Resilience of Distributed Student Teams to Stress Factors: a Longitudinal Case-study

Igor Čavrak[a], Ivana Bosnić[a], Federico Ciccozzi[b], Raffaela Mirandola[c]

[a] *University of Zagreb, Faculty of Electrical Engineering and Computing, Zagreb, Croatia*
[b] *Mälardalen University, School of Innovation, Design and Engineering, Västerås, Sweden*
[c] *Politecnico di Milano, Milano, Italy*

Abstract

Context: Teaching global software engineering is continuously evolving and improving to prepare future software engineers adequately. Geographically distributed work in project-oriented software development courses is both demanding and rewarding for student teams, who are susceptible to various risks stemming from different internal and external factors, being the sources of stress and impacting team performance.
Objective: In this paper, we analyze the resilience of teams of students working in a geographically fully distributed setting. Resilience is analyzed in relation to two representative stress factors: non-contributing team members and changes to customer project requirements. We also reason on team collaboration patterns and analyze potential dependencies among these collaboration patterns, team resilience and stress factors.
Method: We conduct a longitudinal case-study over five years on our Distributed Software Development (DSD) course. Based on empirical data, we study team resilience to two stress factors by observing their impact on process and product quality aspects of team performance. The same performance aspects are studied for identified collaboration patterns, and bidirectional influence between patterns and resilience is investigated.
Results: Teams with up to two non-contributing members experience graceful degradation of performance indicators. A large number of non-contributing students almost guarantees the occurrence of educationally undesirable collaboration patterns. Exposed to requirements change stress, less resilient teams tend to focus on delivering the functional product rather than retaining a proper development process.
Conclusions: Practical recommendations to be applied in contexts similar to our case have been provided at the end of the study. They include suggestions to mitigate the sources of stress, for example, by careful planning the team organization and balancing the number of regular and exchange students, or by discussing the issue of changing requirements with the external customers before the start of the project.

1. Introduction

Team dynamics is a crucial success (or failure) factor in any teamwork. Software engineering is no exception, especially since it is very often carried out in a geographically distributed manner [31]. To properly train students to become skillful software engineers, virtually any higher education institution offers some kind of project-based software engineering course in their master programs, giving students the opportunity to work on a project in teams whose members are either geographically co-located or distributed. The teams are exposed to all types of risk and sources of team stress, stemming from organizational, social and technical issues. When a team experiences such adversities, negative repercussions tend to affect both the development process and the resulting product quality. The ability of a team to endure and overcome stressful happenings lies in a specific quality: team resilience. We address resilience as per definition by [24]: *"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"*. Research efforts have been devoted to study team resilience (a general discussion can be found in [36]); in the era of distributed software development, sources of stress have been analyzed to evaluate their impact on the project success. In [3, 4] Avritzer et al. focus on issues related to communication, cultural diversity and process deficiencies to define a project survivability model. Noll and Beecham [38] study the global distance aspects and their impact on distributed project. A set of strategies to be applied to increase the project resilience in a distributed setting is presented in [31].

In this study, we analyze the impact of two sources of stress for teams involved in our distributed project-based Distributed Software Development (DSD)[1] course. The first stress source is internal to the team and is caused by the existence of one or more non-contributing team members. The second one is external to the team and is caused by significant changes in project requirements and propagation of information on those changes within the distributed team.

We have selected these two stress sources as stemming from our 15 years of experience running the DSD course, we found these to be the causes of the most pressing risks to student project teams. The two addressed stress sources were, by no means, the only ones we have encountered on the DSD course; a number of different factors impeded student team performance, ranging from omnipresent factors in software development industry [50], factors stemming from the global context [40], and from various diversities specific to the educational context and particular course setup [7]. The most frequent internal stress sources we have encountered, apart from the non-contributing team members, included diversities in cultural and educational backgrounds, and personal conflicts between team members. External stress sources were predominantly related to

[1]DSD course webpage: `https://www.fer.unizg.hr/rasip/dsd`

project customers (initial requirements specification and their variability, customer availability) and, to a lesser extent, organizational context shared among the three participating universities.

Although we have conducted several short- and long-term studies on distributed student teamwork, we have not yet explicitly addressed the effects of non-contribution and requirements change to team performance in detail and the ability of student teams to cope with them – their resilience. The additional motivation for this study was, to our best knowledge, the absence of similar studies addressing those two topics in depth. Considering the individual and team challenges in GSD courses, a significant number of publications exist, as listed in [14], however, we have not identified those directly addressing the impact of student non-contribution to project team performance and team organization. Requirements management in the GSD education has also been a topic of a number of publications (for example, [25], [16], [44]), and external customer-related issues have been identified such as feature-creep [20] and non-responsiveness [8]. Unfortunately, none of the studies focused on the impact of requirements volatility, predominantly caused by external customers, on student team performance and the resilience of student teams to such occurrences.

We assess resilience to a specific stress factor by observing its impact on team performance indicators derived from the end-of-semester project evaluation. Team performance is assessed on three aspects: *(i)* overall project performance – taking into consideration four major aspects of student project work: process, product, documentation and presentation quality, *(ii)* adherence to mandatory development process and *(iii)* resulting product quality.

Non-contributing team members present an internal source of stress that may more or less dramatically reduce the team's ability to perform properly in certain project work aspects. We investigate the potential roots of students' lack of contribution, the effect on the team performance and the ability of the team to compensate it given a number of non-contributing students in the team. We also study the impact of this stress on internal team dynamics by identifying project-wide collaboration pattern that the team resorts to when faced with non-contributing students, and individual collaboration roles played by each of those students.

The entailed external source of stress, *requirements change*, broadly relates to the type of project and project customer, given the level of initial requirements definition quality and the level of changes during the project duration. Around 25% of initial requirements change in medium-sized projects and more than 35% in larger projects [28]. These frequent changes become a relevant source of errors in the product [27] and a strong coordination issue in distributed teams [29]. This issue increases team communication and calls for more complex collaboration methods [21]. Well-organized distributed teams facilitate information flow and balance out potential issues related to geographical distribution, while poorly organized teams struggle. Team resilience, indicated by graceful degradation of performance indicators in the context of frequently changing requirements, should present a strong indication of a sound distributed team organization. Considering this stress source, we assess respective team resilience by:

- clustering teams according to observed differences in requirements change perceptions across distributed sub-teams, thus indicating the quality of collaboration and information flow across the distance boundary,
- determining performance indicators for each cluster in order to assess their resilience towards requirement changes,
- revealing typical internal project organization by studying collaboration patterns incidence for each of the clusters and thus
- allowing the characterization of team resilience to requirements change according to adopted collaboration pattern and inter-pattern properties.

This paper is an extended version of "Team Resilience in Distributed Student Projects", published at the International Conference on Global Software Engineering (ICGSE 2018) [11]. In this extended version, we further investigate our initial findings, including the definition and detailed analysis of project teams' collaboration patterns on project-wide and individual team member levels. The analysis starts with a general study of the occurrence of each educationally desired or undesired pattern in the student projects, being extended to the analysis of individual collaboration roles for each non-contributing team member, as well as the analysis of occurrence of project-wide collaboration patterns in teams depending on the number of non-contributing team members. We also extended our previous analysis on an external stress factor – the perception of requirements change – by clustering projects and thoroughly analyzing their performance indicators in relation to the collaboration patterns observed. In order to extend the original analysis, the units of analysis in this paper are the same as in the conference paper – the course projects from the year 2012 to 2016.

The remainder of the paper is organized as follows. Section 2 depicts the most relevant related literature in terms of team resilience, both in general and in distributed settings, collaboration patterns and distributed development courses similar to DSD, which is described in detail in Section 3. Section 4 briefly describes collaboration patterns, pattern identification method and educational value of collaboration patterns. In Section 5 we describe our research method as a longitudinal case-study and also argument on the potential threats to the validity of our study method and results. The results of the study are presented and discussed in Section 6. The discussion on educational implications and recommendations is presented in Section 7. The paper is concluded with a summary and a description of possible future increments in Section 8.

2. Background and Related Work

In this section we introduce the three main elements of our study: team resilience (the concept under study), collaboration patterns (tools for studying team resilience), and global/distributed software engineering courses (the scope of the study). For each of the three elements we provide a dedicated subsection with background and related work.

2.1. Team Resilience in Software Engineering

Various efforts have been made to define and study resilience, mostly in the psychology field. In general, resilience is the capacity to rebound from adversities strengthened and more resourceful [49]. Starting from an individual or personal resilience, it can be described as an adaptive system which enables a person to “bounce back” from a setback or failure [15]. In a similar manner, team resilience can be conceptualized as a team’s belief that it can absorb and cope with the strain, as well as a team’s capacity to cope, recover, and adjust positively to difficulties [9]. In software engineering, team resilience is regarded as “the capacity of a team to react and recover quickly from problems/crises”, definition polished by Diegmann and Rosenkranz [19], but originally defined by Meneghel et al. [35] in a non-software engineering context. In their work, Diegmann and Rosenkranz highlight how recent research in organizational psychology exposes team diversity and psychological safety to be core factors for team performance and resilience in agile software engineering. Based on that, they propose a model and research design to investigate the effects of team diversity, psychological safety, and social agile practices on team resilience and team performance in agile collaborative software development.

Research on resilience brings good news: competencies for team resilience can be improved. Coutu [15] describes seven “streams” of behavior used to improve team resilience, shortly introduced as: *community*, *competence*, *connections*, *commitment*, *communication*, *coordination* and *consideration*. Mallak [34] proposes 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 Alliger et al. list around 20 team resilience challenges, but also 40 practical behaviors of resilient teams, grouped into Minimize (Before), Manage (During) and Mend (After) categories related to the period of adversity.

In software engineering context, Amaral et al. [2] surveyed 115 members of teams working on software-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. The influence of process flexibility was the center of Maccormack and Verganti’s [32] study, who 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. They 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 occurs.

Sharma et al. [46] report on the great importance of resilience for teams working in complex and diverse environments. Besides existing studies on team resilience, they highlight the lack of reliable and valid scale to measure team resilience in the literature. To fill this gap, they design and develop a reliable and valid measure to assess the resilience capacity of collaborative development teams. They find out that team resilience is a hierarchical and multidimensional scale comprising of four primary dimensions along with ten sub-dimensions.

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. As concluded by Hinds and Mortensen [23], spontaneous communication plays an important role in maintaining team resilience, with direct effects in conflict management. Stahl et al. [47] observe that diversity in multicultural teams is studied with a specific focus on its negative effects on team performance and resilience. They take another perspective to investigate whether and how such diversity can instead become an asset to improve the team's performance and resilience; they propose a research agenda based on a collected set of existing literature.

For quantifying team resilience and identifying possible points of risk, Xiao et al. [53] propose an automatic approach to visualize team hierarchies. They apply their approach to six Apache open-source projects to show its effectiveness.

Summarizing, team resilience has been studied with various nuances in different domains and disciplines providing suggestions about possible sources of stress and consequent mitigating actions. In this paper, we focus on a specific context, a university distributed software development course, and we analyze the impact of two sources of stress, non-contributing team members and changes to customer project requirements, considering the period 2012-2016.

2.2. Collaboration Patterns

When analyzing team dynamics, a helpful method is to study collaboration patterns represented by sociograms, graphical representations of a social network created in a team. Several works use social network analysis as a means for determining relations in the team. The studies are performed in various settings: local and distributed, industrial and academic.

Damian et al. [17] observed the development of a main product feature, in a distributed sub-team (part of a larger project), where six developers from two sites (the US and Canada) agreed to provide their data. The social network of interactions was formed, and over the course of 19 days, some interesting findings were made. Social networks evolved in time, and interactions dramatically changed throughout the development phases. Those interactions were "almost completely unrelated to those documented in the project proposal."

Hinds and McGrath [22] found, based on the research of one multinational company, on 33 research & development teams (16 collocated, 17 distributed), that the informal hierarchical structure works better than a fully flat, flexible organization. In their case it was also found that dense communication was related to more - instead of less - coordination problems. However, in contrast to their findings, which showed that the efficiency of a distributed team is ensured by "point people" through whom much of the team communication flows to the remote teams [22], Nguyen et al. [37] find that having a large core of multiple active members of each sub-team, connecting their teams in other locations, reduces possible communication bottleneck problems and introduces redundant communication channels, enabling fast communication.

Cataldo and Herbsleb [10] performed a longitudinal study of collaboration patterns in a distributed SE project, lasting for 39 months, including 114 developers. They found that the core people in the team are not only in charge of communication, being technically competent, but they performed a significant part of actual development as well. Also, these "communication hubs" were even more important in communication across sites, than in local context. The team dynamic changed over time, with people temporarily moving in and out of the "core" hub.

Social interactions are also related to project requirements, as presented by Damian et al. [18]. Their paper presents a case study of a distributed industrial project between the US and Brazil, where authors were studying requirements-centric collaboration and awareness in both local and distributed contexts. The results showed that social interactions were dynamic and different from initial plans, there was considerable cross-site interaction with the main communication reason being related to requirements changes. Interestingly, there was no evidence of reduced awareness of remote team members, although the distance was a factor in remote members' availability.

Moving further from the conventional industry to the open source context, Surian et al. [48] analyzed a snapshot of Sourceforge.Net data using graph theory, extracting topological sub-graph patterns and presenting some of the common topological collaboration patterns appearing in these collaboration clusters of distributed developers. While the open source developers' context is different from ours, it is useful to observe the patterns of communication in a huge community.

Although communication pattern analysis is often used in e-learning context and courses, not many papers focus on software engineering courses. One of these is [26], where Knutas et al. performed the social network analysis of intra- and inter-group collaboration patterns over three "code-camp style" software development courses/projects (each lasting five days). Their findings include observing intensive collaboration outside of defined project teams, discussing the issues with members from other teams based on pre-existing social connections. These patterns also showed that the projects also had a strong "center of collaboration" (core), but such a setting was not beneficial for all groups.

In a more conventional course setting, MacKellar [33] provided sociograms of a big course project with student interactions, finding out how the collaboration patterns influenced the outcome for each project sub-team. At that time, this was still a work-in-progress, so no strong conclusions were being made.

Finally, Paasivara et al. [41] presented the GSE course carried out using agile methods (distributed Scrum), showing using sociograms that the team which had very high scores for teamwork quality, was communicating in a true Scrum manner, having "the most egalitarian communication structure."

The course described in this paper, Distributed Software Development course, was the basis of a previous analysis of collaboration patterns, where Čavrak et al. [13] analyzed 14 DSD course project teams and their collaboration patterns, identifying and visualizing 14 patterns of local and remote collaboration.

The presented studies show how collaboration patterns are a widely adopted way to analyze the communication and team dynamics. In this paper, we adopt, with minor changes, the patterns proposed in [13] to study how they are influenced by non-contributing team members and how they can impact team resilience to project requirement changes.

2.3. Global Software Engineering Courses

In an educational context, there is a number of international courses dealing with the same topic – global/distributed software engineering courses. Clear et al. [14] carried out an extensive literature review of 82 papers about global software engineering courses, published in 2015, providing a list of challenges and recommendations in such education. 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 [39].
- 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 [43].
- 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 [42].

Most research in the educational field of global software engineering has focused on communication and collaboration in this educational context; however, team dynamics – especially team resilience – has not been addressed yet.

3. DSD Course Overview

The Distributed Software Development course (DSD) is a project–based course given jointly by three universities: University of Zagreb, Croatia (FER), Mälardalen University, 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. Students have to deal with several challenges like distance collaboration, language and cultural differences, knowledge transfer, team organization and dynamics, etc. In 15 years, 510 students from 45 countries and 6 continents participated in 77 projects – usually 4-5 project teams per each year. More details about projects and demographics are available in the Results section of this study.

Besides introductory lectures and guest lectures given in the first weeks by industry professionals 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 (15 weeks). All sites participate in every project phase, working together in a synchronous manner, using several collaboration and communication tools to organize their distributed work. Students are explicitly and consistently informed on the importance of conducting a proper (distributed) development process, and that delivering a well-working final product without a well-conducted process will not be rewarded with high course grades.

Throughout the DSD history, there have been several types of projects and project customers, presented in [5], but in recent years our main focus is having projects with external industry partners, as well as ICSE SCORE[2] 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.

The project topics are suitable for 6-8 team members, big enough to experience the teamwork challenges but small enough to be feasible. They are usually complex web applications, often including the additional mobile application. In some instances, projects include specific elements like hardware sensors or big data sources. Some of the project examples are:

- visualization of world air quality open data obtained by satellites
- an application for the organization of cycling and running races, with real-time tracking of participants
- a system for easier garage parking using Bluetooth beacons, etc.

While customers have a general project idea and an initial general set of requirements, during the discussions in the requirements gathering phase, students and

[2]ICSE SCORE competition: `http://score-contest.org/`

customers jointly decide on the list of stories and functionalities to implement. Both sides could ask to change the requirements during the project, due to several reasons (misunderstandings, lack of time/resources, project focus shift, better insight into the customers' needs, more knowledge of technology constraints, etc.). In general, such changes are not numerous nor too dramatic, but students' perceptions of the changes can be different, depending on the team dynamics, as is described in this paper.

In the first decade of course delivery (2003 - 2013), our projects followed the iterative methodology, with the students taking over the roles of Project Leader and Team Leader. Starting with the year 2014, all student projects leverage an agile development methodology, specifically Scrum framework. Teams follow the usual SCRUM rules, divide the work in Sprints, produce artifacts like Product and Sprint Backlogs, organize different types of meetings, etc.

During the course, students, in general, invest 100-150 working hours in the project. Besides delivering the interim versions of documentation and product, the teams present their ongoing work in 6 distributed presentations, covering the different project phases and deliverables (Project Plan, Requirements and Design, Project status presentation, Alpha and Beta prototype, Final presentation).

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. Regular meetings among supervisors and team members, in general, take place every week, enabling teaching staff to track the team's progress, especially in the process part, particularly important for our course.

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. The template of the Final evaluation table, including all the elements, is available at our website[3] 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 the distribution of points, but also the 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 (e.g. technology used in the course, students' motivation, project types, customers' involvement, etc.) are provided in other papers; the list of publications is available at our course website[4]. The complete project archive, including all

[3]Templates of data sources: `https://www.fer.unizg.hr/rasip/dsd/research_data_sources`

[4]DSD course website: `https://www.fer.unizg.hr/rasip/dsd`

presentations and deliverables, is available on the same website as well.

In comparison to the conventional course methods, this method of education – using project-based courses, especially distributed and "tightly-coupled" among course partners – brings along a number of additional, wider risks described in [6], related to course organization or project organization. Some of these risks are related to team resilience, as highlighted in this paper.

4. Collaboration Patterns in the DSD course

In this section, we briefly introduce collaboration patterns, used to characterize the true organization of a distributed student team and collaboration among team members. The description of the pattern identification process and the set of identified principal collaboration patterns was originally published in [13]. We start this section by providing the rationale for using collaboration patterns as a means of gaining insight into distributed team dynamics, then describe the process used to identify principal collaboration pattern within a project team and conclude it with the discussion of educational desirability of different collaboration patterns.

4.1. Team Dynamics and Collaboration Patterns

Gaining insight into team dynamics, collaboration intensity and information flows of a distributed development team, especially in the educational context where many of the environment variables (such as team composition, individual motivation, communication pathways and tools, etc.) are not easily controllable, can be a very challenging task. One of the feasible approaches is to conduct a post-mortem analysis of collaboration intensity among team members based on their feedback and reconstruct it in the form of a sociogram. Further analysis of such sociograms, acquired from a large number of projects, can reveal repeating graph (sub-)structures and typical individual roles, allowing the identification of representative collaboration patterns at the project level and collaboration roles at the level of individual team members.

In our collaboration pattern model *principal patterns* characterize distributed project-level collaboration and necessarily involve team members from both locations; principal patterns reflect distributed team organizational structure and information pathways between distributed sub-teams. *Collaboration roles* address individual team member roles within a project team, either in a local or a distributed context.

By systematically collecting, storing and analyzing data collected from the DSD course student projects since the year 2009, we have identified seven principal collaboration patterns and nine individual collaboration roles. Principal collaboration patterns and their properties are detailed in Table 1 whereas individual collaboration roles are described in Table 2. A simplified visual representation of principal patterns and individual roles is given in Figure 1.

Table 1: Principal collaboration patterns [13]

Pattern	*Description*
Backbone	There exists a single significant collaboration link between distributed teams; all information between locations is exchanged using only one communication channel and two mediators.
Core	Represents the existence of a collaboration nucleus within the project, where nucleus members are from both locations and are mutually well connected by significant collaboration links.
Sandglass	Experienced in projects where two Triangle patterns exist, symmetrical to the distance gap between remote teams.
Split	Implies a division of a distributed project team into two (or more) sub-teams, where each of the sub-teams retains a distributed nature. Within each sub-team, there exists a significant collaboration over the distance gap, but there is a lack of significant collaboration between members of different sub-teams.
Star (principal)	All significant collaboration links within a project focus in only one project member, regardless of his location. The pattern represents a complete breakdown of local collaboration and the existence of a project *master* that is a single source of coordination among distributed non-cooperating individuals.
Triangle	A constellation of significant collaboration links where the most team members from one location express collaboration links with only one team member from the other location.
Virtual team	Represents a project team with significant collaboration between team members regardless of their location. The criteria for determining the existence of a Virtual team pattern are: a) there exists a subset of team members from both locations with at least 80% of possible collaboration links existing among them, b) such a subset is composed of at least 70% of all project members, and c) all subset members express collaboration links with at least 50% of subset members from remote location. Project teams failing only in criterion b) are considered to exhibit a Core collaboration pattern.

Table 2: Individual collaboration roles [13]

Pattern	*Description*
Island	Represents a complete lack of significant collaboration links of an individual member with any of the other project members. Lack of collaboration does not automatically imply lack of contribution, only a complete absence of proactivity and will to participate in collaborative project activities.
Local	Significant collaboration links of a team member are restricted to other local team members, no significant collaboration over the distance gap.
Loose	Individual team member with only a few local and remote collaboration links.
Parallel	A team member is a member of a minor subgroup of a Split principal pattern.
Core	A team member is a member of a Core principal pattern.
Virtual Team	A team member is a member of a Virtual Team principal pattern.
Backbone	A team member is a member of a Backbone principal pattern.
Star	A team member is the focus of a Star principal pattern.
Proxy client	A team member "attached" to the project team by having a single significant collaboration link with a well-connected local team member (usually classified as another individual pattern) acting as a proxy.

4.2. Pattern Identification

Pattern identification method follows the method described by Čavrak et al. [13], with several minor adjustments to accommodate for changes in the DSD course organization and resulting data properties. The main source of data required to construct the collaboration patterns are students' Final Questionnaire documents (subsection 5.5), specifically the questions related to intensity of collaboration with local and remote team members:

> *Describe the intensity (not quality!) of collaboration with each of the local team members with the number 0-5; 0-no contact at all, 5-very intensive collaboration*
>
> *Describe the intensity (not quality!) of collaboration with each of the remote team members with the number 0-5; 0-no contact at all, 5-very intensive collaboration*

Resulting data, collected from all team members and grouped at the project level, is represented in the form of a directed weighted graph describing the perception of collaboration intensity within a project team. Corresponding adjacency matrix provides the input to the pattern identification process, where only the significant collaboration links among team members are retained and team member *reputation* is assessed based on the asymmetry of collaboration intensity perceptions. For each project, a sociogram is constructed, visually revealing principal collaboration pattern, as well as roles of individual team members within those patterns. As the final step of the identification process, the resulting sociogram is compared to the catalog of already identified patterns and their key properties, selecting one that most closely matches, or a new pattern is introduced in the catalog. Major criteria for identifying a principal pattern are: *(i)* the number of significant collaboration links across distance gap dividing two project sub-teams, and *(ii)* number and location of team members in a strongly connected graph component. A more detailed description of the identification process can be found in [13].

Since the original pattern identification process and the corresponding collaboration patterns were defined only with the iterative development process and hierarchical team organization in mind, currently used process has been adjusted to accommodate both hierarchical- and Scrum-based projects. The original process treated all collaboration links with average weight greater or equal to 3 as significant; the updated process requires this link weight to be greater or equal to 3.5. This change stems from the far greater number of border-strength collaboration links found in Scrum-based projects but only marginally changes the interpretation of collaboration patterns in the older, hierarchical-based projects. In addition, collaboration strength perception asymmetry is given more significance in the process, eliminating all collaboration links with asymmetry larger than one from a set of significant links. Team member *reputation* is determined based on her systematic over- or under-estimation of collaboration link weights towards other team members, adjusting her link weights accordingly.

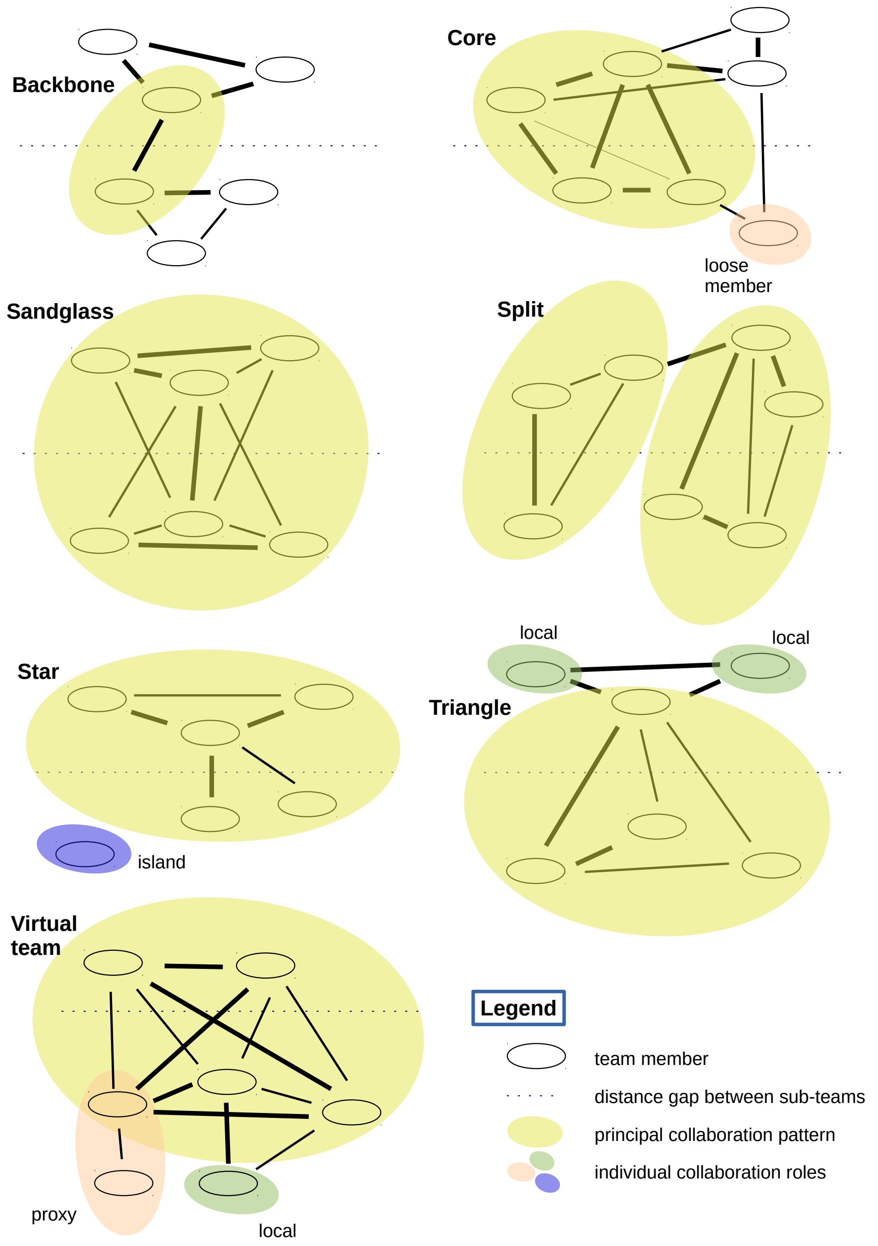

Figure 1: Examples of principal collaboration patterns and individual collaboration roles [13]

4.3. Educational Value of Collaboration Patterns

Given the optimal size of distributed student project teams (6-8 students), and educational goals emphasizing the quality of development process over the quality of final product, we expect the emergence of *educationally desirable* collaboration patterns in students' distributed projects. Such patterns should reflect intensive coordination and information flow at both local sub-teams and between remote sub-teams, as well as promote inclusiveness of all project team members.

A project's *principal collaboration pattern* should ideally reflect one of the two types of *educationally desirable* distributed team *organization*: *virtual team* or *distributed team. Virtual team organization* is characterized by strong collaboration among all team members, regardless of their location and project role. We consider Virtual team principal collaboration pattern as the representative of the *Virtual team organization. Distributed team organization*, on the other hand, characterizes teams manifesting strong local collaboration and limited, but sufficient, communication links between remote teams. Representative patterns for this *team organization* are Backbone, Triangle, and Sandglass. *Educationally undesirable* collaboration patterns – Core, Split, and Star – mirror team weaknesses in local and/or remote collaboration. Split and Star patterns are considered especially undesirable since they portray the project team's complete failure to act as a coherent group (Split) or severely degraded local and remote collaboration (Star).

Individual collaboration role for a productive team member should portray her participation in a project's principal pattern or at least some level of collaboration with one or more team members from both locations. We consider Core, Virtual Team, and Backbone roles as positive ones. Local, Loose and Parallel roles are considered neutral, while Island, Star and Proxy client roles are treated as negative.

5. Research Formulation and Study Design

In this section, we first address our study design and its main phases. In the remainder of this section, we provide details on the core components of the study design: we formalize research objective and questions, define the case and units of analysis, and describe data collection, data analysis and limitations of the study.

5.1. Study Design

In this paper we report on a *longitudinal case-study*, designed as follows according to the guidelines by Runeson et al. [45]. The three main phases were: planning, conducting and documenting.

Planning. The objective of this phase was to formalize the research objective and the research questions, as well as to identify the case and units for analysis, the relevant data sources (among all data stored throughout the years) and potentially interesting data analyses.

Conducting. In this phase, we performed the study itself by following two main steps:

- *Data collection*: we investigated stored data and thereby isolated and extracted relevant data for the intended analyses.

- *Data analysis*: during this activity we analyzed and summarized the extracted data with the aim of answering our research questions.

Documenting. In this phase we elaborated the analyzed and synthesized data, and performed an accurate analysis of possible threats to validity. Eventually, we wrote this paper, which describes the performed study.

5.2. Research Objective

Our goal is to reason on team resilience and how two core stress factors – non-contributing team members in a distributed project environment and changes of project requirements[5] – influence the project performance of student teams. We identify the collaboration patterns of distributed student teams, and analyze their educational desirability. In addition, we seek possible correlations between team collaboration patterns, resilience and stress factors.

As per definition of a case-study [45], we do not attempt nor claim generalization of the findings to other cases with different characteristics (e.g., industrial distributed projects), but rather focus on the similar cases, that is to say distributed student team projects in regular courses (within one study period).

5.3. Research Questions

To achieve our objective, we designed a longitudinal case-study with the aim of answering the following research questions (RQs):

– RQ1a: *What is the resilience of distributed student teams to non-contributing team members?*

– RQ1b: *How do non-contributing team members influence project team collaboration patterns?*

– RQ2a: *What is the resilience of distributed student teams to project requirements change?*

– RQ2b: *How do different collaboration patterns influence team resilience to project requirements change?*

[5]Note that the focus is NOT only on the requirements specification phase. In fact, we analyzed how changing requirements throughout the entire development process, AFTER the requirements phase, affects resilience.

5.4. Case and Units of Analysis

The investigated case is the Distributed Software Development course. We had five units of analysis, represented by the analyzed instances of the course during the period 2012–2016. Moreover, since we also analyze at a finer granularity than course instance level, we had 23 subunits of analysis, represented by the analyzed projects across all five course instances. In each (sub)unit of analysis, our aim was to identify the potential impact of specific stress factors on the resilience of student teams.

5.5. Data Collection

We collected data according to the four principles by Verner et al. [51]: use multiple data sources, create a database with them, validate data and maintain a chain of evidence. For data collection we leveraged the *independent method*, i.e. based on documentation analysis, as introduced by Lethbridge et al. [30].

A large volume of data is generated during one DSD course instance, at different time periods (before, during and after the project work), as part of different documents and by different actors (individual students, project teams and teaching staff). All the documents are in a structured or semi-structured format, and have undergone only minor changes in format and content over the years. Produced documents are systematically collected and archived, providing a stable data source for conducting longitudinal studies.

Data necessary for conducting this study is extracted from the following data sources (templates available at our course website[6]:

Project Evaluation Forms. Reports created by teaching staff as a result of the evaluation process. Each project is evaluated on the basis of more than 50 criteria, divided into four major criteria groups: *(i)* documentation quality, *(ii)* presentation quality, *(iii)* process quality and *(iv)* product quality. Each criterion is assigned a corresponding weight, reflecting the importance of that criterion for the overall project grade. Criteria groups have the following relative importance in the overall project evaluation, based on cumulative weights of contained criteria: *documentation quality* - 22%, *presentation quality* - 10%, *process quality* - 33%, and *product quality* - 35%. During the evaluation process, each evaluation criterion is awarded a grade 1 (low) – 5 (high), reflecting the quality of the evaluated project work aspect. Overall project performance and per-criteria-group performances are expressed as the percentage of points awarded – calculated as the weighted sum of criterion grades – compared to the maximal number of overall or per-criteria-group points.

Final Questionnaires. Final Questionnaire reports are individually filled out by DSD students, after all the project activities are over. The aim of this questionnaire is to allow students to record a "guided" retrospective of their project work by posing a set of open and closed questions. The form of the required answers varies from purely textual and unstructured, to highly structured in the

[6]Templates of data sources available at: `https://www.fer.unizg.hr/rasip/dsd/research_data_sources`)

form of Likert-like scales. Major question groups address students' experience of collaboration within the local team, the collaboration between remote teams, perceived cultural differences, work organization, communication tools etc.
Individual Student Grades. Individual student grades are a combination of their project performance (as perceived by their teachers and their teammates) and the quality of their Final Questionnaire. Although each institution may (or does) in the end locally use different grading scales, a scale of 1–5 is used in our joint evaluation process for the individual student, where 5 represents the highest grade, and grade 3 being the lowest grade for passing the course.

5.6. Data Analysis

To assess the impact of internal or external stress source on project performance, we use three project performance indicators: *overall project performance*, *process performance*, and *product performance*. All three indicators are extracted from *Project evaluation forms* as values of overall project quality, process quality and product quality, their values expressed on a percentage scale 0–100%. The *overall project performance* indicator collectively measures all project aspects (product, process, documentation, and presentation quality) and provides an overarching measure of project success. More specific *process* and *product performance* indicators are used to reveal potential differences in how project teams react to different stress sources – was the team's effort more on delivering a functional product or on adhering to the required development process. In the studied context, prioritizing process over product quality brings far more educational value for students and is the preferred reaction of student teams to stress, than resorting to ad-hoc processes and organizational structures in order to finish a functional product.

We classify project team members to *contributing* and *non-contributing* based on the received individual grade, extracted from *Individual Student Grades* reports. *Non-contributing* team members, classified as such by receiving final project grades 3 (barely passing the course) or 1-2 (failing the course) effectively reduce the *nominal* team size after the project kick-off. Those students usually restrict their activities to only dealing with project documentation and presentations, or are isolated by other team members as unproductive in the technical part of the project due to their insufficient technical/educational background.

To study the possible effect of team size on project performance, we classify projects according to the number of team members into three classes: *undersized* (< 6 students), *optimally sized* (6–8 students) and *oversized* (> 8 students). In addition, we observe this classification for *nominal* and *effective* team sizes. *Nominal* team size denotes the number of students allocated to the project team at the beginning of the project. *Nominal* team size may deviate from the optimal team size due to imbalances in enrollment on involved institutions, resulting in undersized or oversized teams from the very project beginning. In addition to sub-optimal team sizes, imbalances of team member numbers on two involved locations may occur, posing additional risks for effective distributed work. *Effective* team size denotes the number of *contributing* team members in project and is always equal or less than project's *nominal* team size.

The following three subsections (5.6.1 – 5.6.3) define the three major groups of analyses conducted in order to address the posed research questions, with analyses results presented in subsections 6.2 – 6.4 respectively.

5.6.1. Collaboration Patterns and Project Performance

For each of the projects under study, we identify principal collaboration pattern and individual collaboration roles, employing the pattern identification process briefly described in Section 4.

To gain an overall view on the presence and impact of collaboration patterns, we conduct the initial analysis of *(i)* principal pattern incidence within the set of studied projects, *(ii)* principal pattern incidence within two sets of projects employing different development methodologies (i.e., iterative and Scrum, as described previously), and *(iii)* project performance indicators for each of the identified principal collaboration patterns. By pattern incidence analysis we are examining the relative occurrence of educationally desirable and undesirable principal collaboration patterns in the analyzed population of distributed projects and identify dominant ones. As the mandatory development process methodology was changed in 2014 from iterative to agile (more precisely, Scrum framework), we also analyze whether this change affects our data analysis results, in particular how the collaboration patterns change in moving from one methodology to another.

Finally, we examine the distribution of project performance indicators (*overall*, *process*, *product*) for each of the identified principal patterns within the project population. Resulting distributions should reveal median project performance and its variability for projects employing a specific principal collaboration pattern. Observed performance allows comparison of performance indicators among different principal patterns, as well as comparison among *overall*, *process* and *product* performance indicators for a single pattern. Results of this analysis should confirm or deny our perception of educational desirability or undesirability of specific patterns, as well as the impact of the principal pattern on process and product performance aspects.

5.6.2. Resilience to Non-Contributing Team Members

Resilience to internal stress in the form of non-contributing team members is assessed by observing the effect of the number of (non-)contributing students on project performance indicator distributions, and on occurrence of principal collaboration patterns that affected student teams tend to adopt.

Non-contributing students are divided into two classes: those finishing the course and receiving lower grades and those leaving the course without receiving a grade and not providing feedback (e.g., not submitting their Final Questionnaire). The option of leaving the course is institution-dependent and student type dependent – for example, exchange students can drop out much more easily than regular students – influencing students' motivation to participate in project work actively.

We start our analysis by observing the occurrence of non-contributing students in the analyzed projects, examining non-contribution and dropout rates among regular and exchange student sub-populations, and identifying collaboration roles non-contributing team members tend to adopt within their project teams. The impact of non-contributing students on project performance is assessed by studying the performance indicator distributions of projects grouped by the number of non-contributing students present in project teams. Incidence of principal patterns per project groups is revealed with the aim of identifying principal patterns the project teams tend to adopt, depending on the number of non-contributing team members.

Next, instead of the number of non-contributing project team members, we take into account the number of contributing team members (effective team size). We analyze the distribution of project performance indicators for teams grouped by the effective team size.

In the end, we observe differences in project size classification when nominal and effective team sizes are considered. Consequently, we study the impact of those differences on project performance indicators, as well as the occurrence of project principal collaboration patterns.

5.6.3. Resilience to Project Requirements Changes

Student project team resilience to requirement changes is assessed indirectly, by observing the effect of differing requirement change perceptions between remote sub-teams on project performance indicator distributions. To identify projects with satisfactory and low resilience, we group them into three clusters. In addition, we identify dominant principal collaboration patterns within each cluster and assess pattern impact in shaping team resilience.

Clusters were manually formed according to values of two variables:

- *Requirements change perception* — mean value of requirements change perception of all project team members (values 1–5); reflects the magnitude of requirements change as perceived by the whole project team, but not necessarily the same for remote sub-teams.

- *Difference in requirements change perceptions* — the difference between mean values of requirements change perceptions per distributed sub-team (values 0–4); reflects the difference in perception of remote sub-teams, mostly due to lack of communication and/or coordination between remote sub-teams of a distributed project.

Perceived requirement change data are extracted from students' *Final Questionnaire* documents, specifically from the students' qualitative assessment of requirement change magnitude submitted as answers to the following question:

> *Rate (1-no or minor changes, 5-huge changes) and describe the level of requirement changes during the project*

We define three clusters – A, B and C with the following semantics:
Cluster A encompasses projects with the *non-existent or minimal difference in requirements change perceptions* between distributed sub-teams, while project–wide mean requirements change perception 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 requirements change stress, and remote sub-teams acting in synchronization.
Cluster B represents projects that expose *significant differences in requirements change perceptions*. The true magnitude of the requirements change stress for those teams cannot be reliably assessed (due to the difference between perceptions of sub-teams), so mean value of all team member perceptions 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 represents projects with *minimal-to-moderate differences in requirements change perceptions*, but with *moderate-to-high requirements change stress* (value between 2.5 and 5). These teams presumably had a clear joint view on the project state (therefore presenting small differences in perceptions) but needed to deal with significant requirements changes.

Distribution of project performance indicator values is analyzed for projects grouped into aforementioned clusters, revealing their typical overall response to stress, as well as specific responses in domains of the process and product quality. We also examine and compare per-cluster effective team sizes and employed development methodologies, detecting their possible influence on analysis results.

The incidence of principle collaboration patterns per cluster is analyzed and the most representative patterns for clusters B and C are selected for deeper analysis. To further investigate the observed effect of smaller project team *nucleus*, for each cluster B project we analyze the *difference in requirement change perceptions* between *nucleus* and all team members.

5.7. Threats to validity

We assess the threats to the validity of the reported study guided by the model proposed by Wohlin et al. [52] for empirical studies in the field of software engineering. We focus on threats to the following three validity categories: *construct* – validity of data sources used in this study, *internal* – validity of analysis process and results, and *external* – a generalization of the results outside the scope of our DSD course.

Neither of the seven data sources was designed for conducting a specific type of empirical research study, but to assist in different phases of the student projects. 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 Questionnaire has an 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, collaboration pattern 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 considering their relevance within the context of other distributed software engineering courses. In this case, its relevance primarily depends on the project framework used in the course: results could be highly relevant to courses utilizing the full development cycle. However, other factors could affect the results of the replicated studies in seemingly similar environments: differing educational background, enrollment policies, cultural background, and work ethics.

As per definition of a case-study [45], we do not attempt nor claim generalization of the findings to other cases with different characteristics (e.g., industrial distributed projects), but rather focus on the similar cases, that is to say, distributed student team projects in distributed courses. 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 [12]. Generalization to cases with different characteristics, such as industrial/real-world projects, is not sought nor claimed, and in general, would not hold due to idiosyncratic differences with educational context and settings.

6. Results and Discussion

In this section, we present the results of the analysis performed according to the study design described in Section 5 and discuss them in the context of the four research questions. First, in Section 6.1 we provide basic properties of the analyzed data covering the period 2012–2016 of the DSD course, followed by the assessment of project performance indicators for each of the identified principal collaboration patterns in Section 6.2 (as defined in Section 5.6.1). Sections 6.3 and Section 6.4, (with respective methods described in Section 5.6.2 and Section 5.6.3) lay out analysis results of student teams' resilience to internal and external stress factors, providing a foundation for addressing the posed research questions. Finally, research questions are answered in two Highlights frames, based on the analysis results provided in sections preceding them.

6.1. General Results

Table 3: 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

Data analyzed in this paper stems from 23 distributed student projects conducted in academic years 2012 – 2016 as part of our DSD course. The total number of students attending the course during the analyzed period was 156, coming from 32 countries and five continents (Table 3). Institutional distribution of those students was as follows: FER 60, MDH 56, and POLIMI 40 students. The data for this analysis was therefore acquired from 156 students’ Final Questionnaires, 23 project evaluation forms and individual students’ grades.

As a result of relatively balanced numbers of enrolled students on three involved institutions, in the analyzed time period there were only two occurrences of undersized and two occurrences of oversized project teams, while a strong majority of projects (19 projects – 82%) were of optimal size.

We further assessed the analyzed projects’ *cultural diversity* – by identifying the number of unique UN sub-regions[7] from which the project team members originate – and *educational background diversity* – by estimating the number of unique educational institutions from which project team members come. Obtained results suggest that the analyzed projects were characterized with a low-to-moderate cultural diversity of their members (team members originating from 2 or 3 different regions – 78% of the projects) and a low-to-moderate educational diversity (team members originating from 2 or 3 educational institutions – 74% of the projects). Therefore, we conclude that the cultural and educational diversity present in the analyzed project teams is of an expected magnitude and should not negatively affect the results of further analyses.

Usage of iterative development methodology and hierarchical team organization were mandatory for ten projects conducted in years 2012 and 2013, while Scrum methodology and “flat” team organization had been employed in 13 projects in years 2014 – 2016.

6.2. Collaboration Patterns and Project Performance

The incidence of principal collaboration patterns within the set of analyzed projects is presented in Table 4 , where the patterns are grouped by their *ed-*

[7]`https://unstats.un.org/unsd/methodology/m49/`

Table 4: Incidence of principal collaboration patterns

Principal Pattern	*Projects*	*Percentage*	
Virtual team	7	30.4%	*desir-able*
Backbone	3	13.0%	
Triangle	2	8.7%	
Sandglass	1	4.3%	
Core	7	30.4%	*unde-sirable*
Star	2	8.7%	
Split	1	4.3%	

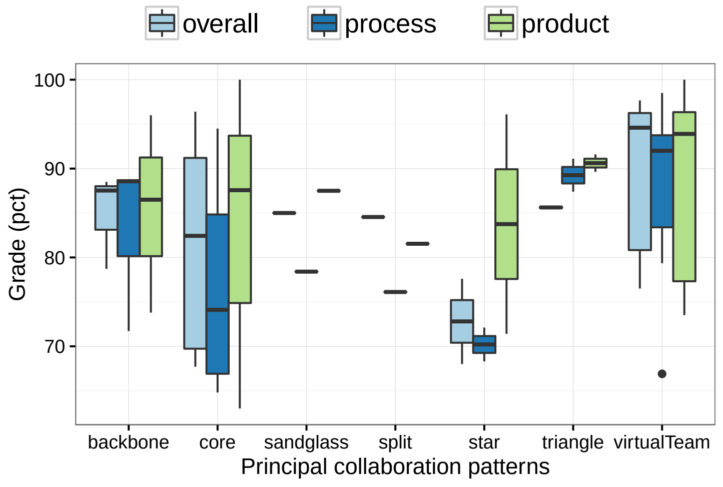

Figure 2: Performance indicators of principal collaboration patterns

ucational desirability and ordered according to their overall incidence. Two similar patterns, *Core* and *Virtual team*, are present in more than 60% of the projects. If observed from the perspective of educationally *desirable* or *undesirable* principal collaboration patterns, 57% of the projects employed *desirable* and 43% *undesirable* collaboration patterns. Analysis of pattern incidence per team's development process revealed no significant differences: for projects using iterative development method and hierarchical organization, five out of ten projects (50%) were using educationally desirable collaboration patterns, while for Scrum-based projects mentioned ratio was eight out of thirteen (62%). The same applies to the most represented patterns: *Core* pattern appears in three iterative and four Scrum projects, while *Virtual team* pattern is identified in four iterative and three Scrum projects.

Performance indicators (*overall*, *process*, and *product*) of the projects grouped by identified principal patterns (Figure 2) reveal that the performance of the *Core* pattern is significantly lower than the structurally similar *Virtual team* pattern, on all three performance indicators. Low *process* indicator score of the *Core* pattern is almost comparable to one of the projects characterized by the highly undesirable *Star principal* pattern. Both patterns share the tendency to score significantly higher on the *product* than *process* indicator, thus opposing the educational goals the DSD course insists on. On the other hand, projects leveraging *Virtual team* and *Backbone* patterns show high and balanced *product* and *process* indicator values, effectively representing educationally desirable *virtual team* and *distributed team* project organizations.

6.3. Resilience to Non-Contributing Team Members

In this section, we provide and discuss the results of the analysis addressing the resilience of student teams to non-contributing team members, by following the approach defined in Section 5.6.2. We begin by studying the incidence of non-contributing team members, associated individual collaboration roles and impact on process performance indicators in Section 6.3.1. Correlation between effective team size and project performance indicators is presented in Section 6.3.2. Section 6.3.3 studies the impact of differing nominal and effective project classifications on process performance indicators and resulting principal collaboration patterns.

6.3.1. Non-contributing Team Members

Table 5: Number of projects with n non-contributing team members

	Non-contributing team members in project				
	0	1	2	3	4
Number of projects with n non–contributing members	10	6	2	3	2
Pct. of projects with n non–contributing members	43.5%	26%	8.7%	13%	8.7%

Table 5 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.

Table 6: Incidence of non-contributing students in Regular and Exchange student groups

	Regular	*Exchange*
Total number of students	132	24
Number of non-contributing students	17	10
Pct. of non-contributing students	12.9%	41.7%
Number of dropout students	2	4
Pct. of dropout students	1.5%	16.7%

Table 6 summarizes student types attending the DSD course during the analyzed period; a much higher rate of non-contribution and course drop-out can be observed in the sub-population of exchange students.

Table 7: Non-contributing students and individual collaboration roles

Collaboration	*All students*		*Regular students*		*Exchange students*	
role	#	%	#	%	#	%
Other	5	18.5%	4	23.5%	1	10.0%
Island	10	37.0%	5	29.4%	5	50.0%
Local	5	18.5%	3	17.6%	2	20.0%
Loose member	3	11.1%	1	5.9%	2	20.0%
Parallel	1	3.7%	1	5.9%	0	0.0%
Proxy client	3	11.1%	3	17.6%	0	0.0%

Table 7 lists the occurrences of individual collaboration roles for non-contributing team members. The most frequently observed role in the overall non-contributing student group, as well as in regular and exchange subgroups, is *Island* – denoting detached team members with no significant collaboration links to any of the other team members. The occurrence of this role is especially dominant in the exchange student subgroup. Collaboration roles collectively represented as *Other* denote involvement of non-contributing students in principal patterns (*Core*, *Star*) or lack of any specific role. We ascribe the absence of the *Proxy client* role in the exchange students subgroup to the difficulty of exchange students to quickly create social bonds with at least one local team member in a new project environment.

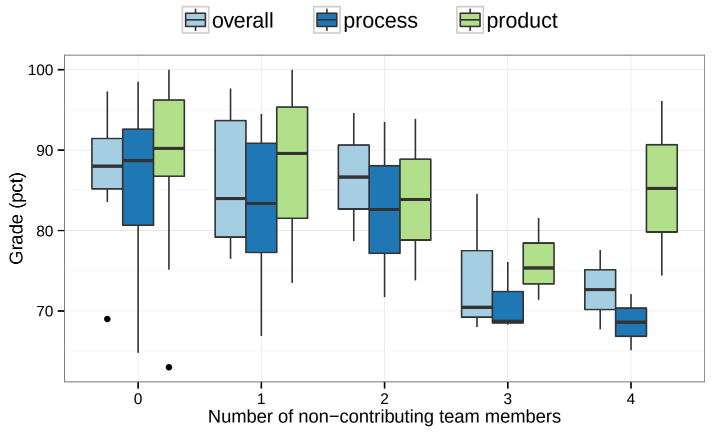

Figure 3: Project performance vs number of non–contributing team members

The effect of the number of non-contributing students on project performance is presented in Figure 3. Project teams having up to two non-contributing team members experience graceful degradation of process and project performance indicators. *Product* score is consistently higher than *process* score, but is degrading much faster than *process*; the number of team members available for product development is smaller, but the distributed team manages to preserve its organization. Substantial degradation of performance indicators is visible for the project with more than two non-contributing team members, where *process* grade experiences a sharper decrease. Such teams are trying to salvage the product resorting to a 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.

Table 8: Number of non-contributing team members and incidence of principal patterns

	Non-contributing team members					
Principal Pattern	0	1	2	3	4	
Virtual team	3	3	1	0	0	*desir-able*
Backbone	2	0	1	0	0	
Triangle	1	1	0	0	0	
Sandglass	1	0	0	0	0	
Core	3	2	0	1	1	*unde-sirable*
Star	0	0	0	1	1	
Split	0	0	0	1	0	

Table 8 confirms that, in scenarios with strong lack of contribution, distributed team organization tends to adopt educationally *undesirable* collaboration patterns. Projects with three and four non-contributing team members exhibit *Core*, *Split* and *Star* patterns – characterized by low *process* grades. On the other hand, projects with less than two non-contributing team members do not exhibit prevalence of *desirable* collaboration patterns: the *Core* pattern is almost as much present as *Backbone* and *Virtual team* patterns. This suggests that a large number of non-contributing students almost guarantees the occurrence of *undesirable* patterns, but the absence of non-contributing ones does not guarantee that *desirable* collaboration patterns will emerge.

Presence of *undesirable* Core pattern in projects with none or a small number of non-contributing students can be explained by its ambivalence towards the individual team member (non-)contribution; only reflecting the existence of a well-connected subgroup taking over the coordination and decision making in the project. Non-core members can either be non-contributing to the project at all, or contributing ones – but their effort is mostly reflected in the *product* quality aspect of the project, not the *process* one (as depicted on Figure 2).

6.3.2. Effective Project Team Size

Table 9 summarizes the data on the number of DSD course projects during the analyzed period with a number of contributing team members.

Table 9: Number of projects with n contributing team members (effective project team size)

	Effective project team size							
	2	3	4	5	6	7	8	9
Number of projects	1	2	3	4	7	2	3	1

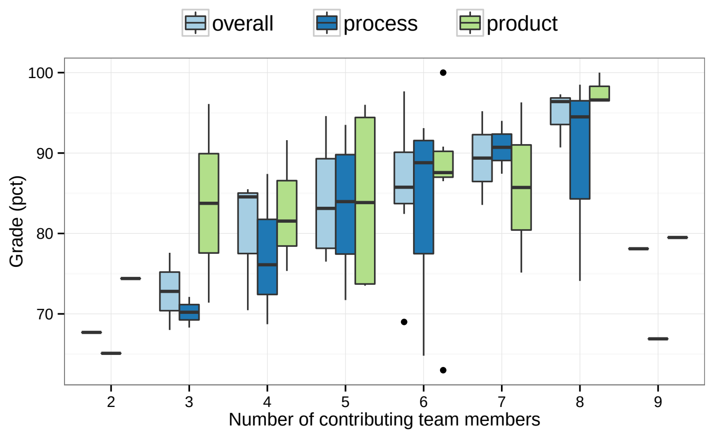

Figure 4: Project performance vs number of contributing team members

Figure 4 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, therefore 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* indicator equalizes with or outgrows *product* indicator value; in larger teams distributed collaboration is (almost) unavoidable, thus emphasizing the need to address process aspect, and, at the same time, lowering team resources for investing in product aspect.

Table 10: Project team sizes (nominal and effective)

Classification	*undersized*	*optimal*	*oversized*
nominal size	2 (8.7%)	19 (82.6%)	2 (8.7%)
effective size	10 (43.5%)	12 (52.2%)	1 (4.3%)
effective - iterative	4 (40.0%)	5 (50.0%)	1 (10.0%)
effective - Scrum	6 (46,2%)	7 (53.8%)	0 (00.0%)

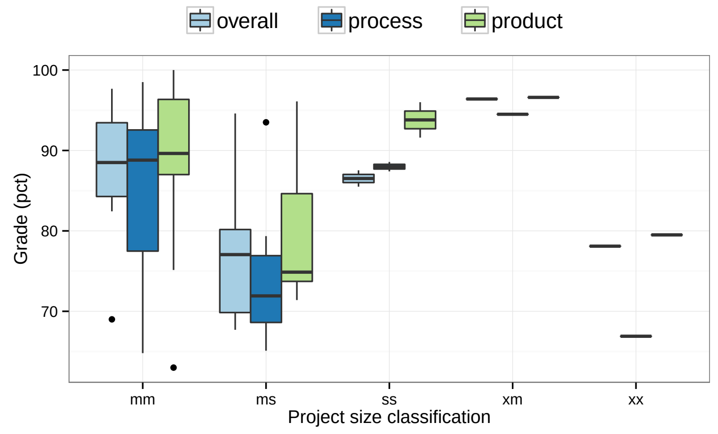

Figure 5: Project performance vs number of contributing team members

6.3.3. Team Size Classification Change

Table 10 summarizes the number of projects within each category when all (nominal) or only contributing (effective) team members are taken into consideration. Large differences between nominal and effective sizes can be observed for undersized and optimal classes, suggesting that a significant number of projects changes their category from *optimal* to *undersized*. Two bottom rows of Table 10 confirm there is no significant difference in effective project size classification between projects employing iterative and Scrum development methodologies.

When observing differences between nominal and effective categorization of analyzed projects, we found that:

- two undersized, eleven optimal and one oversized projects have the same nominal and effective categorization,

- eight projects with nominal optimal categorization change their category to undersized when only effective team size is considered,

- one nominally oversized project is effectively categorized as optimally-sized.

Figure 5 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*); this indicates the existence of a threshold that, when crossed, represents an internal stress source that distributed teams show a low resilience to.

Table 11: Team size classification change and principal patterns

	Size classification change					
Principal Pattern	mm	ms	ss	xm	xx	
Virtual team	4	2	0	0	1	*desir-able*
Backbone	1	1	1	0	0	
Triangle	1	0	1	0	0	
Sandglass	1	0	0	0	0	
Core	4	2	0	1	0	*unde-sirable*
Star	0	2	0	0	0	
Split	0	1	0	0	0	

Table 11 reveals that the overall incidence of *undesirable* collaboration patterns (*Core*, *Split* and *Star*) is more frequent within projects that transition from optimally-sized to undersized. While 64% of the projects that do not change classification (column mm of Table 11) exhibit *desirable* and 36% of them *undesirable* collaboration patterns, projects that do change classification (column ms) adopt *desirable* patterns in 37% and *undesirable* ones in 63% of the cases.

Highlights

RQ1a. What is the resilience of distributed student teams to non-contributing team members?

- ▶ Student project teams exhibit reasonable resilience to up to two non-contributing team members; experiencing a graceful degradation of process and product performance indicators.
- ▶ When resilience threshold is reached, process performance indicator drops sharply; effectively undersized teams tend to focus on product aspect neglecting the (educationally more important) process aspect of the project work.

- ► There exists a relation between the number of contributing team members and the *overall* and *process* performance indicators; no such relation exists with the *product* performance indicator.
- ► The stress of becoming an undersized team has a significant impact on all project performance indicators, whereas being an undersized team from the project start has no such impact.
- ► Exchange students represent a high risk of being non-contributing team members.

RQ1b. How do non-contributing team members influence project team collaboration patterns?

- ► *Island* is the dominant individual collaboration role among non-contributing students.
- ► A large number of non-contributing students almost guarantees the project team will resort to an educationally *undesirable* principal pattern, but their absence does not guarantee that *desirable* principal pattern will emerge.

6.4. Resilience to Project Requirements Changes

In this section, we provide and discuss results of analyses addressing the resilience of student teams to changing project requirements, by following the approach defined in Section 5.6.3. We begin by analyzing project performance indicators for three project clusters in Section 6.4.1. Section 6.4.2 identifies principal pattern incidence in project clusters and correlated performance indicator distributions, further analyzes dependency of performance indicators to collaboration within project nucleus, and discusses implications of projects' principal patterns on resilience.

6.4.1. Requirements Change Perception Differences

Clustering of the analyzed projects, according to criteria defined in Section 5.6.3, yielded cluster sizes A, B and C of 6, 9 and 8 respectively (Figure 6). Figure 7 presents the project success evaluation results for clustered projects. Projects in cluster A (*projects with minimal difference in requirements change perceptions*) score better than projects in other clusters on all indicators, as expected due to a lack of requirement change stress. Interesting differences in evaluation scores can be observed for clusters B (*projects with significant differences in requirements change perceptions*) and C (*projects with minimal-to-moderate differences in requirement change perceptions and moderate-to-high requirements change stress*); projects in cluster B, on average, tend to score higher in the *product* than the *process* indicator and have a better score in the

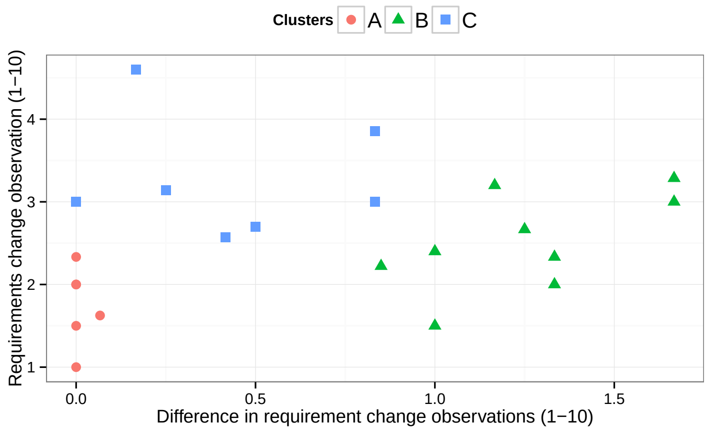

Figure 6: Projects grouped into tree clusters (A, B and C) according to the value of project-wide mean requirement change perceptions and difference between mean requirement change perceptions of remote sub-teams

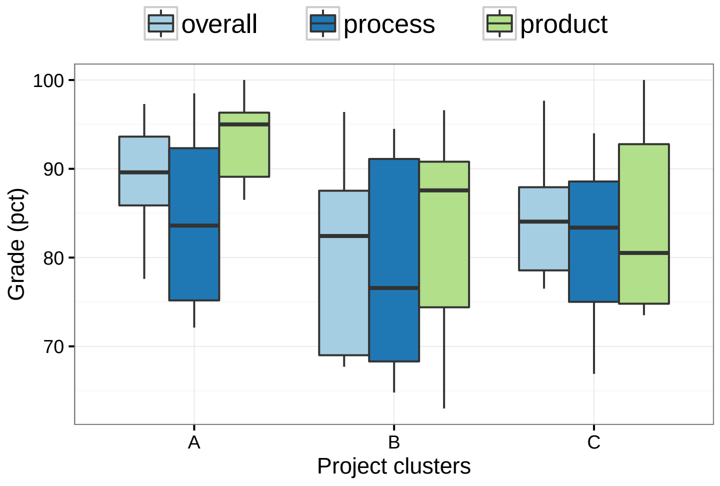

Figure 7: Comparison of project performance indicator values among project clusters

product indicator than cluster C projects. However, they have worse scores and higher score variability than cluster C projects in the *process* indicator.

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* indicator 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* indicator 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* indicator value and by the rather high difference between *product* and *process* indicator values. These teams tend to fragment themselves, not being completely aware of the *overall* project status (resulting in a difference in perceived 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.

Regarding the effective team sizes for projects grouped in three clusters, cluster C exhibits a slightly smaller median team size (5.5) than projects in clusters A and B (median team size 6). We conclude that the effective team size does not have a significant impact on project clustering.

However, interesting results emerge when inspecting the distribution of development methodologies among clusters; 50% of the projects employing iterative methodology are found within cluster A, 30% within cluster B and 20% within cluster C. The distribution of projects employing Scrum methodology is as follows: cluster A – 7.7%, clusters B and C – 46.2%. This implies that the requirements change stress is far more present in Scrum-based student projects, a cause of which we cannot reliably confirm without additional analysis.

6.4.2. Collaboration Patterns and Resilience to Changing Requirements

Table 12: Principal collaboration pattern incidence within project clusters

	Cluster			
Principal Pattern	A	B	C	
Virtual team	2	0	5	*desir-able*
Backbone	1	1	1	
Triangle	0	1	1	
Sandglass	1	0	0	
Core	1	6	0	*unde-sirable*
Star	1	1	0	
Split	0	0	1	

Table 12 reveals the incidence of collaboration patterns per cluster. Only the *Backbone* pattern is present in all three clusters, and the rather surprising result is the appearance of *Core* and *Virtual team* patterns clearly separated into cluster B (*Core* pattern strongly present, no *Virtual team* pattern) and cluster C (strong presence of *Virtual team* pattern, *Core* pattern absent).

We further studied project performance under the requirement change stress, focusing only on *Core* and *Virtual team* principal patterns within clusters B and C, due to the prohibitively small number of occurrences for the other pat-

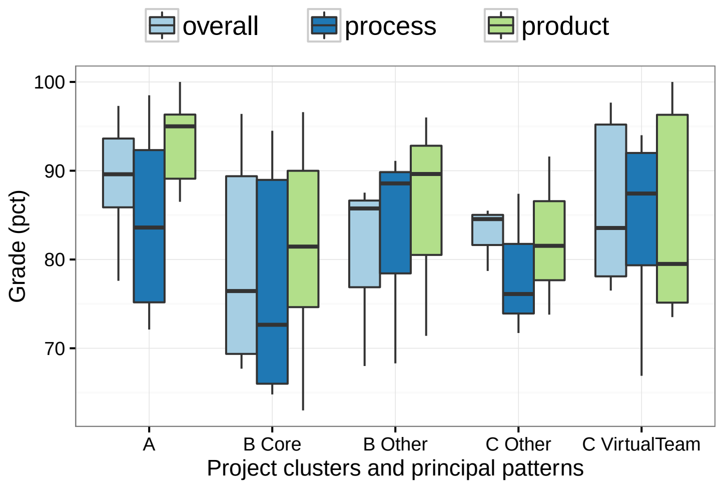

Figure 8: Comparison of project performance indicator values among project cluster A, and dominant and other patterns in clusters B and C

terns. Figure 8 provides a basis for comparison of performance indicators within clusters B and C, by separating performance indicators for projects with dominant pattern (*Core* or *Virtual team*) and other projects present within that cluster. Projects employing *Core* pattern dictate low-performance indicators of cluster B, while good performance indicators of cluster C are dominated by high-performance grades of projects using *Virtual team* pattern. Those results are consistent with the initial performance assessments given in Figure 2; *Virtual team* represents a *desirable* collaboration pattern that reflects well in the team performance, scoring the highest in all performance indicators compared to other patterns. The exclusive presence of *Virtual team* in cluster C strongly suggests that the distributed project teams with such collaboration pattern are also highly resilient to significant changes in requirements. Such resilience can be attributed to the high density of strong collaboration links within the project team *nucleus* and the size of that nucleus compared to the overall team size.

On the other hand, projects employing *Core* pattern, although structurally similar to the *Virtual team* pattern, score lower – especially on the *process* performance indicator. Their confinement to cluster B suggests that smaller project team *nucleus*, compared to the nominal team size, cannot efficiently absorb external stress in the form of changing requirements. *Core*'s *nucleus* size is four team members for all but one project, in contrast with *Virtual team*'s *nucleus*, which varies from five to seven.

Values of performance indicators for cluster B and C projects grouped under *Other* are ambiguous: both groups contain two educationally *(semi-)desirable* patterns (*Backbone* and *Triangle*) with strong average performance indicators (Figure 2), and differ in one *undesirable* pattern (B with *Star*, C with *Split*).

However, performance indicators for cluster B *Other* projects are higher than those in cluster C. We ascribe these results to an overly small number of individual patterns within those groups and leave further investigation for future work if more data collected on those patterns would become available.

A comparison between the *difference of requirement change perceptions* calculated involving all team members and the *difference* calculated involving only *nucleus* team members revealed the following:

- If the calculated *difference* was higher for *nucleus* team members or equal to *difference* for all team members, the project's performance indicator scores were consistent with the *overall Core* pattern indicator scores. The majority of the cluster B *Core* projects (67%) fall into this category.

- If the calculated *difference* was lower for *nucleus* team members than the *difference* for all team members, the project's performance indicator scores were significantly higher than the *overall Core* pattern indicator scores, almost matching the *overall Virtual team* pattern scores.

A similar analysis was conducted on *Virtual team* projects, where only a smaller portion of projects (20%) exhibited higher *difference in requirements change perceptions* for *nucleus* team members than for all team members and, consequently, lower-than-expected performance indicator scores.

Considering the previous findings for projects adopting *virtual team* project organization, represented by *Core* and *Virtual team* patterns, we conclude that the resilience of student teams to the external stress represented by requirements change is primarily dependent on the ability of the team's *nucleus* to handle it: the larger the nucleus, the higher the probability of absorbing the stress. For projects adopting *distributed team* project organization, we are unable to draw firm conclusions due to a too small number of projects adopting its typical principal patterns.

Highlights

RQ2a. What is the resilience of distributed student teams to project requirements change?

- ▶ Cohesive distributed teams exposed to the stress of changing requirements express strong resilience from the process aspect of the project work but tend to sacrifice the product quality aspect (educationally desirable behavior).

- ▶ Non-cohesive distributed teams exposed to the stress of changing requirements exhibit lower-quality process aspect of the project work and focus on delivering a functional product (educationally undesirable behavior).

RQ2b. How do different collaboration patterns influence team resilience to project requirements change?

- ▶ Educationally *undesirable Core* pattern is strongly present within less resilient teams, while *desirable Virtual team* pattern within more resilient ones.
- ▶ The larger the project team *nucleus* with respect to the overall team size and the more uniform perception of requirements changes within that *nucleus* are, the more resilient the team is to requirements changes.

7. Implications to Practice

The results of this study point to a set of recommendations for the practice in educational contexts similar to our case – teaching global software engineering through a distributed project work. Both sources of stress should be tackled before the course begins. The first stress source, effective change in the number of team members (particularly due to non-contributing students) is a factor that is dealt with through team organization and assignment of members to teams. Special attention should be given to exchange students; staff should provide clear information on the nature of the course and the expectations on team members before the course starts. When forming student teams, the risk of this stress source to arise is reduced if the ratio of exchange and regular students is adequate. In this way, possible drop-outs or low-performance of one or two students would not influence the team nor lead to undesirable collaboration patterns. This stress source is much less prominent in industrial settings, as other methods for selecting team members can be applied and the companies possess the means to address unsatisfying performance. In educational settings, teaching staff often doesn't have any influence on students' enrollment to courses, nor they can do much about non-contributing students besides making them fail the course or giving them low grades. However, a sanction for the particular student should not affect the other team members' performance; therefore teaching staff should pay particular attention to the countermeasures they take to reduce this stress source.

The other source of stress, changes in project requirements, is very often found in industrial contexts, as stated at the beginning of our paper. While this is undesirable, it is part of the formative process to allow students to experience such stress. For them to take the most out of it, teaching staff should guide and help them to deal with it in a productive and resilient way. As part of this specific training, ad-hoc discussions about it with the external project customers (from industry) should take place before the course starts. The customer should be ready to add or change requirements, but this should always be done in a thoughtful manner, in line with the educational context of the course.

Another recommendation, helpful in both cases, is the following. Currently, our collaboration patterns are analyzed at the end of the course, as students provide their insights on collaboration only in the Final Questionnaire. This means that we can observe *educationally undesirable* patterns only after the project is completed. While this gives us the knowledge to act accordingly for the future course instances, it would be very beneficial to observe individual and principal collaboration patterns before, e.g., during the mid-term project evaluation. In this way, *educationally undesirable* patterns (or their inceptions) would emerge before and give teaching staff time to steer teams towards one of the equally recommended *desirable* ones.

Finally, we would like to emphasize again that, as such courses are a "taste" of industrial contexts for students, the staff should not attempt to sanitize the project work or over-protect the students, but rather to support them so to make the experiences as useful as possible for their future careers.

8. 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 changing requirements. Team reactions were evaluated by observing changes to project performance indicators – *overall*, *process*, and *product*, resulting from final project evaluations. To effectively analyze project team dynamics, we identified principal collaboration patterns and individual collaboration roles for each of the analyzed student projects and correlated them to other observed project characteristics.

Non-contributing team members, a stress source internal to the team, reduce the project team size and impact its ability to produce a quality product and follow the desired development process. These students take more relaxed project roles, are isolated from other team members due to their inability to effectively collaborate, or present a lack of motivation to participate in the project work actively. Their lack of motivation is often correlated to their status; exchange students represent a high risk of becoming non-contributing team members. Concerning the individual collaboration roles, such non-contributing students predominantly take on the *Island* role, with fewer instances of *Local*, *Loose member* and *Proxy client* roles.

Distributed teams exhibit a fair level of resilience, given that the team size is not reduced so that the distributed development process becomes more of a burden than a tool needed to coordinate the work effort. In such a case, reduced teams focus on the product and neglect the process-related part of the development, hampering the educational goals set before them. In general, projects exhibit a correlation between the number of contributing team members and the quality of the development process, while such an explicit dependency for the final product quality does not exist.

This finding is reflected in the most common collaboration patterns found in student project teams of varying effective sizes: projects with large numbers of non-contributing team members and/or changing classification from optimally-sized to undersized tend to adopt educationally *undesirable* and process-wise low performing principal patterns such as *Core*, *Split* or *Star*. However, such low performing patterns are present in optimally-sized projects in numbers equal to educationally *desirable* patterns, suggesting that the project team size is a necessary, but not the sufficient requirement for achieving our educational goals.

Requirements change, 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 changing requirements have lower product quality, but preserve high resilience in their process quality aspect. Non-cohesive teams, identified by significant differences in perception of requirements volatility between remote sub-teams, tend to focus on the product, experiencing a sharp drop in process quality.

Principal collaboration patterns analysis showed that distributed teams that are cohesive and resilient to requirements changes tend to implement educationally *desirable* collaboration patterns, predominantly the *Virtual team* pattern. However, the structurally very similar but *undesirable Core* pattern is dominant among non-cohesive teams. The results suggest that there exists a minimal project team *nucleus* size, with respect to the nominal team size, required for reaching team cohesion and, consequently, resilience to external stress factors. Furthermore, the same cohesion indicator used at team level – differences in perception of requirements changes – can be used on the *nucleus* itself to predict the performance of the project team.

The ability of a distributed student team to adapt to stress without significantly sacrificing performance is a crucial property in the educational settings of a distributed development course. 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 resilience of distributed teams to external and internal stress sources in an educational context will focus on assessing the impact of additional stress sources and reliable indicators revealing weaknesses in distributed team performance.

This research includes projects realized using two development processes - iterative (used in the first decade of the course) and agile development. As observed, the change didn't affect the elements on which we focused in this study. However, having completed five years of using an agile approach (more precisely, Scrum) in the course projects, we plan to perform an in-depth study to analyze the various implications of moving from iterative to agile development. This will also include a thorough analysis of team dynamics and resilience in both course settings, from a qualitative perspective as well.

Acknowledgements

The authors would like to thank all former and current colleagues from Mälardalen University, Sweden, Politecnico di Milano, Italy, and University of

Zagreb, Croatia, for the joint effort in working on DSD course. This work is partially supported by the Knowledge foundation through the projects MOMENTUM and HERO.

References

[1] George M. Alliger, Christopher P. Cerasoli, Scott I. Tannenbaum, and William B. Vessey. Team resilience: How teams flourish under pressure. *Organizational Dynamics*, 44(3):176 – 184, 2015. ISSN 0090-2616.

[2] António Amaral, Gabriela Fernandes, and João Varajão. Identifying useful actions to improve team resilience in information systems projects. *Procedia Computer Science*, 64:1182 – 1189, 2015. ISSN 1877-0509. 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] A. Avritzer, S. Beecham, J. Kroll, D. S. Menasche, J. Noll, and M. Paasivaara. Survivability models for global software engineering. In *2014 IEEE 9th International Conference on Global Software Engineering (ICGSE)*, volume 00, pages 100–109, Aug. 2014. doi: 10.1109/ICGSE.2014.19. URL `doi.ieeecomputersociety.org/10.1109/ICGSE.2014.19`.

[4] A. Avritzer, S. Beecham, R. Britto, J. Kroll, D. S. Menasche, J. Noll, and M. Paasivaara. Extending survivability models for global software development with media synchronicity theory. In *2015 IEEE 10th International Conference on Global Software Engineering*, pages 23–32, July 2015. doi: 10.1109/ICGSE.2015.29.

[5] Ivana Bosnić, Igor Čavrak, Mario Žagar, Rikard Land, and Ivica Crnkovic. Customers' role in teaching distributed software development. In *Software Engineering Education and Training (CSEE&T), 2010 23rd IEEE Conference on*, pages 73–80. IEEE, 2010.

[6] Ivana Bosnić, Federico Ciccozzi, Igor Čavrak, Raffaela Mirandola, and Marin Orlić. Multi-dimensional assessment of risks in a distributed software development course. In *Collaborative Teaching of Globally Distributed Software Development (CTGDSD), 2013 3rd International Workshop on*, pages 6–10. IEEE, 2013.

[7] Ivana Bosnić, Federico Ciccozzi, Ivica Crnković, Igor Čavrak, Elisabetta Di Nitto, Raffaela Mirandola, and Mario Žagar. Managing diversity in distributed software development education – a longitudinal case study. *ACM Trans. Comput. Educ.*, 19(2):10:1–10:23, January 2019. ISSN 1946-6226.

[8] B. Bruegge, A. H. Dutoit, R. Kobylinski, and G. Teubner. Transatlantic project courses in a university environment. In *Proceedings Seventh Asia-Pacific Software Engeering Conference. APSEC 2000*, pages 30–37, 2000.

[9] Abraham Carmeli, Yair Friedman, and Asher Tishler. Cultivating a resilient top management team: The importance of relational connections and strategic decision comprehensiveness. *Safety Science*, 51(1):148 – 159, 2013. ISSN 0925-7535.

[10] Marcelo Cataldo and James D Herbsleb. Communication networks in geographically distributed software development. In *Proceedings of the 2008 ACM conference on Computer supported cooperative work*, pages 579–588. ACM, 2008.

[11] Igor Čavrak and Ivana Bosnić. Team resilience in distributed student projects. In *Proceedings of the 13th Conference on Global Software Engineering*, pages 112–120. ACM, 2018.

[12] Igor Čavrak and Rikard Land. Taking global software development from industry to university and back again. In *Proceedings of ICSE International Workshop on Global Software Development (GSD)*, 04 2003.

[13] Igor Čavrak, Marin Orlić, and Ivica Crnković. Collaboration patterns in distributed software development projects. In *Proceedings of the 34th International Conference on Software Engineering*, pages 1235–1244. IEEE Press, 2012.

[14] Tony Clear, Sarah Beecham, John Barr, Mats Daniels, Roger McDermott, Michael Oudshoorn, Airina Savickaite, and John Noll. 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*, pages 1–39. ACM, 2015.

[15] Diane L. Coutu. How resilience works. *Harvard Business Review*, 2002.

[16] Daniela Damian, Ban Al-Ani, Davor Cubranic, and Lizveth Robles. Teaching requirements engineering in global software development: A report on a three-university collaboration. In *Workshop on Requirements Engineering Education and Training*, pages 685–690, 2005.

[17] Daniela Damian, Luis Izquierdo, Janice Singer, and Irwin Kwan. Awareness in the wild: Why communication breakdowns occur. In *Global Software Engineering, 2007. ICGSE 2007. Second IEEE International Conference on*, pages 81–90. IEEE, 2007.

[18] Daniela Damian, Sabrina Marczak, and Irwin Kwan. Collaboration patterns and the impact of distance on awareness in requirements-centred social networks. In *Requirements Engineering Conference, 2007. RE'07. 15th IEEE International*, pages 59–68. IEEE, 2007.

[19] Phil Diegmann and Christoph Rosenkranz. Team diversity and performance – how agile practices and psychological safety interact. In *International Conference on Information Systems*, 12 2017.

[20] Olly Gotel, Christelle Scharff, and Sopheap Seng. Preparing computer science students for global software development. In *Proceedings - Frontiers in Education Conference*, pages 9 – 14, 12 2006. doi: 10.1109/FIE.2006.322632.

[21] Jo Hanisch and Brian Corbitt. Impediments to requirements engineering during global software development. *European Journal of Information Systems*, 16(6):793–805, 2007.

[22] Pamela Hinds and Cathleen McGrath. Structures that work: social structure, work structure and coordination ease in geographically distributed teams. In *Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, pages 343–352. ACM, 2006.

[23] Pamela J. Hinds and Mark Mortensen. Understanding conflict in geographically distributed teams: The moderating effects of shared identity, shared context, and spontaneous communication. *Organization Science*, 16(3): 290–307, 2005.

[24] John F. Home and John E. Orr. Assessing behaviors that create resilient organizations. *Employment Relations Today*, 24(4):29–39, 1998.

[25] Lorraine Johnston, Dirk Peters, Jean-Guy Schneider, and Ursula Wellen. Requirements analysis in distributed software engineering education – an experience report. In *Proceedings of 6th Australian Workshop on Requirements Engineering*, 2001.

[26] Antti Knutas, Jouni Ikonen, and Jari Porras. Communication patterns in collaborative software engineering courses: a case for computer-supported collaboration. In *Proceedings of the 13th Koli Calling International Conference on Computing Education Research*, pages 169–177. ACM, 2013.

[27] Seija Komi-Sirviö and Maarit Tihinen. Lessons learned by participants of distributed software development. *Knowledge and Process Management*, 12 (2):108–122, 2005.

[28] Craig Larman. *Agile and iterative development: a manager's guide.* Addison-Wesley Professional, 2004.

[29] Gwanhoo Lee, J Alberto Espinosa, and William H DeLone. Task environment complexity, global team dispersion, process capabilities, and coordination in software development. *IEEE Transactions on Software Engineering*, 39(12):1753–1771, 2013.

[30] Timothy C Lethbridge, Susan Elliott Sim, and Janice Singer. Studying software engineers: Data collection techniques for software field studies. *Empirical software engineering*, 10(3):311–341, 2005.

[31] Brian Lings, Björn Lundell, Pär J. Ågerfalk, and Brian Fitzgerald. Ten strategies for successful distributed development. In Brian Donnellan, Tor J. Larsen, Linda Levine, and Janice I. DeGross, editors, *The Transfer and Diffusion of Information Technology for Organizational Resilience*, pages 119–137, Boston, MA, 2006. Springer US. ISBN 978-0-387-34410-2.

[32] Alan MacCormack and Roberto Verganti. Managing the sources of uncertainty: Matching process and context in software development. *Journal of Product Innovation Management*, 20(3):217–232, 2003.

[33] Bonnie MacKellar. A case study of group communication patterns in a large project software engineering course. In *Software Engineering Education and Training (CSEE&T), 2012 IEEE 25th Conference on*, pages 134–138. IEEE, 2012.

[34] Larry Mallak. Putting organizational resilience to work. *Industrial Management (Norcross, Georgia)*, 40(6 NOV./DEC.):8–13, 1998. ISSN 0019-8471.

[35] Isabella Meneghel, Marisa Salanova, and Isabel M Martínez. Feeling good makes us stronger: How team resilience mediates the effect of positive emotions on team performance. *Journal of Happiness Studies*, 17(1):239–255, 2016.

[36] Sal Mistry, A C. Stoverink, and B Rosen. Team resilience: A theoretical model of teams that bounce back from adverse events. *Academy of Management Proceedings*, 2015:17642–17642, 01 2015. doi: 10.5465/AMBPP.2015.17642abstract.

[37] Thanh Nguyen, Timo Wolf, and Daniela Damian. Global software development and delay: Does distance still matter? In *Global Software Engineering, 2008. ICGSE 2008. IEEE International Conference on*, pages 45–54. IEEE, 2008.

[38] John Noll and Sarah Beecham. Measuring global distance: A survey of distance factors and interventions. In *Software Process Improvement and Capability Determination - 16th International Conference, SPICE 2016, Dublin, Ireland, June 9-10, 2016, Proceedings*, pages 227–240, 2016.

[39] Martin Nordio, H Christian Estler, Bertrand Meyer, Nazareno Aguirre, Rafael Prikladnicki, Elisabetta Di Nitto, and Anthony Savidis. An experiment on teaching coordination in a globally distributed software engineering class. In *Software Engineering Education and Training (CSEE&T), 2014 IEEE 27th Conference on*, pages 109–118. IEEE, 2014.

[40] Helena Olsson, Eoin Ó Conchúir, Pär Ågerfalk, and Brian Fitzgerald. Global software development challenges: A case study on temporal, geographical and socio-cultural distance. In *Proceedings - 2006 IEEE International Conference on Global Software Engineering*, pages 3–11, 2006.

[41] Maria Paasivaara, Casper Lassenius, Daniela Damian, Petteri Räty, and Adrian Schröter. Teaching students global software engineering skills using distributed scrum. In *Proceedings of the 2013 International Conference on Software Engineering*, pages 1128–1137. IEEE Press, 2013.

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

[43] Arnold Pears and Mats Daniels. Developing global teamwork skills: The runestone project. In *IEEE EDUCON, April 14-16, 2009, Madrid, SPAIN*. IEEE, 2010.

[44] A. Peters, W. Hussain, A. Cajander, T. Clear, and M. Daniels. Preparing the global software engineer. In *2015 IEEE 10th International Conference on Global Software Engineering*, pages 61–70, July 2015. doi: 10.1109/ICGSE.2015.20.

[45] Per Runeson, Martin Host, Austen Rainer, and Bjorn Regnell. *Case study research in software engineering: Guidelines and examples*. John Wiley & Sons, 2012.

[46] Shikha Sharma and Sanjeev Kumar Sharma. Team resilience: scale development and validation. *Vision*, 20(1):37–53, 2016.

[47] Günter K Stahl, Kristiina Mäkelä, Lena Zander, and Martha L Maznevski. A look at the bright side of multicultural team diversity. *Scandinavian Journal of Management*, 26(4):439–447, 2010.

[48] Didi Surian, David Lo, and Ee-Peng Lim. Mining collaboration patterns from a large developer network. In *Reverse Engineering (WCRE), 2010 17th Working Conference on*, pages 269–273. IEEE, 2010.

[49] Kathleen Sutcliffe and Timothy Vogus. Innovation and intellectual property rights. In J. E. Dutton K. S. Cameron and R. E. Quinn, editors, *Positive Organizational Scholarship: Foundations of a New Discipline*, pages 94–110. Berrett Koehler Publishers, San Francisco, CA, 2003.

[50] Adam Trendowicz and Jurgen Munch. Factors influencing software development productivity–state-of-the-art and industrial experiences. *Advances in Computers*, 77:185 – 241, 2009.

[51] June M Verner, Jennifer Sampson, Vladimir Tosic, NA Abu Bakar, and Barbara A Kitchenham. Guidelines for industrially-based multiple case studies in software engineering. In *Research Challenges in Information Science, 2009. RCIS 2009. Third International Conference on*, pages 313–324. IEEE, 2009.

[52] Claes Wohlin, Per Runeson, Martin Höst, Magnus C Ohlsson, Björn Regnell, and Anders Wesslén. *Experimentation in software engineering.* Springer Science & Business Media, 2012.

[53] Lu Xiao, Zhongyuan Yu, Bohong Chen, and Xiao Wang. How robust is your development team? *IEEE Software*, 35(1):64–71, 2018.