Team Resilience in Distributed Student Projects

Igor Čavrak
University of Zagreb
Faculty of Electrical Engineering and Computing
Croatia
igor.cavrak@fer.hr

Ivana Bosnić
University of Zagreb
Faculty of Electrical Engineering and Computing
Croatia
ivana.bosnic@fer.hr

ABSTRACT

Global software engineering education is steadily advancing to fully prepare students for future challenges at work, by providing opportunities for real-life experiences, especially in distributed project-based courses. These international student teams are, as in real companies, susceptible to various risks stemming from different internal and external factors, being the sources of stress and impacting team resilience.

In this paper, we focus on studying the resilience of teams affected by two adversities specific to project team composition and project dynamics. The first, internal one, is the teams' ability to compensate for the missing efforts of its non-contributing members in a distributed project environment. The second, external, is the ability to cope with changes of customer requirements, which can be real, or just perceived by the team members.

Based on the empirical data, acquired from a number of Distributed Software Development course instances, we identify different sub-factors of these two risks and analyze the correlations of these elements to the final project evaluations, more specifically on the evaluation of both the final product and project process.

CCS CONCEPTS

• **Social and professional topics → Project management techniques**; **Software engineering education**; • **Human-centered computing** → *Computer supported cooperative work*;

KEYWORDS

Software engineering education, global software engineering, distributed software development, team resilience, requirements volatility

ACM Reference Format:

Igor Čavrak and Ivana Bosnić. 2018. Team Resilience in Distributed Student Projects. In *ICGSE '18: ICGSE '18: 13th IEEE/ACM International Conference on Global Software Engineering , May 27–29, 2018, Gothenburg, Sweden.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3196369.3196370

ICGSE '18, May 27–29, 2018, Gothenburg, Sweden

ACM ISBN 978-1-4503-5717-3/18/05…$15.00
https://doi.org/10.1145/3196369.3196370

1 INTRODUCTION

It is a known fact that team dynamics is crucial for having successful software engineering projects. This is even more relevant in projects distributed over two or more locations, which is today a common way of collaboration. To prepare future engineers, many higher education institutions have a kind of project-based course in the final years, presenting students the opportunity to try project work, either locally or in distributed project groups. These groups are susceptible to all sorts of risks and sources of team stress, stemming from either organizational, social or technical challenges. These adversities introduce some kind of change in the project that can be detrimental for the ongoing project process and/or the final product. The successful answer of the team to such stress lies in the quality called team resilience. As defined by [13], resilience is "a fundamental quality of individuals, groups, organizations, and systems as a whole to respond productively to significant change that disrupts the expected pattern of events without engaging in an extended period of regressive behavior".

In this paper we study the impact of two team stress sources in a distributed, project-based course called Distributed Software Development, analyzing the resilience of stressed teams and correlations to the final project success. We address one internal and one external project team stress source and the ability of the team to respond and absorb those stresses while preserving its performance.

Non-contributing team members present the internal source of stress, effectively reducing the team capacity to perform in certain project work aspects. We try to determine roots of students' lack of contribution, the effect on the team performance and the ability of the team to compensate given the number of non-contributing students within the team.

An external source of stress studied in this paper broadly relates to the type of project and type of project customer, given the *level of initial requirement definition quality and changes of those requirements* during the project duration. Well-organized distributed teams facilitate information flow and awareness regardless of the distance, but poorly organized ones struggle. Team resilience, indicated by preservation or graceful degradation of performance in the context of frequently changing requirements should present a strong indication of sound distributed team organization.

The rest of the paper is organized as follows. Section 2 shortly refers to related work on team resilience. Section 3 presents a brief overview of the DSD course. Section 4 describes research method with data sources used for the analysis, and Section 5 presents the analysis results related to two aspects of distributed team resilience. Section 6 discusses the lessons learned and concludes the paper.

2 RELATED WORK

Various efforts have been made to define and study resilience, mostly in psychology field. In general, resilience is the capacity to rebound from an adversity strengthened and more resourceful [19]. Starting from individual or personal resilience, it can be described as an adaptive system which enables a person to "bounce back" from a setback or failure [8]. In a similar manner, team resilience can be conceptualized as a team's belief that it can absorb and cope with strain, as well as a team's capacity to cope, recover, and adjust positively to difficulties [5]. It is interesting to note that, in comparison to personal resilience, a collection of resilient individuals within a company might not add up to a resilient organization. These resilient persons might dominate the team and override the shared vision of other team members [13].

Research on resilience brings good news: competencies for team resilience can be improved. In [8], seven "streams" of behavior used to improve team resilience are shortly introduced as: *community*, *competence*, *connections*, *commitment*, *communication*, *coordination* and *consideration*. In [15], authors propose seven basic resilience principles for organizations: (a) *perceive experiences constructively*, (b) *perform positive adaptive behaviors*, (c) *ensure adequate external resources*, (d) *expand decision-making boundaries*, (e) *practice bricolage*, (f) *develop tolerance for uncertainty*, and (g) *build virtual role systems*. An experience from collaboration with NASA is described in [1], where authors list around 20 team resilience challenges, but also 40 practical behaviors of resilient teams, grouped into Minimize (Before), Manage (During) and Mend (After) groups related to the period of adversity.

IT-related field, with quality teamwork indispensable for successful projects, also needs to research the topic of resilience. In [2], authors surveyed 115 members of IT-related projects, developed in academia, collecting 48 actions to improve project team resilience. As part of the survey, participants ranked the actions proposed, having "promote collaboration" and "promote solidarity" perceived as the most useful. Authors of [14] focused their research of 29 Internet software development projects on the following question: *Do development practices that support a more flexible process have a stronger association with performance in projects that face more uncertain contexts?* The question of flexibility is related to team resilience, as resilient teams should be more flexible to adverse changes. Authors study related hypotheses depending on platform uncertainty and market uncertainty and conclude that merely reacting to change is just an appearance of flexibility, while real flexible processes should be invested into from the early phases of the project, long before the need for changes occurred.

In psychology literature, it is often stated that teams become more hierarchical, authority-oriented and less democratic in situations of high stress, an important test situation for team resilience. Building on this, [10] finds that leaders of small teams also become more receptive of others' opinions in high-stress situations. In bigger teams, it is possible that the leader chooses only a subset of team members whose opinion becomes more relevant. In a meta-analysis of research papers on conflicts, authors of [9] conclude that conflicts related to tasks have a strong negative correlation to team performance and team member satisfaction, especially in highly complex (process-related) tasks compared to less complex (production) tasks. For authors of [11], building team resilience is improved by direct engaging not only into task conflicts, but relationship conflicts as well, as they will eventually appear and produce a substantial team damage. These "hot topics" should be discussed only in "cool manner" – managing self, managing conversations and managing relationships.

There are not many resources on team resilience in distributed project work. The field study of 43 teams in a large multinational company showed that geographically distributed teams have more both task and interpersonal conflicts than their similar collocated teams. In mitigating these conflicts to maintain team resilience, spontaneous communication plays an important role, with direct effects in conflict management [12].

In educational context, there is a number of international courses dealing with the same topic - global/distributed software engineering courses. An extensive literature review of 82 papers about global software engineering courses was published in 2015, providing a list of challenges and recommendations in such education. [7]. Examples of similar courses include:

- The DOSE project (Distributed and Outsourced Software Engineering), organizing the course on three continents, among 12 universities. Several course instances included a role–playing game - a contest - to prepare students to work in a distributed manner and present them the challenges of such work.[16]
- The Runestone project, at the time presenting the course among Sweden, Finland and China, which involved working on projects related to LEGO NXT robots, proposed by teaching staff, in a period of 10-13 weeks. [18]
- A GSE course carried out between Finland and Canada, focusing on agile development methods, including GSE best practices. In this course the members move across teams during the course, so they gain experience working in both local and remote teams [17].

Most research in this field was focused on communiation and collaboration in this educational context; however, team dynamics - especally team resilience - was not in focus of these papers.

3 DSD COURSE OVERVIEW

The Distributed Software Development course (DSD) is a joint project–based course of three universities: University of Zagreb, Croatia (FER), Mälardalen Högskola, Sweden (MDH) and Politecnico di Milano, Italy (POLIMI). Started in academic year 2003/04, the course has now been running – and improving – for 15 consecutive years.

The course is run in a distributed manner, with international student teams working together on a project throughout all its phases, from project plan and requirements definition, to implementing, testing, documenting and deploying the full product, while presenting their project status several times during the semester. In 15 years, 510 students from 32 countries and 6 continents participated in 77 projects.

Besides introductory lectures and guest lectures given by professionals from industry involved in global software engineering, team project work is the central theme in the course, giving students practical experience. Projects are sized for teams of 6-8 members (3-4 per site) working for one semester. Throughout the DSD history, there have been several types of projects and project customers, presented in [3], but in recent years our main focus is having projects with external industry partners, as well as ICSE SCORE[1] competition – students' contest on software engineering, organized by International Conference on Software Engineering (ICSE). Both of these project types pose additional challenges to project teams and their resilience, but have additional educational benefits for students and their learning experience.

Our projects currently use agile development methodology, specifically SCRUM framework. In it, students choose the Product Owner and Scrum Master among themselves; the first one being more responsible for communication with customers, while the second one focusing on team organization. Although the other students do not have a specific SCRUM-role, we try to steer the teams to be as democratic as possible, and to share the responsibilities among themselves equally. Teams follow the usual SCRUM rules, divide the work in Sprints, produce artifacts like Product and Sprint Backlogs, organize different types of meetings, etc. As they work distributed and this course is not their "real" workplace, students sometimes customize some Scrum elements, for instance, organize daily stand–up meetings by writing their time sheets on–line, or have such meetings every few days.

Teaching staff, organized as team supervisors, invests big efforts to support and accompany the team, not mainly in solving technical and programming issues, but in providing feedback and advising students on team issues, problems stemming from distributed nature of work, differences in students' levels of knowledge and motivation, communication and collaboration challenges, work organization, etc.

Towards the end of the course, after submitting the final versions of their product and documentation, projects are evaluated on more than 50 weighted elements: the quality of documentation, final product, process and presentations. This evaluation outputs a number of points, which are sent to student teams, who are asked to jointly propose the points division among team members. After this, the final course grades are given by the teachers, having in mind not only points distribution, but teachers' perception of each student's work and effort, as well as student's reflection on events in the team and the method of distributed development in general.

More details on several aspects of DSD course organization and rationale, are provided in other papers; the list of publications is available at our course website[2].

This method of education – using project-based courses, especially distributed and "tightly-coupled" among course partners, bring along a number of additional, wider risks. Two levels of risks, course-level (dealing mostly with organization) and project-level (focusing on students), occurring in our DSD course, are described in [4]. Some of these risks are related to team resilience, studied in this paper.

[1]ICSE SCORE competition: http://score-contest.org/
[2]DSD course website: http://www.fer.unizg.hr/rasip/dsd

4 RESEARCH METHOD

The research on the various aspects of the distributed software engineering education, being conducted since the inception of the DSD course, is a combination of a longitudinal case study and action research. Data, originating from various sources, addressing multiple course aspects and actors involved has been systematically collected, stored and analyzed. Data and corresponding sources used for the research described in this paper have been summarized in Table 1.

To assess the impact of internal or external stress source to project performance, we use three compound measures: overall project performance, process performance, and product performance. Project scores on aforementioned performance indicators are determined during the project evaluation process at the end of the course and recorded in the Project evaluation form. Performance scores are expressed on a percentage scale 0-100%. Overall performance indicator collectively measures all project aspects (product, process, documentation, and presentation quality aspects) and provides an overarching measure. More specific process and product indicators are also used, to reveal potential differences in how project teams react to different stress sources and their levels.

A number of perception indicators are used in the analysis of team resilience. Students evaluate their perception of a subject matter either quantitatively or qualitatively, for example, quality of information distribution within the distributed team or volatility of requirements. Most of the qualitative indicators use scales 1–5 or 1–10. Such perception indicators are predominantly extracted from students' Final questionnaires, but other, project-wide sources are also used, for example, Final project report documents containing qualitative and quantitative data on various project aspects. In this study, we restrict the analyzed indicators only to quantitative, reducing the risk of miscoding or misinterpretation of qualitative ones.

Individual student grades are used in assessing student contribution to teamwork. Although each institution may (or does) in the end locally use different grading scales, grading scale 1–5 is used in our joint evaluation process of each student, where 5 represents the highest grade.

5 DISTRIBUTED TEAM RESILIENCE

5.1 General Project Data

Data analyzed in this paper originates from various data sources (described in the previous section) resulting from 23 distributed student projects conducted in academic years 2012. – 2016. The total number of students attending the course during the analyzed period was 156, coming from 45 countries and five continents (Table 2). Institutional distribution of those students was as follows: FER 60, MDH 56, and POLIMI 40 students.

Optimal team size for each distributed project is 6-8 students, equally divided between two remote locations. The imbalance in enrollment on involved institutions may cause a diversion from this guideline, resulting in undersized or oversized teams. In addition to suboptimal team sizes, imbalances of team members on two involved locations may occur, posing additional risks for effective distributed work. In the analyzed period there were two occurrences of undersized and oversized project teams (Table 3).

Table 1: Data Sources

Source	*Description*
Student self-assessment polls	Initial polls collecting students' self assessment of their knowledge of tools and technologies
Student project polls	Initial polls collecting students' preferences on proposed projects
Final project reports	Contain summarized qualitative and quantitative data on various project aspects. Jointly written by team members.
Summary week reports	Contain per-week quantitative and qualitative information on project advancement.
Final questionnaires	Individually filled out by students at the end of the course, reflecting students experience and knowledge gained during the distributed project work. Contain qualitative and quantitative observations / perceptions on many aspects related to working on a distributed project.
Project evaluation forms	Projects are evaluated using more than 50 criteria grouped into four criteria groups (documentation, presentation, process and product)
Student grades	Student final grades on a scale 1-5

Table 2: Course statistics 2012.–2016.

Year	'12	'13	'14	'15	'16
Students	49	22	26	21	38
Nations	14	11	14	12	13
Projects	7	3	4	3	6

Table 3: Project team sizes 2012.–2016.

	undersized	optimal			oversized	
Team size	5	6	7	8	9	10
Projects	2	9	8	2	1	1

Optimal team size and balance guidelines for distributed teams, in the context of the DSD course, have been derived from our experience and address the following educational aspects of distributed project work:

i Too small teams fail in exposing students to complexities of distributed coordination and collaboration, not requiring teams to form distributed organizational structures and collaboration patterns, resulting in tightly-coupled virtual teams that can succeed in the product, but usually fail in the process aspect of the course.

ii Too large teams also tend to fail in the process aspect of the course due to inability to effectively coordinate and utilize a larger number of team members.

iii Team size of 6-8 is acceptable for participation in international software engineering contests, such as SCORE competition.

iv Team member imbalance between locations creates an imbalance of power within project teams, often resulting in the dominance of one location over the other in decision making, effective exclusion of one location from contributing to the project work, subsequently failing in process aspect of the course.

Cultural and educational background diversity within project teams can exert a significant influence on the team's performance. Table 4 summarizes cultural diversity of the analyzed projects based on the number of unique UN subregions project team members originate from, while Table 5 summarizes educational background diversity by estimating the number of unique educational institutions project team members originate from. The data presented in both tables suggest that the analyzed projects were characterized with a low-to-moderate cultural diversity of their members (78% of projects had their team members originating from 2 or 3 different regions[3]) and a low-to-moderate educational diversity (74% of projects' team members originating from 2 or 3 educational institutions).

Table 4: Cultural Diversity

# UN subregions	1	2	3	4
Projects	2	9	9	3

Table 5: Educational Diversity

Educational backgrounds	2	3	4	5	6
Projects	8	9	4	1	1

5.2 Resilience to Non-Contributing Team Members

Non-contributing team members, classified as such by receiving a final project grade less than 4 (on a scale 1–5) effectively reduce the project team size after the project kick-off, with a more adverse effect on teamwork than having a smaller team from the start. Those students usually restrict their activities to only dealing with project documentation and presentations, or are isolated by other team members as useless in the technical part of the project due to their insufficient technical/educational background.

We distinguish two types of non-contributing team members: those that finish the course and receive lower grades and those that leave the course without receiving a grade and leaving any kind of useful feedback (not submitting the final questionnaire etc.). The option of leaving the course is institution-dependent and student type dependent (exchange students can drop out much more easily than regular students), influencing the students' motivation to actively participate in project work. Table 6 summarizes student types

[3] https://en.wikipedia.org/wiki/United_Nations_geoscheme

Table 6: Non-contributing students

	Regular	*Exchange*
# students	132	24
# non-contributing	17	10
% non-contributing	12.9%	41.7%
# dropout	2	4
% dropout	1.5%	16.7%

Table 7: Projects with non–contributing students

Non–contributing students	0	1	2	3	4
Projects #	10	6	2	3	2
Projects %	43.5%	26%	8.7%	13%	8.7%

attending the DSD course during the analyzed period; much higher rate of non-contribution and leaving the course can be observed in the subpopulation of exchange students.

Table 7 summarizes the data on non-contributing students per project, during the analyzed period. Almost 60% of projects experienced at least one non-contributing team member.

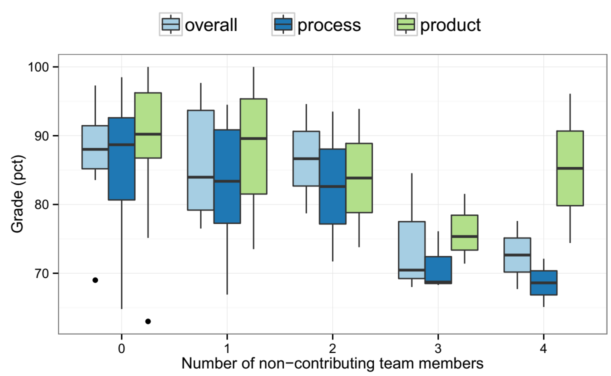

Figure 1: Project performance vs number of non–contributing team members

The effect of the number of non-contributing students on project performance is presented in Figure 1. Project teams having up to two non-contributing members experience graceful degradation of performance indicators (*overall* project grade, *process*, and *product* criteria grades). *Product* criteria score is consistently higher than *process* criteria score, but is degrading much faster than *process* criteria; the number of team members available for product development is smaller, but the distributed team manages to preserve its organization. Strong degradation of performance indicators is visible for the project with more than two non-contributing members, where *process* criteria experiences sharper decrease. In such scenarios with strong lack of contribution, distributed teams tend to degenerate into one local, collocated main team with optionally attached remote individuals, trying to salvage the product resorting

Table 8: Number of projects with contributing students

Contributing students	2	3	4	5	6	7	8	9
Projects	1	2	3	4	7	2	3	1

to simple ad-hoc development process, which might even end up obtaining decent *product* grades, but misses the point of our course in which the collaboration aspect is emphasized.

Besides observing the effect of the number of non-contributing team members, it can be insightful to evaluate the performance of distributed teams with respect to the number of contributing team members. Table 8 summarizes the data on the number of DSD course projects during the analyzed period with a number of contributing team members.

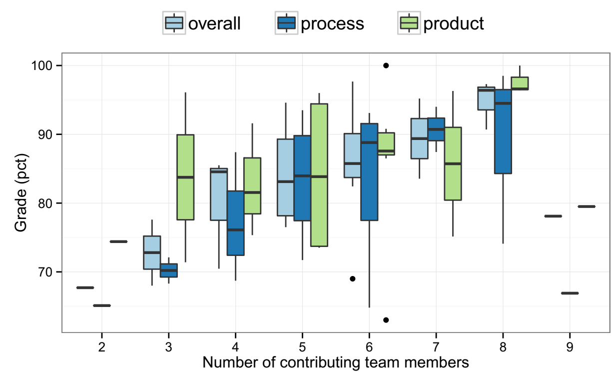

Figure 2: Project performance vs number of contributing team members

Figure 2 reveals the linear relationship of the number of contributing team members with the overall and process performance indicators; availability of work hours and team size allow for conducting comprehensive projects and require adhering to processes used in a distributed setting in order to coordinate distributed project work. Smaller teams cannot address all project aspects (product, process, documentation, presentations) at the same time and with the same quality, and tend to focus on product aspect only.

It is interesting to note the absence of correlation between a number of contributing team members and the product performance indicator. For smaller teams (up to four) the aforementioned product-focus applies, but for larger teams process criteria equalizes with or outgrows product criteria score; in larger teams distributed collaboration is unavoidable, thus emphasizing the need to address process aspect, and, at the same time, lowering team resources for investing in product aspect.

If observing from the coarse-grained classification level of team sizes (under-, optimal and over-sized project teams), interesting repercussions on resilience to non-contributing team members can be noticed. Table 9 summarizes the number of projects within each category when all (nominal) or only contributing (effective) team members are taken into consideration.

Table 9: Project team sizes (nominal and effective)

Classification	*undersized*	*optimal*	*oversized*
# nominal	2	19	2
# effective	10	12	1

Table 10: Projects preserving team size classification

Classification	*undersized (s)*	*optimal (m)*	*oversized (x)*
# projects	2	11	1

Table 11: Projects changing team size classification

Classification	*oversized to optimal*	*optimal to undersized*
# projects	1	8

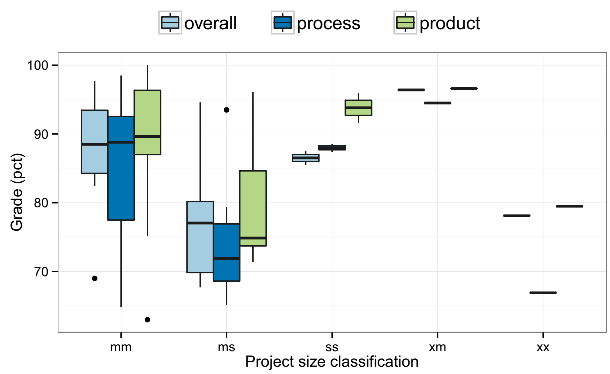

Figure 3: Project performance vs number of contributing team members

Table 10 and Table 11 summarize the number of projects that preserve and change category respectively, in case the effective team member size is considered instead of the nominal one. Figure 3 depicts the process performance indicator scores for such projects (*m* – optimally sized projects, *s* – undersized projects, *x* – oversized projects, pairs of letters on the x–axis indicate nominal and effective size classification; for example, *ms* denotes nominally optimal (*m*) but effectively undersized (*s*) projects). Observing only the categories with a sufficient number of represented projects (*mm*, *ms* and *ss*), the figure suggests that projects that do not change category (*mm* and *ss*) perform much better than projects that do change category due to the reduced effective number of team members. It is also interesting to note that original undersized teams (*ss*) significantly outperform new undersized teams (*ms*) indicating that there exists a threshold that, when crossed, distributed teams exhibit low resilience to such an internal stress.

5.3 Resilience to Changing Requirements

To identify distributed project properties that impact the resilience of the team to external and internal threats, we extensively studied possible dependencies between different data collected over the years of running the DSD course. One of the correlations that emerged was between the *overall project success score* and the observed *differences in sub-teams' perceptions of requirement changes*. (r=-0.47, p=0.02).

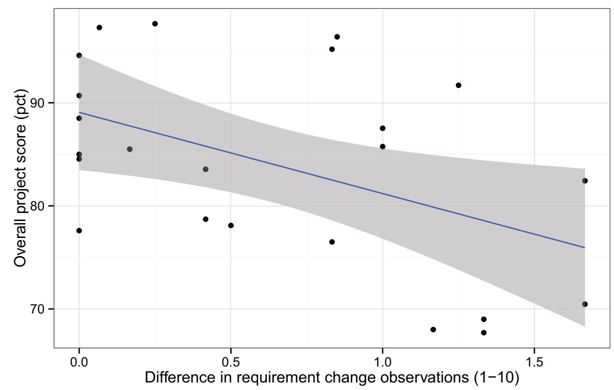

Figure 4: Relation between overall project score and differences in requirements change observations

This moderate correlation implies that the more difference in perception of requirements changes between sub-teams, the less overall project score (Figure 4). However, similar relations could not be confirmed for *product* and *process* criteria, nor similar correlations could be identified using more explicit indicators collected from students (such as perceptions of local and remote collaboration quality, information flow, decision process quality etc.)

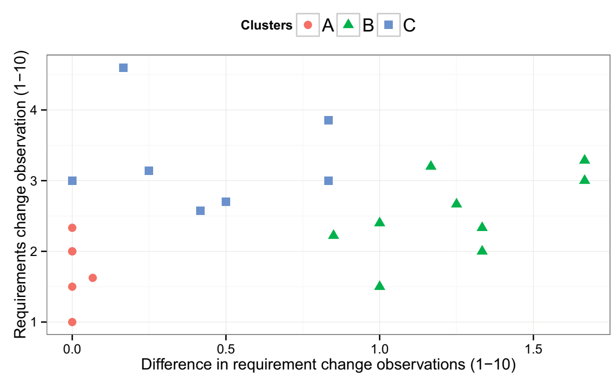

Figure 5: Project clustering

To further investigate the resilience of distributed project teams on changes of requirements during the project work, we grouped the projects into three clusters (Figure 5).

Table 12: Cluster sizes

Cluster	*A*	*B*	*C*
# Projects	6	9	8

Table 13: Mean evaluation criteria scores per project cluster

Cluster	*A*	*B*	*C*
Overall (%)	88.95	79.89	84.97
Product (%)	93.40	82.75	83.92
Process (%)	84.24	78.97	81.86

Clusters were manually formed according to the values of two compound variables:

Requirements change observation — mean value of requirement change observations of all project team members; reflects the magnitude of requirements change as perceived by the whole project team, but not necessarily the same for remote sub-teams.

Difference in requirement change observations — the difference between mean values of requirement change observations per distributed sub–team; reflects the difference in perception of remote sub-teams, mostly due to lack of communication and/or coordination between remote sub–teams.

Cluster A encompasses projects with the *non-existent or minimal difference in requirements change observations* between distributed teams, while project–wide mean requirement change being confined to values less than 2.5 (less than half of the variable range). The rationale for cluster A lies in the formation of a "control group" of projects, with project teams not being exposed to significant requirement change stress, and remote teams acting in synchronization.

Cluster B is made of projects that expose *significant differences in requirements change observations*. The true magnitude of the requirements change stress for those teams cannot be reliably assessed (due to the difference between observations of remote teams), so mean value of all team member observations is used as a project–wide measure. We presume that cluster B projects were exposed to medium–to–high requirements change stress and could not effectively compensate it using internal team mechanisms.

Cluster C contains projects with *minimal to moderate differences in requirements change observations*, but with *moderate to high requirements change stress* (variable value between 2.5 and 5). These teams presumably had a clear joint view on the project state (therefore presenting small differences in observations on changes), but they did have to deal with significant requirement changes.

Figure 6 presents the project success evaluation results for clustered projects. Projects in cluster A score better than projects in other clusters on all criteria, as expected due to a lack of requirement change stress. Interesting differences in evaluation scores can be observed for clusters B and C; projects in cluster B, on average, tend to score higher in the *product* than the *process* criteria and have a better score in the *product* criteria than cluster C projects. However, they have worse scores and higher score variability than cluster C projects in the *process* criteria.

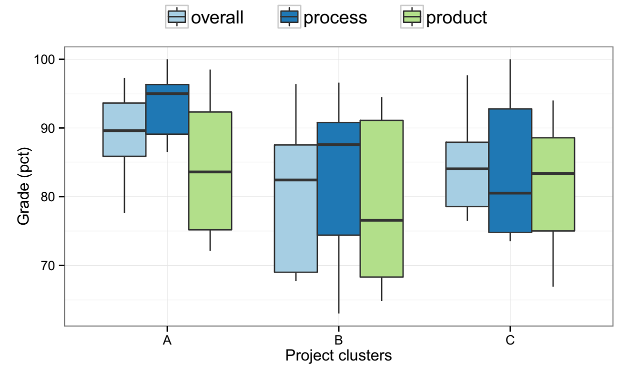

Figure 6: Compound evaluation criteria values for project clusters

Our interpretation of the results is the following: as the cluster C projects have confirmed moderate–to–high requirements change stress, their product grade suffers due to unclear and volatile requirements, but they tend to keep the overall criteria score higher than cluster B projects due to better intra–team cohesion and the consequent resilience to external stress factors. They also exhibit high *process* criteria grade, almost the same as the stress–free cluster A projects, also due to their internal cohesion and preservation of collaboration among remote sub–teams. On the other hand, cluster B projects indicate lowered resilience to requirements change stress by both the lowest *process* criteria grade and by rather high difference between *project* and *process* criteria grades: such teams tend to fragment themselves on two loosely coupled remote sub–teams, not being completely aware of the overall project status (resulting in difference in observed requirement changes). For the sake of *product* success and lowered overhead related to distributed collaboration, one of the remote sub–teams, or several more skilled team members regardless of their location, tend to squeeze out others and take over the project — directly contradicting our educational goals.

5.4 Threats to Validity

We assess the threats to validity present in this study guided by the model proposed by Wohlin et al. [20] for empirical studies in the field of software engineering. Threats to validity are divided into three groups: *construct* – validity of data sources used in this study, *internal* – validity of analysis process and results, and *external* – generalization of the results outside the scope of our DSD course.

Neither of the seven data sources were designed with conducting of a specific empirical research study in mind, but to assist in different phases of the student projects; project inception phase (student self-—assessment and project polls) and evaluation phase (remaining sources in Table 1). The two major data sources used in this study do present potential sources of risk to *construct validity*: Final questionnaires and Project evaluation forms. Students are aware that the quality of their Final questionnaires has influence on their final grade, as well as it could affect the grades of their project teammates, therefore we expect them to present certain project aspects in a better light than they actually were. However, questions addressing project requirements were posed in a rather neutral way, without implicating anyone's responsibility, reducing the potential bias towards more positive valuation of this project aspect.

Project evaluation forms provide a detailed evaluation framework listing evaluation criteria and corresponding criteria weights, allowing evaluators only a reduced variation of criteria (limited to 5%) to adjust criteria to individual project properties. In addition, to reduce location bias, each project is evaluated by two teachers; each from a location involved in the specific project; with final evaluation being negotiated between them. Therefore, an additional low *internal validity* threat remains in the small variation of evaluation criteria, and as a result of the negotiation process, teachers' personalities and criteria involved in the project evaluation.

Another*internal validity* threat lies in a moderate number of projects analyzed, where certain project groups resulting from different classifications (number of team members, non–contributing team members, projects changing size classification, etc.) do not possess a valid population size for drawing any statistically significant conclusions. In those borderline cases, we refrained from drawing any conclusions. At the moment, we have no valid way to mitigate this threat as the lack of borderline data can only be remedied by new data becoming available in the future iterations of the course.

The *external validity* of the findings presented in this study can be assessed in two separate contexts: (i) their relevance within the context of other distributed software engineering courses and (ii) their applicability within the industrial context. Concerning the other courses addressing the distributed aspect of the SE, its relevance primarily depends on the project framework used in the course: results could be highly relevant only to courses utilizing the full development cycle. However, other factors could affect the results of the replicated studies in the seemingly similar environments: differing educational background, enrollment policies, cultural background, and work ethics.

The educational context of (distributed) software engineering projects imposes many limitations, primarily in the dimensions of project complexity, available time, ethical issues and fairness, as well as limited funding [6]. Therefore, implicating that presented results could be generalized to the industrial setting would be overly ambitious.

6 CONCLUSIONS

In this paper, we focused on the study of distributed student team resilience by exploring their reactions to two stress sources: non–contributing team members and volatile requirements. Team reactions were evaluated by observing changes to project performance indicators – overall project, process, and product criteria, resulting from final project evaluation.

Non–contributing team members, stress source internal to the project, reduce the project team size and impact its ability to produce a quality product and follow the desired development process. Distributed teams exhibit a fair amount of resilience, given that the team is not reduced to the size where following the distributed development process becomes more of a burden than a tool needed to coordinate the distributed effort. In such a case, reduced teams focus on the product and neglect the process part of the development, hampering the educational goals set before them.

The volatility of requirements, stress source external to the distributed team, tends to expose weaknesses in team coordination and information flow. Cohesive distributed teams exposed to moderate–to–high requirement volatility have lowered product performance indicator, but preserve high resilience of process indicator. Non–cohesive teams, identified by significant differences in perception of requirement volatility between remote sub-teams, tend to focus on product aspect of the distributed project, experiencing a sharp drop of process indicator values.

The ability of the student project team to adapt to stress without significantly sacrificing performance is a crucial property in the educational setting. It provides the opportunity to the teaching staff to expose student teams to external stress sources in a controllable fashion and to estimate and control risk factors endangering the desired educational outcomes.

A further study of the distributed student team resilience to external and internal stress sources in educational context will explore two paths: the first one focused on discovering additional stress sources and reliable indicators revealing weaknesses in distributed team performance. The second one will aim at identifying effects of stress to changes and adaptation mechanisms within distributed teams, such as the impact of project's collaboration patterns on team resilience to particular stress.

ACKNOWLEDGMENTS

The authors would like to thank all former and current colleagues from Mälardalen University, Sweden and Politecnico di Milano, Italy, for the joint effort in working on DSD course.

REFERENCES

[1] George M. Alliger, Christopher P. Cerasoli, Scott I. Tannenbaum, and William B. Vessey. 2015. Team resilience: How teams flourish under pressure. *Organizational Dynamics* 44, 3 (2015), 176 – 184. https://doi.org/10.1016/j.orgdyn.2015.05.003

[2] António Amaral, Gabriela Fernandes, and João Varajão. 2015. Identifying Useful Actions to Improve Team Resilience in Information Systems Projects. *Procedia Computer Science* 64 (2015), 1182 – 1189. https://doi.org/10.1016/j.procs.2015.08.549 Conference on ENTERprise Information Systems/International Conference on Project MANagement/Conference on Health and Social Care Information Systems and Technologies, CENTERIS/ProjMAN / HCist 2015 October 7-9, 2015.

[3] I. Bosnic, I. Cavrak, M. Zagar, R. Land, and I. Crnkovic. 2010. Customers' Role in Teaching Distributed Software Development. In *2010 23rd IEEE Conference on Software Engineering Education and Training*. 73–80. https://doi.org/10.1109/CSEET.2010.27

[4] I. Bosnic, F. Ciccozzit, I. Cavrak, R. Mirandola, and M. Orlic. 2013. Multi-dimensional assessment of risks in a distributed software development course. In *2013 3rd International Workshop on Collaborative Teaching of Globally Distributed Software Development (CTGDSD)*, Vol. 00. 6–10. https://doi.org/10.1109/CTGDSD.2013.6635238

[5] Abraham Carmeli, Yair Friedman, and Asher Tishler. 2013. Cultivating a resilient top management team: The importance of relational connections and strategic decision comprehensiveness. *Safety Science* 51, 1 (2013), 148 – 159. https://doi.org/10.1016/j.ssci.2012.06.002

[6] Igor Cavrak and Rikard Land. 2003. Taking Global Software Development from Industry to University and Back Again. In *Proceedings of ICSE International Workshop on Global Software Development (GSD)*.

[7] Tony Clear, Sarah Beecham, John Barr, Mats Daniels, Roger McDermott, Michael Oudshoorn, Airina Savickaite, and John Noll. 2015. Challenges and recommendations for the design and conduct of global software engineering courses: A systematic review. In *Proceedings of the 2015 ITiCSE on Working Group Reports*. ACM, 1–39.

[8] Diane L. Coutu. 2002. How resilience works. *Harvard Business Review* (2002).

[9] Carsten K W De Dreu and Laurie R. Weingart. 2003. Task versus relationship conflict, team performance, and team member satisfaction: A meta-analysis. *Journal of Applied Psychology* 88, 4 (2003), 741 – 749. https://doi.org/10.1037/0021-9010.88.4.741

[10] James E. Driskell and Eduardo Salas. 1991. Group Decision Making Under Stress. *Journal of Applied Psychology* 76, 3 (1991), 473 – 478. https://doi.org/10.1037/0021-9010.76.3.473

[11] Amy C. Edmondson and Diana McLain Smith. 2006. Too Hot to Handle? How to Manage Relationship Conflict. *California Management Review* 49, 1 (2006), 6–31. https://doi.org/10.2307/41166369 arXiv:https://doi.org/10.2307/41166369

[12] Pamela J. Hinds and Mark Mortensen. 2005. Understanding Conflict in Geographically Distributed Teams: The Moderating Effects of Shared Identity, Shared Context, and Spontaneous Communication. *Organization Science* 16, 3 (2005), 290–307. https://doi.org/10.1287/orsc.1050.0122 arXiv:https://doi.org/10.1287/orsc.1050.0122

[13] John F. Home and John E. Orr. 1998. Assessing behaviors that create resilient organizations. *Employment Relations Today* 24, 4 (1998), 29–39. https://doi.org/10.1002/ert.3910240405 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/ert.3910240405

[14] Alan MacCormack and Roberto Verganti. 2003. Managing the Sources of Uncertainty: Matching Process and Context in Software Development. *Journal of Product Innovation Management* 20, 3 (2003), 217–232. https://doi.org/10.1111/1540-5885.2003004 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/1540-5885.2003004

[15] Larry Mallak. 1998. Putting organizational resilience to work. *Industrial Management (Norcross, Georgia)* 40, 6 NOV./DEC. (1998), 8–13.

[16] M. Nordio, H. C. Estler, B. Meyer, N. Aguirre, R. Prikladnicki, E. Di Nitto, and A. Savidis. 2014. An experiment on teaching coordination in a globally distributed software engineering class. In *2014 IEEE 27th Conference on Software Engineering Education and Training (CSEE T)*. 109–118. https://doi.org/10.1109/CSEET.2014.6816788

[17] Maria Paasivaara, Kelly Blincoe, Casper Lassenius, Daniela Damian, Jyoti Sheoran, Francis Harrison, Prashant Chhabra, Aminah Yussuf, and Veikko Isotalo. 2015. Learning global agile software engineering using same-site and cross-site teams. In *Proceedings of the 37th International Conference on Software Engineering-Volume 2*. IEEE Press, 285–294.

[18] A. Pears and M. Daniels. 2010. Developing global teamwork skills: The Runestone project. In *IEEE EDUCON 2010 Conference*. 1051–1056. https://doi.org/10.1109/EDUCON.2010.5492460

[19] Kathleen Sutcliffe and Timothy Vogus. 2003. Sutcliffe, K. M. and T. J. Vogus (2003). Organizing for Resilience. Positive Organizational Scholarship: Foundations of a New Discipline. K. S. Cameron, J. E. Dutton and R. E. Quinn. San Francisco, CA, Berrett-Koehler: 94-110. (03 2003), 94-110 pages.

[20] C. Wohlin, P. Runeson, M. Höst, M.C. Ohlsson, B. Regnell, and A. Wesslén. 2012. *Experimentation in Software Engineering* (1 ed.). Springer-Verlag Berlin Heidelberg. https://doi.org/10.1007/978-3-642-29044-2