T. and Kassabova, D., 2008. A Course in Collaborative Computing: Collaborative Learning and Research with a Global Perspective. In *Proceedings of the 39th ACM Technical Symposium on Computer Science Education*, M. Guzdial and S. Fitzgerald Eds. ACM, Portland, Oregon, 63-67.

[20] Clear, T., Young, F., Goldweber, M., Leidig, P., and Scott, K., 2001. ITiCSE 2001 Working Group Reports - Resources for Instructors of Capstone Courses in Computing. *SIGCSE Bulletin* 33, 93-113.

[21] Conchuir, E., Agerfalk, P., Olsson, H., and Fitzgerald, B., 2009. Global Software Development: Where Are The Benefits? *Commun. ACM* 52, 8, 127-131. DOI= http://dx.doi.org/10.1145/1536616.1536648.

[22] Cramton, C., 2001. The Mutual Knowledge Problem and its Consequences for Dispersed Collaboration. *Organization Science* 12, 3 (May-Jun), 346-371.

[23] Crnković, I., Bosnić, I., and Žagar, M., 2012. Ten Tips to Succeed in Global Software Engineering Education. In *Proceedings of the 34th International Conference on Software Engineering (ICSE)* IEEE Press, 1225-1234.

[24] Damian, D., Hadwin, A., and Al-Ani, B., 2006. Instructional Design and Assessment Strategies for Teaching Global Software Development: a Framework. In *Proceedings of the 28th International Conference on Software Engineering (ICSE)*, ACM, Shanghai, China, 685-690. DOI= http://dx.doi.org/10.1145/1134285.1134391.

[25] Daniels, M., Berglund, A., and Petre, M., 1999. Reflections on International Projects in Undergraduate CS Education. *Computer Science Education* 9, 3, 256-267.

[26] Daniels, M., Cajander, Å., Clear, T., and Mcdermott, R., 2015. Collaborative Technologies in Global Engineering: New Competencies and Challenges *International Journal of Engineering Education* 31, 1 (B), 267-281.

[27] Daniels, M., Cajander, Å., Pears, A., and Clear, T., 2010. Engineering Education Research in Practice: Evolving Use of Open Ended Group Projects as a Pedagogical Strategy for Developing Skills in Global Collaboration (Special issue on Applications of Engineering Education Research). *International Journal of Engineering Education* 26, 4, 795-806.

[28] Daniels, M., Petre, M., Almstrum, V., Asplund, L., Bjorkmann, C., Erickson, C., Klein, B., Last, M., and Berglund, A., 1998. RUNESTONE, an International Student Collaboration Project. In *IEEE Frontiers in Education Conference* IEEE, Tempe, Arizona.

[29] Filipovikj, P., Feljan, J., and Crnković, I., 2013. Ten Tips to Succeed in Global Software Engineering Education: What do the Students Say? In *3rd International Workshop on Collaborative Teaching of Globally Distributed Software Development (CTGDSD)*, 20-24. DOI= http://dx.doi.org/10.1109/CTGDSD.2013.6635241.

[30] Fortaleza, L.L., Conte, T., Marczak, S., and Prikladnicki, R., 2012. Towards a GSE International Teaching Network: Mapping Global Software Engineering Courses. *In Collaborative Teaching of Globally Distributed Software Development Workshop (CTGDSD)*, 1-5. DOI= http://dx.doi.org/10.1109/CTGDSD.2012.6226944.

[31] Gotel, O., Kulkarni, V., Say, M., Scharff, C., and Sunetnanta, T., 2009. A Global and Competition-Based Model for Fostering Technical and Soft Skills in Software Engineering Education. In *22nd Conference on Software Engineering Education and Training (CSEET)*, 271-278. DOI= http://dx.doi.org/10.1109/CSEET.2009.36.

[32] Hauer, A. and Daniels, M., 2008. A Learning Theory Perspective on Running Open Ended Group Projects (OEGPs). In *Conferences in Research and Practice in Information Technology*, Simon and M. Hamilton Eds. ACS, Wollongong, NSW, Australia, 85-92.

[33] Herrington, J., Oliver, R., and Reeves, T., 2002. Patterns of Engagement in Authentic Online Learning Environments. In *19th Annual Conference of the Australasian Society for Computers in Learning in Tertiary Education (ASCILITE) 2002*, A. Williamson, C. Gunn, A. Young and T. Clear Eds. UNITEC Institute of Technology, Auckland, New Zealand, Auckland, 279-286.

[34] Hettinga, M., 2002. Understanding Evolutionary Use of Groupware. In *Telematica Instituut* Delft University of Technology, Enschede, 191.

[35] Hirschheim, R., 2008. Some guidelines for the critical reviewing of conceptual papers. *Journal of the Association for Information Systems* 9, 8, 3.

[36] Holmström, H., Fitzgerald, B., Ågerfalk, P.J., and Conchúir, E.Ó., 2006. Agile practices reduce distance in global software development. *Information Systems Management* 23, 3, 7-18.

[37] Jalali, S. and Wohlin, C., 2012. Global Software Engineering and Agile Practices: A Systematic Review. *Journal of Software: Evolution and Process* 24, 6, 643-659.

[38] Jarvenpaa, S. and Leidner, D., 1999. Communication and Trust in Global Virtual Teams. *Organization Science* 10, 6, 791-815.

[39] Kitchenham, B. and Charters, S., 2007. Guidelines for Performing Systematic Literature Reviews in Software Engineering, version 2.3. In *EBSE Technical Report* Keele University, UK.

[40] Krippendorff, K., 1980. *Content Analysis: An Introduction to its Methodology*. Sage Publications, Beverly Hills.

[41] Lago, P., Muccini, H., and Babar, M.A., 2012. An Empirical Study of Learning by Osmosis in Global Software Engineering. *Journal of Software: Evolution and Process* 24, 6, 693-706.

[42] Le Blanc, R., Sobel, A., Ben-Menachem, M., Lethbridge, T., Díaz-Herrera, J., Hilburn, T., Mcgettrick, A., Atlee, J., Hawthorne, E., Leaney, J., Budgen, D., Matsumoto, Y., and Thompson, J., 2015. *Software Engineering 2014: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering*.

[43] Matthes, F., Neubert, C., Schulz, C., Lescher, C., Contreras, J., Laurini, R., Rumpler, B., Sol, D., and Warendorf, K., 2011. Teaching Global Software Engineering and International Project Management-Experiences and Lessons Learned from Four Academic Projects. In *CSEDU (2)*, 5-15.

[44] Meyer, B., 2001. Software Engineering in the Academy. *Computer* 34, 5, 28-35. DOI= http://dx.doi.org/10.1109/2.920608.

[45] Monasor, M.J., Vizcaíno, A., and Piattini, M., 2010. A Tool for Training Students And Engineers in Global Software Development Practices. In *Collaboration and Technology* Springer, 169-184.

[46] Monasor, M.J., Vizcaíno, A., and Piattini, M., 2012. Cultural and linguistic problems in GSD: a simulator to train engineers in these issues. *Journal of Software: Evolution and Process* 24, 6, 707-717.

[47] Monasor, M.J., Vizcaino, A., Piattini, M., and Caballero, I., 2010. Preparing Students and Engineers for Global Software Development: A Systematic Review. In *Proceedings of the 5th IEEE International Conference on Global Software Engineering (ICGSE'10)*, 177-186. DOI= http://dx.doi.org/10.1109/ICGSE.2010.28.

[48] Noll, J., Beecham, S., and Richardson, I., 2010. Global Software Development and Collaboration: Barriers and Solutions *ACM Inroads* 1, 3 (Sept), 66-78.

[49] Noll, J., Beecham, S., and Seichter, D., 2011. A Qualitative Study of Open Source Software Development: the OpenEMR Project. In *IEEE Empirical Software Engineering and Measurement Conference – ESEM 2011* September 19-23, Banff, Canada.

[50] Noll, J., Seichter, D., and Beecham, S., 2012. A Qualitative Method for Mining Open Source Software Repositories. In *Open Source Systems: Long-Term Sustainability* Springer Berlin Heidelberg, 256-261.

[51] Noll, J., Seichter, D., and Beecham, S., 2013. Can Automated Text Classification Improve Content Analysis of Software Project Data? In *ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, 2013. 300-303.

[52] Olson, G. and Olson, J., 2000. Distance Matters. *Human-Computer Interaction* 15, 139-178.

[53] Panteli, N. and Duncan, E., 2004. Trust and temporary virtual teams: alternative explanations and dramaturgical relationships. *Information Technology and People* 17, 4, 423-441.

[54] Pauleen, D., 2003. Leadership in a global virtual team: an action learning approach. *Leadership & Organization Development Journal* 24, 3, 153-162.

[55] Pears, A. and Daniels, M., 2010. Developing Global Teamwork Skills: The Runestone Project. In *Proceedings of the 2010 IEEE Education Engineering Conference (EDUCON 2010)*, IEEE.

[56] Peters, A., Hussain, W., Cajander, A., Clear, T., and Daniels, M., 2015. Preparing the Global Software Engineer. In *Proceedings of the IEEE 10th International Conference on Global Software Engineering (ICGSE'15)*, 61-70.

[57] Raza, B., Macdonell, S.G., and Clear, T., 2013. Research in Global Software Engineering: A Systematic Snapshot. In *Evaluation of Novel Approaches to Software Engineering*, J. Filipe and L. Maciaszek Eds. Springer Berlin Heidelberg, 126-140. DOI= http://dx.doi.org/10.1007/978-3-642-54092-9_9.

[58] Richardson, I., Casey, V., McCaffery, F., Burton, J., and Beecham, S., 2012. A process framework for global software engineering teams. *Information and Software Technology (IST)*, 54, 11, 1175-1191.

[59] Sandmann, L.R., 2008. Conceptualization of the Scholarship of Engagement in Higher Education: A Strategic Review, 1996–2006. *Journal of Higher Education Outreach and Engagement* 12, 1, 91-104.

[60] Schön, D., 1987. *Educating the Reflective Practitioner*. Jossey Bass, San Francisco.

[61] Swigger, K., Brazile, R., Harrington, B., Xiaobo, P., and Alpaslan, F., 2006. Teaching Students How to Work in Global Software Development Environments. In *International Conference on Collaborative Computing: Networking, Applications and Worksharing, CollaborateCom*, 1-7. DOI= http://dx.doi.org/10.1109/COLCOM.2006.361849.

[62] Tan, F. and Sutherland, P., 2004. Online Consumer Trust: A Multi-Dimensional Model. *Journal of Electronic Commerce in Organizations* 2, 3, 40-58.

[63] Valentine, D., 2004. CS Educational Research: A Meta-Analysis of SIGCSE Technical Symposium Proceedings. In *SIGCSE Technical Symposium Proceedings (SIGCSE'04*) ACM, Norfolk, VA, 255-259.

Appendix A: List of 82 Accepted Papers (as referenced in our Results Section)

[#1]: Almeida, E. *Teaching Globally Distributed Software Development: An Experience Report*. in *IEEE 25th Conference on Software Engineering Education and Training (CSEE&T)*. 17-19 April 2012. p. 105-109.

[#2]: Berkling, K., M. Geisser, T. Hildenbrand, and F. Rothlauf. *Offshore software development: transferring research findings into the classroom*, in *Software Engineering Approaches for Offshore and Outsourced Development* 2007, Springer. p. 1-18.

[#3]: Bosnić, I., I. Čavrak, M. Orlić, and M. Žagar. *Picking the right project: Assigning student teams in a GSD course*. In *IEEE 26th Conference on Software Engineering Education and Training (CSEE&T)*. 19-21 May 2013. p. 149-158.

[#4]: Bosnić, I., I. Čavrak, M. Orlić, M. Žagar, and I. Crnković. *Avoiding Scylla and Charybdis in distributed software development course*. In *Proceedings of the 2011 community building workshop on Collaborative teaching of globally distributed software development*. 2011. ACM. p. 26-30.

[#5]: Bosnić, I., I. Čavrak, M. Orlić, M. Žagar, and I. Crnković. *Student motivation in distributed software development projects*. In *Proceedings of the 2011 Community Building Workshop on Collaborative Teaching of Globally Distributed Software Development*. 2011, ACM. p. 31-35.

[#6]: Bosnić, I., I. Čavrak, M. Žagar, R. Land, and I. Crnković. *Customers' Role in Teaching Distributed Software Development*. In *23rd IEEE Conference on Software Engineering Education and Training (CSEE&T)*. 9-12 March 2010. p. 73-80.

[#7]: Bosnić, I., F. Ciccozzit, I. Čavrak, R. Mirandola, and M. Orlić. *Multi-dimensional assessment of risks in a distributed software development course*. In *3rd International Workshop on Collaborative Teaching of Globally Distributed Software Development (CTGDSD)*. 2013. IEEE. p. 6-10.

[#8]: Braun, A., A.H. Dutoit, A.G. Harrer, and B. Brugge. *IBistro: a learning environment for knowledge construction in distributed software engineering courses*. In *9th Asia-Pacific Software Engineering Conference*. 2002. p. 197-203.

[#9]: Brazile, R.P., K. Swigger, M.R. Hoyt, B. Lee, and B. Nelson. *A System to Support Teaching Global Software Development*. In *American Society for Engineering Education*. 2012. American Society for Engineering Education.

[#10]: Brooks, I. and K. Swigger. *The role of leadership and its effect on the temporal patterns of global software development teams*. In *8th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*. 14-17 Oct. 2012. p. 381-390.

[#11]: Bruegge, B., A.H. Dutoit, R. Kobylinski, and G. Teubner. *Transatlantic project courses in a university environment*. In *Seventh Asia-Pacific Software Engineering Conference, 2000. APSEC*. 2000. p. 30-37.

[#12]: Cao, L., H. Zhu, and G. Su. *Global Software Development Project*. In *18th Americas Conference on Information Systems, AMCIS 2012 Proceedings. Paper 15* July 29, 2012.

[#13]: Carlson, P. and X. Nan. *Experience and recommendations for distributed software development*. In *Collaborative Teaching of Globally Distributed Software Development Workshop (CTGDSD)*. 9 June 2012. p. 21-24.

[#14]: Casey, V. *Imparting the importance of culture to global software development.* ACM Inroads, 2010. **1**(3): p. 51-57.

[#15]: Cavrak, I., M. Orlic, and I. Crnkovic. *Collaboration patterns in distributed software development projects*. In *34th International Conference on Software Engineering (ICSE)* 2012. 2-9 June 2012. p. 1235-1244.

[#16]: Clear, T. *Replicating an 'Onshore' Capstone Computing Project in a 'Farshore' Setting--An Experience Report*. In *6th IEEE International Conference on Global Software Engineering (ICGSE)* 2011. IEEE. p. 161-165.

[#17]: Clear, T. and M. Daniels. *Using groupware for international collaborative learning*. In *30th Annual Frontiers in Education Conference, 2000. FIE*. 2000. 2000. p. F1C/18-F1C/23 vol.1.

[#18]: Clear, T., J. Whalley, J. Hill, Y. Liu, A. Pears, and B. Plimmer. *A global software project: Developing a tablet PC capture platform for explanograms*. In *Proceedings of the 8th International Conference on Computing Education Research*. 2008. ACM. p. 41-50.

[#19]: Crnković, I., I. Bosnić, and M. Žagar. *Ten tips to succeed in global software engineering education*. In *Proceedings of the 34th International Conference on Software Engineering (ICSE)*. 2012. IEEE Press. p. 1225-1234.

[#20]: Dastidar, S.G. and S. Chatterjee. *Distributed software development: Experience and recommendation*. In *3rd International Workshop on Collaborative Teaching of Globally Distributed Software Development (CTGDSD)*. 2013. 25-25 May 2013. p. 11-14.

[#21]: Deiters, C. *GloSE-Lab: Teaching Global Software Engineering*. In *6th IEEE International Conference on. Global Software Engineering (ICGSE'11)*, 15-18 Aug. 2011. p. 156-160.

[#22]: Doerry, E., R. Klempous, J. Nikodem, and W. Paetzold. *Virtual student exchange: lessons learned in virtual international teaming in interdisciplinary design education*. In *Proceedings of the 5th International Conference on Information Technology Based Higher Education and Training, ITHET*. 31 May-2 June 2004. p. 650-655.

[#23]: Ende, M., R. Lammermann, P. Brockmann, and G. Ayurzana. *A virtual, global classroom to teach global software engineering: A Mongolian-German team-teaching project*. In *Second International Conference on e-Learning and e-Technologies in Education (ICEEE)*. 23-25 Sept. 2013. p. 229-233.

[#24]: Estler, H.C., M. Nordio, C.A. Furia, and B. Meyer. *Awareness and Merge Conflicts in Distributed Software Development*. In *IEEE 9th International Conference on Global Software Engineering (ICGSE)*. 18-21 Aug. 2014. p. 26-35.

[#25]: Fagerholm, F., P. Johnson, A. Sanchez Guinea, J. Borenstein, and J. Munch. *Onboarding in Open Source Software Projects: A Preliminary Analysis*. In *IEEE 8th International Conference on Global Software Engineering Workshops (ICGSEW)*. 26-26 Aug. 2013. p. 5-10.

[#26]: Fagerholm, F., N. Oza, and J. Munch. *A platform for teaching applied distributed software development: The ongoing journey of the Helsinki software factory*. In *3rd International Workshop on Collaborative Teaching of Globally Distributed Software Development (CTGDSD)*. 25 May 2013. p. 1-5.

[#27]: Farley, A., S. Faulk, V. Lo, A. Proskurowski, and M. Young. *Intensive international Summer Schools in Global Distributed Software Development*. In *Frontiers in Education Conference (FIE)*. 3-6 Oct. 2012. p. 1-6.

[#28]: Favela, J. and F. Pena-Mora. *An experience in collaborative software engineering education.* IEEE Software, 2001. 18(2): p. 47-53.

[#29]: Feljan, J., I. Bosnić, M. Orlić, and M. Žagar. *Distributed Software Development course: Students' and teachers' perspectives*. In *Collaborative Teaching of Globally Distributed Software Development Workshop (CTGDSD)*. 9 June 2012. p. 16-20.

[#30]: Filipovikj, P., J. Feljan, and I. Crnković. *Ten tips to succeed in global software engineering education: What do the students say?* In *3rd International Workshop on Collaborative Teaching of Globally Distributed Software Development (CTGDSD)*. 25-25 May 2013. p. 20-24.

[#31]: Giraldo, F., S.F. Ochoa, L. Aballay, C. Clunie, A. Neyem, and R. Anaya. *Supporting Instructional Software Engineering Activities Using CODILA: Some Latin American Experiences*, In *Education and Educational Technology* 2012, Springer. p. 591-598.

[#32]: Giraldo, F.D. *Applying a distributed CSCL activity for teaching software architecture*. In *2011 International Conference on Information Society (i-Society)*. 27-29 June 2011. p. 208-214.

[#33]: Gloor, P., M. Paasivaara, C. Lassenius, D. Schoder, K. Fischbach, and C. Miller. *Teaching a global project course: experiences and lessons learned*, In *Proceedings of the 2011 Community Building Workshop on Collaborative Teaching of Globally Distributed Software Development* 2011, ACM: Waikiki, Honolulu, HI, USA. p. 1-5.

[#34]: Gotel, O., V. Kulkarni, M. Say, C. Scharff, and T. Sunetnanta. *A Global and Competition-Based Model for Fostering Technical and Soft Skills in Software Engineering Education*. In *22nd Conference on Software Engineering Education and Training (CSEET)*. 17-20 Feb. 2009. p. 271-278.

[#35]: Gotel, O., V. Kulkarni, M. Say, C. Scharff, and T. Sunetnanta. *Quality Indicators on Global Software Development Projects: Does "Getting to Know You" Really Matter?* In *4th IEEE International Conference on Global Software Engineering (ICGSE'09)*. 13-16 July 2009. p. 3-7.

[36]: Gotel, O., V. Kulkarni, C. Scharff, and L. Neak. *Students as partners and students as mentors: an educational model for quality assurance in global software development*, In *Software Engineering Approaches for Offshore and Outsourced Development* 2009, Springer. p. 90-106.

[#37]: Gotel, O., C. Scharff, and V. Kulkarni. *Mixing continents, competences and roles: Five years of lessons for software engineering education.* IET Software, 2012. 6(3): p. 199-213.

[#38]: Gotel, O., C. Scharff, and S. Seng. *Preparing Computer Science Students for Global Software Development*. In *36th Annual Frontiers in Education Conference*. 27-31 Oct. 2006. p. 9-14.

[#39]: Honig, W.L. and T. Prasad. *A classroom outsourcing experience for software engineering learning*. In *ACM SIGCSE Bulletin*. 2007. ACM. p. 181-185.

[#40]: Inkeri Verkamo, A., J. Taina, Y. Bogoyavlenskiy, D. Korzun, and T. Tuohiniemi. *Distributed Cross-Cultural Student Software Project: A Case Study*. In *18th Conference on Software Engineering Education & Training*. 18-20 April 2005. p. 207-214.

[#41]: Junhua, D. *A framework for global collaboration in teaching software engineering*. In *3rd International Workshop on Collaborative Teaching of Globally Distributed Software Development (CTGDSD)*. 25 May 2013. p. 30-34.

[#42]: Junhua, D. and Y. Biwu. *Teaching software engineering with Global Understanding*. In *Collaborative Teaching of Globally Distributed Software Development Workshop (CTGDSD)*. 9 June 2012. p. 11-15.

[#43]: Keenan, E. and A. Steele. *Developing a pedagogical infrastructure for teaching globally distributed software development*. In *Proceedings of the 2011 Community Building Workshop on Collaborative Teaching of Globally Distributed Software Development*. 2011, ACM. p. 6-10.

[#44]: Keenan, E., A. Steele, and X. Jia. *Simulating Global Software Development in a Course Environment*. In *5th IEEE International Conference on Global Software Engineering (ICGSE)*. 2010, IEEE. p. 201-205.

[#45]: Lago, P., H. Muccini, and M.A. Babar. *Developing a Course on Designing Software in Globally Distributed Teams*. In *IEEE International Conference on Global Software Engineering (ICGSE'08)*. 17-20 Aug. 2008. p. 249-253.

[#46]: Lago, P., H. Muccini, and M.A. Babar. *An empirical study of learning by osmosis in global software engineering.* Journal of Software: Evolution and Process, 2012. 24(6): p. 693-706.

[#47]: Lago, P., H. Muccini, L. Beus-Dukic, I. Crnkovic, S. Punnekkat, and H. Van Vliet. *Towards a European Master Programme on Global Software Engineering*. In *20th Conference on Software Engineering Education & Training (CSEET '07)*. 3-5 July 2007. p. 184-194.

[#48]: Last, M.Z. *Understanding the group development process in global software teams*. In *33rd Annual Frontiers in Education (FIE)*. 5-8 Nov. 2003. p. S1F-20-5 vol.3.

[#49]: Lescher, C., L. Yang, and B. Bruegge. *Teaching Global Software Engineering: Interactive Exercises for the Classroom*. In *IEEE 9th International Conference on Global Software Engineering (ICGSE'14)*. 18-21 Aug. 2014. p. 163-172.

[#50]: Long, J. *Outsourcing in Next Generation Software Engineering Technology Education*. In *American Society for Engineering Education*. 2010. American Society for Engineering Education.

[#51]: Mäkiö, J. and S. Betz. *On educating globally distributed software development—A case study*. In *24th International Symposium on Computer and Information Sciences (ISCIS)*. 2009. IEEE. p. 480-485.

[#52]: Matthes, F. *Teaching Global Software Engineering and International Project Management-Experiences and Lessons Learned from Four Academic Projects*. In *CSEDU (2)*. 2011. p. 5-15.

[#53]: McDermott, R., J. Bass, and J. Lalchandani. *The learner experience of student-led international group project work in software engineering*. In *IEEE Frontiers in Education Conference (FIE)*. 22-25 Oct. 2014. p. 1-8.

[#54]: Monasor, M.J., A. Vizcaino, and M. Piattini. *VENTURE: Towards a framework for simulating GSD in educational environments*. In *5th International Conference on Research Challenges in Information Science (RCIS)*. 19-21 May 2011. p. 1-10.

[#55]: Monasor, M.J., A. Vizcaíno, and M. Piattini. *A tool for training students and engineers in global software development practices*, In *Collaboration and Technology* 2010, Springer. p. 169-184.

[#56]: Monasor, M.J., A. Vizcaíno, and M. Piattini. *Providing training in GSD by using a virtual environment*, In *Product-Focused Software Process Improvement (Profes)* 2012, Springer. p. 203-217.

[#57]: Monasor, M.J., A. Vizcaíno, M. Piattini, J. Noll, and S. Beecham. *Assessment process for a simulation-based training environment in global software development*. In *Proceedings of the 2014 conference on Innovation & Technology in Computer Science Education (ITiCSE)*. 2014, ACM. p. 231-236.

[#58]: Neto, C.R.L. and E.S. De Almeida. *Five years of lessons learned from the software engineering course: adapting best practices for distributed software development*. In *Proceedings of the Second International Workshop on Collaborative Teaching of Globally Distributed Software Development*. 2012, IEEE Press. p. 6-10.

[#59]: Noll, J., A. Butterfield, K. Farrell, T. Mason, M. McGuire, and R. McKinley. *GSD Sim: A Global Software Development Game*. In *IEEE International Conference on Global Software Engineering Workshops (ICGSEW)*. 18 Aug. 2014. p. 15-20.

[#60]: Nordio, M., H.C. Estler, B. Meyer, N. Aguirre, R. Prikladnicki, E. di Nitto, and A. Savidis. *An experiment on teaching coordination in a globally distributed software engineering class*. In *IEEE 27th Conference on Software Engineering Education and Training (CSEE&T)*. 23-25 April 2014. p. 109-118.

[#61]: Nordio, M., H.C. Estler, B. Meyer, J. Tschannen, C. Ghezzi, and E. di Nitto. *How Do Distribution and Time Zones Affect Software Development? A Case Study on Communication*. In *6th IEEE International Conference on Global Software Engineering (ICGSE'11)*. 15-18 Aug. 2011. p. 176-184.

[#62]: Nordio, M. *Teaching software engineering using globally distributed projects: the DOSE course*. In *Proceedings of the 2011 Community Building Workshop on Collaborative Teaching of Globally Distributed Software Development*. 2011, ACM. p. 36-40.

[#63]: Nordio, M., R. Mitin, and B. Meyer. *Advanced hands-on training for distributed and outsourced software engineering*. In *ACM/IEEE 32nd International Conference on Software Engineering (ICSE)*. 2-8 May 2010. p. 555-558.

[#64]: Paasivaara, M., C. Lassenius, D. Damian, P. Raty, and A. Schroter. *Teaching students global software engineering skills using distributed Scrum*. In *35th International Conference on Software Engineering (ICSE)*. 18-26 May 2013. p. 1128-1137.

[#65]: Peña-Mora, F., R. Struminger, J. Favela, and R. Losey. *Supporting a Project-Based, Collaborative, Distance Learning Lab*. In *Computing in Civil and Building Engineering*. 2000. ASCE. p. 170-176.

[#66]: Petkovic, D., G.D. Thompson, and R. Todtenhoefer. *Assessment and comparison of local and global SW engineering practices in a classroom setting.* ACM SIGCSE Bulletin, 2008. **40**(3): p. 78-82.

[#67]: Petkovic, D., R. Todtenhoefer, and G. Thompson. *Teaching Practical Software Engineering and Global Software Engineering: Case Study and Recommendations*. In *36th Annual Frontiers in Education Conference*. 27-31 Oct. 2006. p. 19-24.

[#68]: Richardson, I., A.E. Milewski, N. Mullick, and P. Keil. *Distributed development: an education perspective on the global studio project*. In *Proceedings of the 28th international conference on software engineering*. 2006, ACM. p. 679-684.

[#69]: Richardson, I., S. Moore, D. Paulish, V. Casey, and D. Zage. *Globalizing Software Development in the Local Classroom*. In *20th Conference on Software Engineering Education & Training, CSEET '07*. 3-5 July 2007. p. 64-71.

[#70]: Romero, M., A. Vizcaino, and M. Piattini. *Teaching Requirements Elicitation within the Context of Global Software Development*. In *Mexican International Conference on Computer Science (ENC)*. 21-25 Sept. 2009. p. 232-239.

[#71]: Romero, M., A. Vizcaíno, and M. Piattini. *Developing the Skills Needed for Requirement Elicitation in Global Software Development*. In *ICEIS (1)*. 2008. p. 393-396.

[#72]: Serce, F.C., F.N. Alpaslan, K. Swigger, R. Brazile, G. Dafoulas, and V. Lopez. *Strategies and guidelines for building effective distributed learning teams in higher education*. In *9th International Conference on Information Technology Based Higher Education and Training (ITHET)*. April 29 -May 1 2010. p. 247-253.

[#73]: Shata, O. *A Crash Undergraduate Course in Global Software Engineering*. In *12th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*. 6-8 July 2011. p. 213-218.

[#74]: Stroulia, E., K. Bauer, M. Craig, K. Reid, and G. Wilson. *Teaching distributed software engineering with UCOSP: the undergraduate capstone open-source project*. In *Proceedings of the 2011 community building workshop on Collaborative teaching of globally distributed software development*. 2011, ACM. p. 20-25.

[#75]: Swigger, K., R. Brazile, B. Harrington, S. Peng, and F. Alpaslan. *A case study of student software teams using computer-supported software*. In *Proceedings of the 2005 International Symposium on Collaborative Technologies and Systems*. 20 May 2005. p. 167-173.

[#76]: Swigger, K., R. Brazile, B. Harrington, P. Xiaobo, and F. Alpaslan. *Teaching Students How to Work in Global Software Development Environments*. In *International Conference on Collaborative Computing: Networking, Applications and Worksharing, CollaborateCom*. 17-20 Nov. 2006. p. 1-7.

[#77]: Swigger, K., R. Brazile, F.C. Serce, G. Dafoulas, F.N. Alpaslan, and V. Lopez. *The Challenges of Teaching Students How to Work in Global Software Teams*. In *IEEE Transforming Engineering Education: Creating Interdisciplinary Skills for Complex Global Environments*. 6-9 April 2010. p. 1-30.

[#78]: Swigger, K., M. Hoyt, F.C. Serçe, V. Lopez, and F.N. Alpaslan. *The temporal communication behaviors of global software development student teams.* Computers in Human Behavior, 2012. 28(2): p. 384-392.

[#79]: Swigger, K., F.C. Serce, F.N. Alpaslan, R. Brazile, G. Dafoulas, and V. Lopez. *A Comparison of Team Performance Measures for Global Software Development Student Teams*. In *4th IEEE International Conference on Global Software Engineering (ICGSE'09)*. 13-16 July 2009. p. 267-274.

[#80]: Swigger, K., F.C. Serce, G. Dafoulas, F.N. Alpaslan, and V. Lopez. *When do distributed student teams work?* In *International Conference on Information Technology Based Higher Education and Training (ITHET)*. 2012. 21-23 June 2012. p. 1-8.

[#81]: Van Solingen, R., K. Dullemond, and B. van Gameren. *Evaluating the Effectiveness of Board Game Usage to Teach GSE Dynamics*. In *6th IEEE International Conference on Global Software Engineering (ICGSE'11)*. 15-18 Aug. 2011. p. 166-175.

[#82]: Woit, D. and K. Bell. *Student communication challenges in distributed software engineering environments*. In *ACM SIGCSE Bulletin*. 2005. ACM. p. 286-290.

Appendix B: Valentine's Taxonomy of Study Types [63] :

In this paper Valentine proposes a six-fold taxonomy to classify the type of articles found in CS Education Research where the usual requirements of an explicit research question, conveyed in a series of hypotheses, tested with a variety of experimental and control groups, with a strict statistical analysis of results are relaxed. Valentine suggests that we do not need such a strictly quantified, statistical model to prove significant educational results. As a result, the set *"as inclusive (and yet reasonable) a bar as possible for this category*" and settled on a very simple rubric:

"**Experimental**": Studies fall into this category if the author makes any attempt at assessing the "treatment" with some scientific analysis. A minimal example would be a study that shows that after a new Breadth-First CS1 course, the number of CS majors earning a 'C' or better in CS2 doubles. A study at the other end of the category would do a complete statistical analysis. A study of 500 introductory students at two institutions to show the impact of math background and prior programming to success in CS1 that uses strict statistical methodology would be an example of this end of the category. Another, less quantitative example (because not all knowledge is quantifiable) would be a study that, through a series of interviews, develops an ethnography of how students develop their own (often faulty) cognitive rules about parameter passing. Another qualitative example would be a philosophical discussion of pedagogy that does a review of existing research literature. Please note that this is a preemptive category, so if a presentation fits here and somewhere else (e.g. a quantified assessment of some new Tool), it would be placed here.

"**Marco Polo**": A "Marco Polo" study basically says "I went there and I saw this." SIGCSE veterans recognize this as a staple at the Symposium. Colleagues describe how their institution has tried a new curriculum, adopted a new language or put up a new course. The reasoning is defined, the component parts are explained, and then (and this is the giveaway for this category) a conclusion is drawn like "Overall, I believe the [topic] has been a big success." or "Students seemed to really enjoy the new [topic]". Now, Marco Polo presentations serve an important function: we are a community of educators and sharing our successes (and failures) enriches the whole community.

"**Philosophy**": This type of study occurs when the author has made an attempt to generate debate of an issue, on philosophical grounds, among the broader community. An example here would be a panel discussion on a topic such as "Integrating Empirical Methods into CS" which is designed to promote discussion within the traditional computer science community. Or it could include an article that tries to stimulate the core language debate along philosophical and educational lines.

"**Tools"**: Among many other things, colleagues have developed software to animate algorithms, to help grade student programs, to teach recursion, and to provide introductory development platforms. This category encompasses papers that discuss such tools and their use. For example, a study might explain a tool that allows novice programmers to use pictograms rather than syntax to create programs or a tool that graphically represents linked data structures for students. Not all tools are software; an author could present a paradigm or an organizing rubric to be a tool for an entire course. A visual design tree and data flow diagram, for example, could be used as an effective teaching tool for CS1.

"**Nifty**": This, the most whimsical category, is taken from the panels of the same name offered at conferences. Nifty assignments, projects, puzzles, games and paradigms are the bubbles in the champagne of SIGCSE. Most of us seem to appreciate innovative, interesting ways to teach students our abstract concepts. Sometimes the difference between Nifty and Tools is fuzzy, but generally a Tool would be used over the course of a semester, and a Nifty assignment was more limited in duration.

"**John Henry**": Every now and then a colleague will describe a course that seems so outrageously difficult that one suspects it is telling us more about the author than it is about the pedagogy of the class. For example, it is possible to teach CS1 as a predicate logic course in IBM 360 assembler – but why would you want to do that? Yes, every once in a while somebody can beat the steam engine, but most of us try to avoid that situation. John Henry's can, however, provide valuable insight into the limits of CS pedagogy.

Note: *this appendix is extracted largely verbatim from Valentine* [63].

Appendix C: Search Terms and Strings

Search terms were derived from the research questions by identifying the population, intervention and outcome and then identifying alternative spellings and synonyms for the terms. The Boolean OR was used to incorporate alternative spellings and synonyms and the Boolean AND was used to link the major terms from population, intervention and outcome.

The same search string was used for the IEEEXplore database and the Scopus database. For the ACM Digital Library the search string was modified to account for syntactical differences in the query language. Table C1 gives an example of a nested Search String as used in the IEEEXplore database. The look-up table can be used to check the precise terms used and years included for each recorded paper. We stored as much information as possible about each paper in our Summary Spreadsheet and accompanying Endnote file. This search yielded 545 papers.

Table C1: IEEEXplore SEARCH TERMS LOOKUP TABLE – 14 June 2015

Date	**Search string**	Comments
	Used Command search and refined by Content Type: Conference Publications Journals & Magazines Year: 2000-2015	IEEEXplore had a limit to number of terms I could use
14 June 2015	**((((software OR "information technology" OR "information system*" OR comput* OR programming) AND (student OR trainee OR learner)) AND ("distributed software" OR "global software") AND (educat* OR train* OR course)))**	Inclusive search: Applies to both RQ1 AND RQ2 – did not limit the papers by including BOOLEAN 'AND' for challenges (RQ1) and recommendations (RQ2).

The look-up table in Table C2 shows the search strings used for the ACM digital library. In the ACM digital library abstracts and titles had to be searched separately. The abstract search yielded 41 papers and the title search yielded 16 papers.

Table C2: ACM digital library SEARCH TERMS LOOKUP TABLE – 16 June 2015

Date	**Search string**	Comments
	Used the query box provided in the Advanced Search option	Due to the constraints of the advanced search option, two queries were performed, one to search abstracts and one to search titles.
16 June 2015	**(Abstract:software or Abstract:programming or Abstract:comput or Abstract:"information technology or information system") and (Abstract:student or Abstract:learner or Abstract:trainee) and (Abstract:"distributed software" or Abstract:"global software") and (Abstract:educat or Abstract:train or Abstract:course)**	Inclusive Abstract search: Applies to both RQ1 AND RQ2 – did not limit the papers by including BOOLEAN 'AND' for challenges (RQ1) and recommendations (RQ2).
16 June 2015	**(Title:software or Title:"information technology" or Title:"information system" or Title:comput* or Title:programming) and (Title:student or Title:trainee or Title:learner) and (Title:"distributed software" or Title:"global software")**	Inclusive Title search: Applies to both RQ1 AND RQ2 – did not limit the papers by including BOOLEAN 'AND' for challenges (RQ1) and recommendations (RQ2). Did not include the restrictions that "educat*", "train" or "course" had to be in the title.

Appendix D: First Categorisation Exercise (30 papers)

Table D1: First Categorisation Exercise (30 papers)

No.	Minor Categories	Type categories
1	failure to support educational objective	Client
2	barriers to synchronous	communication
3	conflict resolution time grows with no of instructors do	communication
4	language related	communication
5	negotiation & accountability to other team members	communication
6	Additional time commitment	communication
7	awareness	communication
8	synchronization	communication
9	overhead due to global distance	communication
10	lack of face to face and non-verbal interactions	communication
11	infrastructure level	conflict management
12	infrastructure/institutional level	conflict management
13	student level	conflict management
14	early/timely conflict detection	conflict management
15	communication style	cultural differences
16	impact on communication	cultural differences
17	impact on motivation	cultural differences
18	institutional	cultural differences
19	in student behaviour and norms	cultural differences
20	motivational aspects	cultural differences
21	students	cultural differences
22	course design incompatibilities	curriculum
23	delivery techniques	curriculum
24	content planning	curriculum
25	different skills & education because of curricular timing	curriculum
26	emerging discipline	curriculum
27	authentic experience	curriculum
28	GSE-Ed immaturity	curriculum
29	soft skills	curriculum
30	module interfaces	development process
31	requirements elicitation	development process
32	mentor the students	development process
33	code comprehension	development process
34	software testing	development process
35	software design	development process
36	software integration	development process
37	Nearshore	Geographical distance
38	gsd course mgt/scheduling support tools	infrastructure
39	gsd team/project admin support tools	infrastructure
40	gsd communication support tools	infrastructure
41	config/rationale mgt/version control	infrastructure
42	tool framework	infrastructure
43	environments/technology platforms	infrastructure
44	technical problems	infrastructure
45	technical support provision	infrastructure
46	tool mismatch	infrastructure
47	gsd tailored tools	infrastructure
48	gsd knowledge mgt tools	infrastructure
49	tool standardization	infrastructure
50	tool preference conflicts	infrastructure
51	technical problems	infrastructure
52	lingua franca as a second language	language differences
53	Performance	language differences
54	power distance	language differences
55	identical terms different meaning	language differences
56	performance outcomes	motivation
57	student commitment impacts	motivation
58	imagination stimulation	motivation
59	cultural norms	motivation/focus
60	learning versus grade driven?	motivation/focus
61	student maturity/willingness to engage.	motivation/focus
62	need for adaptability	management: instructor
63	Workload	management: instructor
64	class management	management: instructor
65	workload balancing immediate vs. management oversight	management: instructor
66	logistical/administration issues	management: instructor
67	student frustration because of lack of predictability	management: student
68	time management	management: student
69	project management	management: student
70	Workload	management: student
71	collaboration patterns	orchestration

Table D1 (Continued): First Categorisation Exercise (30 papers)

72	combination people, processes and tools	orchestration
73	need for adaptability	organization
74	allocation to teams	organization
75	governance	organization
76	theoretical framework	organization
77	institution: standards	organization
78	resource coordination	organization
79	project selection	organization
80	sustained funding	organization
81	institution: regulation	organization
82	institution: scheduling	organization
83	student	organization
84	legal constraints	organization
85	synchronization	organization
86	assessment	pedagogy
87	simulation strategy	pedagogy
88	coaching	pedagogy
89	need for adaptability	pedagogy
90	retaining focus on learning	pedagogy
91	GSE-Ed immaturity	pedagogy
92	Scaffolding	pedagogy
93	team leadership performance	teamwork
94	thinking beyond those you see	teamwork
95	lack of shared vision	teamwork
96	roles and responsibilities	teamwork
97	time zone differences	time
98	Synchronization	time
99	Culture	time
100	[event (predictable), cyclical]	time
101	a small number of tasks performed	time
102	Delay	time
103	risk management by students	trust
104	[no sub-category]	trust
105	negative group cohesion impacts	trust
106	negative task assignments	trust
107	performance based	trust
108	team building	trust
109	unfair treatment/preferential treatment	trust
110	expertise imbalances	trust

Appendix E: 7 Key Themes Defined

Table E1: 7 Key Themes Defined

ID	Major Classification	Minor Classification	Minor String	Definition
GSE-ED_1	Stakeholder/Role	General	Any group involved in the delivery of the GSD course	Includes: Student; Tutor; Client; University Management.
GSE-ED_1.1	Stakeholder/Role	Client	Client; Proxy Client; Product Owner; External client.	Stakeholder in development process who provided the requirements.
GSE-ED_1.2	Stakeholder/Role	Instructor	Instructor; Tutor; project manager, supervisor	Tutor role in GSE-Ed, e.g. Instructors should assist in architectural design and partitioning for independent development of modules.
GSE-ED_1.3	Stakeholder/Role	Student	Student; trainee; learner; student role in the process	Student role (e.g. becoming team leader). Student skills. Student visibility
GSE-ED_1.4	Stakeholder/Role	University representative	University Managers; course leaders; departments.	Funding; buy-in; resource allocation (e.g. classrooms, new tools, servers)
GSE-ED_1.5	Stakeholder/Role	role conflict	Conflicting Roles; need for multi-role; role confusion	Teaching needs merged with motivation to do research (different goals?)
GSE-ED_2	Global Distance	Increased complexity	General differences between groups operating in the same team as viewed in terms of (Cultural; Temporal; Linguistic; Geographic; Organisational, Institutional, Managerial, Student Skill) - Scaling (in communication), levels of communication	Often manifested as increased complexity, communication overhead, and process scaling (to address complexity), lack of informal communication
GSE-ED_2.1	Global Distance	Cultural	Cultural	Communication styles, cultural norms. Ethnic and Religious differences, treatment of gender; culture associated with different institutions or organisations, ethics come under this banner
GSE-ED_2.2	Global Distance	Temporal	Temporal (e.g. synchronisation) or Time related, e.g. delays	Synchronous or synchronous forms of communication, any reference made to time in a GSD context, including time pressure.
GSE-ED_2.3	Global Distance	Linguistic	Language related	Teams may need to communicate in their second language.
GSE-ED_2.4	Global Distance	General	Geographic	Impact geographic distance has on communication
GSE-ED_2.5	Global Distance	Organisational	Organisational/institutional/managerial	This focusses on 'Management' and Synchronisation. This includes distance caused by any mismatch across universities in they way they operate. e.g. term/semester length, course content; Goals mismatch (e.g. commitment levels may vary as a result); Conflict detection; conflict resolution, conflict management, governance. includes open source community differences
GSE-ED_2.6	Global Distance	Skills	Student Skills	Different Skills taught at different universities, different abilities, different course focus, different length of course, immersive.

Table E1: 7 Key Themes Defined (Cont'd)

ID	Major Classif.	Minor Classif	Minor String	Definition
GSE-ED_3	Teamwork/ team creation	General	Negotiation and accountability to other team members	How team spirit is generated across universities and countries; responsibilities; roles. Skills match
GSE-ED_3.1	Teamwork/team creation	synergy	Pulling together, respect, synergy, support, knowledge sharing, tolerance, integration, collaboration; accountability	Understanding strengths and weaknesses, taking personal responsibility for own part in team; shared vision
GSE-ED_3.2	Teamwork/team creation	task allocation	Creating a balanced team, project management, project selection, decision making, task allocation, coordination	Matching skills at the outset at project kick-off. Allocating tasks across the SDLC
GSE-ED_4	Curriculum/ped agogy	General	Course design; delivery techniques; content planning; skill levels of student; emergent discipline; GSD maturity; soft skills training, course content.	Course objectives, grading and assessment schemes; course feedback; learning outcomes, rewards for certain behaviour
GSE-ED_4.1	Curriculum/ped agogy	Course design	Course design, planning, grading, moderation, assessment, course delivery capabilities and techniques	Course design compatibilties
GSE-ED_4.2	Curriculum/ped agogy	Learning Outcomes	Learning outcomes. Value added, skills enhancement.	Retain focus on learning; need for adaptability, tap into prior learning
GSE-ED_5	Development Process	General	Development architecture (module interfaces); requirements engineering; interfaces; testing; handovers, coding; build, implementation. Scaling.	Any activity across the software development lifecycle; and, type of process used (e.g. agile/plan driven).
GSE-ED_5.1	Development Process	Requirements	SDLC: Requirements engineering	Requirements elicitation; negotiation, validation, management.
GSE-ED_5.2	Development Process	Software Development Process	Plan drive/ Agile/ Scrum, etc.	
GSE-ED_5.3	Development Process	System/code integration	Implementation/integration	
GSE-ED_5.4	Development Process	Design	SDLC: Architecture/Design	Students lack experience to partition a software architecture appropriately to allow independent modules.
GSE-ED_5.5	Development Process	Testing		
GSE-ED_5.6	Development Process	Coding		
GSE-ED_6	Infrastructure	General	Anything relating to the external development environment	Development tools, platforms, technical environment; www, repositories, security, programming languages
GSE-ED_6.1	Infrastructure	Tools	Tools	tool support, including type of programming language used, communication tools, wikis
GSE-ED_6.2	Infrastructure	Technical issues	Technical/platforms	The development environment
GSE-ED_6.3	Infrastructure	Version Control	Version Control	SVN, GIT hub, etc
GSE-ED_7	People/soft Issues	General	Non-technical or organisation issues concerning people management	Funding; buy-in; resource allocation (e.g. classrooms, new tools, servers)
GSE-ED_7.1	People/soft Issues	Motivation	Motivation	Personalities, individual focus/needs/drive/ enthusiasm, equity (equal treatment across teams), motivation in different cultures
GSE-ED_7.2	People/soft Issues	Trust	Trust	Willingness to share problems and share experience and knowledge and work. Assumptions, pre-conceptions, tolerance of diversity
GSE-ED_7.3	People/soft Issues	Stress	Stress and Pressure	

Appendix F: Data Extraction Form (Phase 1 and 2)

FIELDS TO COMPLETE (PHASE 1)	**Your Response**	**Comments**
Paper ID:		Use identifier from master /accepted papers list e.g. IEEE_1 or ACM_1 etc.
Paper Title		First few words will suffice
Researcher Name		Your name
Date researcher analysed this paper:		When you completed this form
EXCLUSION/INCLUSION CRITERIA		
Excl Criteria (a): Is study external to global software engineering?		only interested in GSE/GSD as the focus
Excl Criteria (b): Is study external to teaching and learning?		needs also to be focussed on education
Excl Criteria (c): Is study based on personal opinion/viewpoint?		needs a level of rigour so we can trust the results (even from an expect) - anything without a good theoretical foundation or based on evidence/empirical study we reject
Excl Criteria (d): Is this a repeated study?		include key study only (most comprehensive), repeating results when author publishes in several venues will bias our results
Incl Criteria (a): RQ Answered?		State which RQ is addressed in this study (can be both)
Inclusion Criteria (b): Acceptable source?		Exclude: Books, Book chapters; PhD theses, Tech reports, non-peer reviewed sources, posters, proceeding front matters/sets or short papers (<=two pages). Incl conference/workshop proceedings and journal papers.
DECISION		
Decision Status: {Accept/Reject/Waiting for Full paper/Don't Know}		"Don't know" decisions will go to arbitration. Please use exact wording, as papers will be classified according to how you code this field.
Decision Based on: {Abstract/ Intro/ Conclusion/ Method/Whole Paper/ Peer Review/ Arbitration}		at what point did you make your decision
CONTEXT OF STUDY		
Course / subject taught : (one per row – add more if needed)		Applies to theoretical & empirical studies; e.g. a course on cultural awareness in GSD can be an e-learning training tool or an in-class course. ADD more rows if you need to – one per subject taught
Population: {HE student/ practitioner/ other}		ADD more rows if you need to - one per type
Type of study: Valentine's taxonomy		Indicate type: Marco Polo, Tools, Experimental, Nifty, Philosophy, John Henry (Only one type)
For empirical studies add:		
Geographical area : (one country per row, add more if needed)		list countries involved in study (i.e. sites used) One row per country (ADD more rows if needed)
Number of sites used :		give number - if not known state' not given' : use numbers not text. e.g. 2, (not two).

Appendix F continued.

Data extraction form Phase 2.

PHASE 2: Qualitative Data Extraction*. Please complete following ONLY if paper is accepted- i.e. has passed all criteria in Phase 1 above*				
QUALITATIVE DATA EXTRACTION	Challenge/Solution	**Major Category (based on themes spreadsheet or other inductively derived categories that emerge)**	**Minor Category (based on themes spreadsheet or other inductively derived categories that emerge)**	**Comment** PLEASE NOTE: Your lists of how study answers our RQs will go into our 'Data Synthesis' stage - where we aggregate all our findings across ALL our accepted papers. So please do not interpret what the authors have found, and try to keep your description very short (one or two sentences per challenge/practice at most)
Challenge in Teaching GSD (RQ1)				RQ1: What are the key challenges in delivering GSE courses to SE Students? List as many as you find (create additional rows if needed - one row per challenge)
Challenge in Teaching GSD (RQ1)				Add more rows if needed; use exact text from column A in new column A.
Recommendation for Teaching GSD (RQ2)				RQ2: What are the key recommendations for delivering GSE courses to SE Students? List as many as you find (create additional rows if needed - one row per recommendation)
Recommendation for Teaching GSD (RQ2)				Add more rows if needed; use exact text from column A in new column A.
Methodology (if experiment)(Action Research, Field Study, Descriptive Case Study, Experience Report)				Describe the method used in the study (if appropriate)
Method/Analytical technique (if experiment) {Questionnaire/survey; Face to face interviews; Observation; Focus Groups, prototyping}				Describe the method used in the study (if appropriate)
Quality of execution (if experimental in line)				
Goal of paper (optional)				What was the overall goal of the study?
Emerging Theme (optional)				List any themes in terms of GSD challenges or recommendations
ADDITIONAL DATA/FOLLOW UP				
Other observations or useful quotes found in paper				Record useful text here / exact quotes we can use in our report
Other observations or useful quotes found in paper				
References found in paper/snowballing (to follow up)				Can pre-date year 2000
References found in paper/snowballing (to follow up)				