ARENBERG DOCTORAL SCHOOL
FACULTY OF ENGINEERING TECHNOLOGY

ETHER, GROUP T LEUVEN CAMPUS
ANDREAS VESALIUSSTRAAT 13 - BOX 2203
B-3000 LEUVEN, BELGIUM

EXPLORING THE INFLUENCE OF TEAM COHESION ON TEAMWORK IN SOFTWARE ENGINEERING EDUCATION: THE ASEST FRAMEWORK

Daymy Tamayo Avila

Dissertation presented
in partial fulfilment of
the requirements for
the degree of Doctor of
Engineering
Technology

Acknowledgment

I dedicate my doctorate to my family, and especially to my mother, for giving me the core values required for the success of a work like this, hoping it can inspire the youngest to conquer their dreams. Thank you for your teachings and love.

This work would not have been possible without the unconditional support of my supervisor, Professor Wim Van Petegem. To him my deepest gratitude. Thank you for believing in me and giving me the opportunity, be always there in the difficult moments, for so many hours spent reviewing my work, for all the discussions about science, and also for the philosophical ones. Words are not enough to express my appreciation.

My sincere gratitude also goes to my co-supervisor, Professor Monique Snoeck, for constantly inspiring me to bring my work one step further. Thank you for your ideas to enrich my research, and for all your efforts in revising my work. My sincere appreciation goes further to my co-supervisor, Professor Marcia Esther Noda Hernández, for all the time devoted to providing me with guidance and supporting my work in Cuba.

Special thanks to Professor Arno Libotton for your continued methodological advice and support along with this research. Thank you to all the members of the jury for your valuable feedback on my work, your review, and your suggestions.

I thank Professors Jenny Ruiz, Roxana Cañizares, David Leyva, Fernando Robles, Neysi Pupo, and Yarianny Torrente; for your contributions to the validation of the basis of this proposal and the instruments used. Thank you for sharing with me your valuable ideas and expertise.

Thank you to my colleagues in the Informatics department at the University of Holguin in Cuba. I appreciate very much the support of the head of the department Prof. Esmereldo Carbó, and the dean of the faculty of Informatics-Mathematics Prof. Miguel Escalona. I especially thank the assistant teachers in the experiments Ivet Challenger and Antonio López, my master and bachelor students, the management staff of the software company Datys, as well as the clients of the students' project and all the participants of the experiments.

I would also like to give special thanks to my friends in the Faculty of Engineering Technology Dean's office and to the supporting staff of my supervisor for providing me with a warm environment to work in Belgium. Thank you all for your help and kindness.

Finally, I express my immense gratitude to all my friends and to all that contributed somehow to this work.

Abstract

Ingenieursonderwijs is van cruciaal belang voor de ontwikkeling van de industrie en dus van de samenleving in het algemeen in elk land. Tegenwoordig wordt algemeen erkend dat om een effectieve software-ingenieur te zijn in de moderne software-industrie men vaardigheden nodig heeft om in een team te kunnen functioneren. Een groot aantal gepubliceerde artikelen bevestigen dat teams op grote schaal zijn onderzocht in het software engineering onderwijs (SEE). Beroemde Chaos-rapporten uit de industrie hebben echter aangetoond dat het succes van softwareprojecten rond 30% ligt. Meer dan twee decennia zijn voorbijgegaan sinds Demarco en Lister, in hun klassieke boek "Peopleware, Productive Projects and Teams", verklaarden dat wanneer softwareprojecten mislukken dit over het algemeen meer te wijten is aan problemen met teamwerk dan aan technische problemen. Hun uitspraak is nog steeds actueel in de discussie over de menselijke en sociale aspecten van software engineering. Zij toonden toen aan dat voor typische grote softwareprojecten teamwerk ongeveer 70% van de projecttijd in beslag neemt. Tegenwoordig zou dit zelfs nog hoger kunnen liggen door het algemene gebruik van agile methoden door de softwaregemeenschap, aangezien deze hun succes baseren op zelfsturende teams. Daarom moet meer aandacht worden besteed aan het overbruggen van de kloof tussen wat software-ingenieurs leren over teamwerk en wat nodig is om goed te presteren in teams in de moderne software-industrie.

Sommige auteurs hebben de prestaties van teams in SEE bestudeerd. Er zijn echter verschillende conceptualisaties gebruikt; en ze leggen allemaal de nadruk op de procesresultaten in plaats van op de gedragingen die software-ingenieurs in staat stellen teamdoelstellingen te bereiken. Daarom stelt dit proefschrift het volgende **wetenschappelijke probleem** aan de orde: Hoe kan het teamperformantiegedrag in software engineering onderwijs verbeterd worden? Cohesie wordt beschouwd als een van de belangrijkste factoren die software teams beïnvloeden. Het is belangrijk gebleken voor aspecten als teameffectiviteit, motivatie, productiviteit, teamsamenwerking, en agile praktijken. Cohesie blijkt ook sterk gecorreleerd te zijn met performantie. Cohesie is echter nauwelijks bestudeerd in SEE en er is geen empirisch bewijs over de relatie met teamperformantiegedrag.

Aangezien verschillende wetenschappers overlappende conceptualiseringen hebben voorgesteld, onderzoeken we in dit proefschrift zowel de resultaten of uitkomsten van teamprestaties (simpelweg aangeduid als 'teamprestaties') als het gedrag van teamprestaties in termen van teamleren. Het **algemene doel** is om een onderwijs-leerkader te bieden, genaamd Agile Software Engineers Stick Together (ASEST), dat

gericht is op het ontwikkelen van teamcohesie, wat leidt tot beter teamleren en teamprestaties van SE studententeams. Het proces dat gevolgd is om de doelstelling van dit onderzoek te bereiken omvatte twee iteraties. De eerste iteratie is gericht op het bepalen van de basis van het raamwerk, het opzetten van een voorlopige versie (ASEST0), en het testen ervan. Drie **onderzoeksvragen** worden beantwoord: RQ1: Welke huidige onderwijs- en leerbenaderingen kunnen worden gebruikt om het raamwerk op te zetten?; RQ2: Verbetert de toepassing van ASEST0 de ervaren teamcohesie, het teamleren, en de teamprestaties? En; RQ3: Wat zijn de percepties van de studenten over ASEST0? Iteratie 2 was gericht op het verbeteren van het voorlopige voorstel en het valideren van het uiteindelijke raamwerk (ASEST+). De volgende **onderzoeksvragen** worden beantwoord: RQ4: Wat zijn antecedenten voor de cohesie van teams op één locatie in SEE?; RQ5: Wat zijn de meest relevante geïdentificeerde antecedenten voor agile studententeams op één locatie?; RQ6: Wat zijn de benaderingen in Agile Software Development (ASD) met betrekking tot de relevante antecedenten om het raamwerk te verbeteren? RQ7: Welke benaderingen met betrekking tot agile teamwerk in SEE kunnen worden gebruikt om de leerstrategieën van het raamwerk te verbeteren? RQ8: Verbetert de toepassing van de ASEST+ de waargenomen teamcohesie, het teamleren en de teamprestaties? RQ9: Speelt teamcohesie een mediërende rol door de toepassing van de ASEST+? En, RQ10: Wat zijn de percepties van docenten op ASEST+?

ASEST0 combineert team-based learning, project-problem-based learning, en role-playing game leerstrategieën in drie fasen en acht stappen. De eerste fase heeft tot doel de leeromgeving tot stand te brengen en de leerlingen voor te bereiden op vaardigheden in teamwerk. De kern van het raamwerk is de tweede fase met het opstellen van een overeenkomst over teamregels die communicatie en conflicthantering ondersteunen. De derde fase richt zich op het bijstellen van de overeenkomst via een zelf- en peer-evaluatie van de bijdragen van de teamleden. Een quasi-experiment met een groep studenten die ASEST0 toepasten, gaf aan dat hun positieve percepties van teamcohesie, teamleren en teamprestaties significant toenamen in vergelijking met de percepties van de studenten in een controlegroep. Een quasi-experiment van ASEST0 met teams die presteerden in een bedrijfsomgeving toonde eveneens aan dat ASEST0 effectief was. Uit een enquête bleek dat de studenten ons voorstel aanvaardden.

ASEST+ wil de geïdentificeerde moeilijkheden uit iteratie 1 oplossen en rekening houden met antecedenten voor cohesie. Het is opgebouwd rond Scrum-teams en combineert leerstrategieën om studenten te trainen in collaboratieve en technische agile praktijken. ASEST+ stelt beleidslijnen op voor rolverdeling en teamafspraken om communicatie te

reguleren en conflicthantering gekoppeld aan agile praktijken aan te pakken. ASEST+ richt zich op persoonlijkheid, conflicten en taakafhankelijkheid omdat deze antecedenten als de belangrijkste zijn geïdentificeerd. Een quasi-experiment toonde aan dat het gebruik van ASEST+ de positieve percepties van de studenten over teamcohesie, teamprestaties en teamleren significant verhoogt in vergelijking met een controlegroep. Een ander quasi-experiment met één groep studenten zonder deelname van de onderzoeker repliceerde de interventie. Ook hieruit bleek dat ASEST+ effectief was. Bovendien bevestigde een longitudinale analyse in deze studie dat cohesie een mediator is tussen de antecedenten persoonlijkheid, conflicten, en taakafhankelijkheid en de uitkomsten teamleren en teamprestatie.

De belangrijkste bijdragen van dit proefschrift zijn de identificatie van trends op het gebied van teamwerk in SE, de identificatie van de relevante aspecten die de cohesie van software engineering studententeams beïnvloeden, de onderbouwing van het ASEST raamwerk om cohesie voor software engineering studententeams te ontwikkelen, en de experimentele validatie met het ASEST raamwerk. Hoewel enkele beperkingen in termen van generalisatie in acht moeten worden genomen, is het onze verwachting dat de bevindingen in dit proefschrift kunnen bijdragen tot het afleveren van beter voorbereide software-ingenieurs aan de industrie en nieuwe deuren kunnen openen voor toekomstig gerelateerd onderzoek.

Abstract

Engineering education is critical for the development of industry and hence of society in general in any country. Nowadays it is generally recognized that in order to be an effective software engineer in the modern software industry one requires skills for performing in a team. A large number of published papers confirm that teams have been widely researched in software engineering education (SEE). However, famous industry Chaos Reports have shown that software project success is around 30%. More than two decades have passed since Demarco and Lister, in their classic book "Peopleware, Productive Projects and Teams", stated that when software projects fail it is generally more because of teamwork problems than technical issues. Their statement remains a current concern for progress discussion on human and social aspects of software engineering. They showed then that for typical large software projects team working is about 70% of the project time. Nowadays this could be even higher because of the general use of agile methods by the software community as they base their success on self-managed teams. Accordingly, more attention should be paid to bridging the gap between what software engineers are learning about teamwork and what is required to properly perform in teams in the modern software industry.

Some authors have studied team performance in SEE. However, diverse conceptualizations have been used; and they all put emphasis on the process results rather than on the behaviors that allow software engineers to obtain team objectives. Thus, this dissertation addresses the following **scientific problem**: How to improve team performance behaviors in software engineering education? Cohesion is considered one of the most important factors that influence software teams. It has been found important for aspects like team effectiveness, motivation, productivity, team collaboration, and agile practices. Cohesion has also been found strongly correlated with performance. However, cohesion has been scarcely studied in SEE and there is no empirical evidence on the relationship with team performance behaviors.

Since overlapping conceptualizations have been proposed by different scholars, in this research we examine both team performance outcomes (referred to as 'team performance') and team performance behaviors in terms of team learning. The **overall aim** is to provide a teaching-learning framework named Agile Software Engineers Stick Together (ASEST) that aims to develop team cohesion, leading to better team learning and team performance of SE student teams. The process followed to address the objective of this research included two iterations. Iteration 1 aimed at determining the basis of the framework, setting up a preliminary version (ASEST 0), and testing it. Three **research questions** are answered: RQ1: What current teaching and learning approaches can be used to set up the framework?; RQ2: Does the application of the ASEST0 improve perceived team cohesion, team learning, and team performance? And; RQ3: What are the students' perceptions of ASEST0? Iteration **2** aimed at improving the preliminary proposal and validating the final framework (ASEST+). The following **research questions** are answered: RQ4: What are the cohesion antecedents for collocated teams in SEE?; RQ5: What are the most relevant identified antecedents for agile collocated student teams?; RQ6: What are the approaches in Agile Software Development (ASD) regarding the relevant antecedents to improve the framework?; RQ7: What approaches regarding agile teamwork in SEE can be used to improve the framework learning strategies?; RQ8: Does the application of the ASEST+ improve perceived team cohesion, team learning, and team performance?; RQ9: Does team cohesion have a mediational role through the application of ASEST+? And, RQ10: What are the perceptions of teachers on ASEST+?

The ASEST0 combines team-based learning, project-problem-based learning, and role-playing game learning strategies in three phases and eight steps. The first phase aims to establish the learning environment and prepare the students on team working skills. The core of the framework is the second phase with an establishment of an agreement on team rules that support communication and conflict management. The third phase focuses on adjusting the agreement via a self and peer evaluation of member contributions. A quasi-experiment of one group of students that applied the ASEST0 indicated that their positive perceptions of team cohesion, team learning, and team performance increased significantly compared with the perceptions of the students in a control group. A quasi-experiment of ASEST0 involving teams performing in a company setting also showed ASEST0 to be effective. A survey showed students' acceptance of our proposal.

ASEST+ aims to solve the identified difficulties from iteration 1 and considers cohesion antecedents. It is built around Scrum teams and combines learning strategies to train students in collaborative and technical agile practices. ASEST+ establishes policies for role allocation and team rule agreements to regulate communication and address conflict management linked to agile practices. ASEST+ addresses personality, conflicts, and task interdependence as these antecedents are identified as the most important. A quasi-experiment showed that the use of ASEST+ significantly increases the students' positive perceptions on team cohesion, team performance, and team learning compared with a control group. Another quasi-experiment over one group of students without the researcher’s participation replicated the intervention. It also showed ASEST+ to be effective. In addition, a mediation longitudinal analysis along this study confirmed cohesion to be a mediator between the antecedents personality, conflicts, and task interdependence and the outcomes team learning and team performance.

The main contributions of this dissertation are the identification of trends on teamwork in SEE, the identification of the relevant aspects affecting cohesion of software engineering student teams, the foundation of ASEST framework to develop cohesion for software engineering student teams, and experimental validation with ASEST framework. Although some limitations in terms of generalization should be considered, it is our expectation that the findings presented in this dissertation can contribute to delivering better-prepared software engineers to the industry and open new doors to future related research.

Contents

Acronyms and abbreviations

ABET	Accreditation Board for Engineering and Technology
ACM	Association for Computing Machinery
ASD	Agile Software Development
BSE	Behavioral Software Engineering
ASEST	Agile Software Engineers Stick Together
CATME-B	Comprehensive Assessment of Team Member Effectiveness (peer and self-evaluation) - Behavioral Anchored
IEEE	Institute of Electrical and Electronics Engineers
IMO	Input-Mediator-Outcome
IMOI	Input-Mediator-Outcome-Input
SEE	Software Engineering Education
TKI	Team Knowledge Inventory
TPC	Team Process Checks

List of tables

List of figures

Chapter 1
General introduction

1.1 Introduction

The engineering profession has evolved over the past decades driven by changes that resulted from events such as the significant advances in technology and the impact of globalization (Rajala, 2013). (K. A. Smith, 2012) have identified five major shifts reshaping engineering education in 100 years in pursuing to face such challenges. One of these shifts is "the application of research in education, learning, and social-behavioral sciences to curricula design and teaching methods". This is particularly important for software engineering education (SEE), which has been recognized under constant pressure to provide students with relevant knowledge and skills for industry (Fagerholm et al., 2017).

Software engineering is a complex socio-technical (Steve Sawyer, 2004) and knowledge-intensive activity that relies on human collaboration. IEEE defines software engineering as "the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software" (IEEE, 1990). While discussing how to prepare the engineer for the future, (Rajala, 2013) states that the second top attribute of this engineer is to be proficient in working in or directing a team. For SEE it is now widely recognized that there is a need to teach students how to work effectively in teams. Software engineering needs engineers able to play different roles during several tasks, coordinating efforts to achieve coherence in different phases of the process.

Several researchers have highlighted the importance of teamwork in software development since the early stages of this industry (Demirors et al., 1997). More recently (Congalton, 2014) stated that "an overriding trend in software development is that software is increasingly created by teams and not individuals". According to (Dingsoyr et al., 2016) teams remain the core organizational form in software development with particular relevance to apply agile methods. Agile software development (ASD) is nowadays a widely applied approach as a response to the need of modern software projects that require not only a high level of quality but also overcoming schedule and budget constraints in a rapidly growing and changing environment. ASD bases its success on self-managed teams able to prioritize and quickly respond to changes and to satisfy customers through early and continuous delivery of quality software.

Teamwork has been included in the ABET students' learning outcomes criteria for accrediting computer programs (ABET, 2012) and in the Curriculum Guidelines for Undergraduate Degree Programs in Computer Science from ACM and IEEE (IEEE; ACM, 2014). A large number of published studies also reflects growing attention to this subject. Industry reports, however, have shown that less than 30% of software projects are successful (Standish Group, 2015). The reason why many software projects fail is due more often to teamwork problems than technical issues, as authoritative authors have suggested (T. DeMarco, 1999). This remains a current concern for progress discussion indicated by (Lenberg et al., 2015). Accordingly, more attention should be paid to bridging the gap between what software engineering students are learning about teamwork and what is required to properly perform in teams in the modern software industry.

Team performance has been one of the related topics addressed in SEE (Lago et al., 2012) (Garcia & Pacheco, 2014). It is usually measured using indicators for effectiveness and efficiency. (Mathieu et al., 2008) and (Beal et al., 2003) however differentiated between performance behaviors and performance outcomes. According to these authors, performance behaviors are actions relevant for achieving goals, whereas performance outcomes are the consequences or results of performance behaviors. Nevertheless, these studies put more emphasis on the results rather than on the behaviors.

Team cohesion is relevant for software teams (Lenberg et al., 2015). Through a systematic literature review, these authors define a research area concerned with cognitive, social, and human aspects of software engineering named Behavioral Software Engineering (BSE), proposing a common platform for future research. At the team level, they found cohesion is one of the most addressed aspects in software engineering.

Through a meta-analysis of studies over a wide range of sectors for more than 50 years (in the period between 1951 and 2002), (Beal et al., 2003) identified cohesion as being strongly correlated with performance. Several authors have also addressed cohesion in SEE (e.g. (Chidambaram & Carte, 2005) (Chung-yang Chen et al., 2014)). Previous studies however did not address the relationship between cohesion and performance behaviors. This doctoral dissertation focuses on rectifying this shortcoming.

In this research, we examine both team performance outcomes (referred to as 'team performance') and team performance behaviors in terms of team learning. The **overall aim** of this doctoral research is to provide a teaching-learning framework called ASEST (Agile Software Engineers Stick Together) for the development of team cohesion aimed at improving team performance and team learning of software engineering student teams. This research focuses on teams that share the same workspace, also referred to as ‘collocated teams’. The following **research questions** are addressed:

1. What current teaching and learning approaches can be used to set up a framework for improving team cohesion leading to better team performance and team learning as perceived by software engineering student teams?
2. Does the application of the proposed preliminary framework improve team cohesion, team learning, and team performance as perceived by software engineering student teams?
3. What are the students’ perceptions of the proposed preliminary framework?
4. What are the cohesion antecedents for collocated teams in Software Engineering Education (SEE)?
5. What are the most relevant identified antecedents for agile collocated student teams?
6. What are approaches in Agile Software Development (ASD) regarding the relevant antecedents to improve the proposed framework?
7. What approaches regarding agile teamwork in Software Engineering Education (SEE) can be used to improve the framework learning strategies?
8. Does the application of the final framework improve team cohesion, team learning, and team performance as perceived by software engineering student teams?
9. Does team cohesion have a mediational role through the application of the final framework?
10. What are the perceptions of teachers on the final framework?

The **methodology** followed in this research is presented in Table 1. The table summarizes the research questions, design, and techniques used in this work. Certain conditions of the context where the experiments took place drove the decision to choose for a quasi-experiment design.

Table 1. Research questions, methodology, research design, and research techniques for the studies

Chapter	Research Questions	Methodology	Research Design	Research techniques
1	General introduction (research motivation, concepts, purpose of the research, research design, and overview of the dissertation)			
2	RQ1	Qualitative research	Literature review 1	Experts consult Descriptive (narrative) synthesis Cluster analysis
3	RQ 2 and RQ 3	Quantitative and qualitative research	Quasi-experiments 1 and 2	T student test Wilcoxon test Survey Content analysis
4	RQ 4- RQ6	Quantitative research	Literature reviews 2 and 3 Correlational study	Descriptive (narrative) synthesis Survey Pearson correlation analysis Analysis of variance Regression analysis
5	RQ 7- RQ 10	Qualitative and quantitative research	Literature review 1 (expanded) Quasi-experiment 3 and 4	Descriptive (narrative) synthesis T student test Wilcoxon test Mediational analysis Survey
6	General conclusions and contributions (overview and discussion of main results, implications and recommendations for future research)			

The research process (shown in Fig 1), was restricted by the fact that the researcher had to switch periods of working in Cuba and Belgium. While she had to teach courses in Cuba she also must attend courses and comply with other requirements of her doctoral school in Belgium.

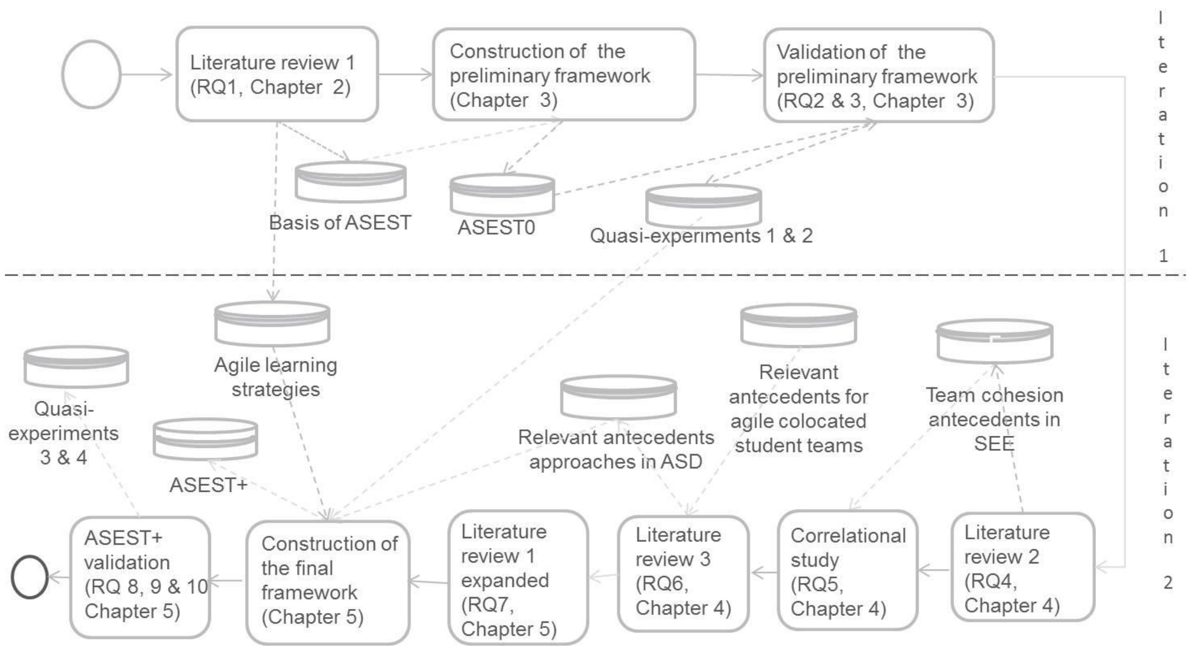

Literature review 1 (RQ1, Chapter 2)
Construction of the preliminary framework (Chapter 3)
Validation of the preliminary framework (RQ2 & 3, Chapter 3)
Iteration 1
Basis of ASEST
ASEST0
Quasi-experiments 1 & 2
Quasi-experiments 3 & 4
Agile learning strategies
ASEST+
Relevant antecedents approaches in ASD
Relevant antecedents for agile colocated student teams
Team cohesion antecedents in SEE
Iteration 2
ASEST+ validation (RQ 8, 9 & 10 Chapter 5)
Construction of the final framework (Chapter 5)
Literature review 1 expanded (RQ7, Chapter 5)
Literature review 3 (RQ6, Chapter 4)
Correlational study (RQ5, Chapter 4)
Literature review 2 (RQ4, Chapter 4)

Figure 1. Doctoral research process

1.2 Teamwork teaching-learning frameworks and team cohesion in engineering education

Literature on teamwork in engineering education is quite extensive. Among the aspects that have been studied are team member effectiveness assessment (Loughry et al., 2007), teamwork skill assessment (Loughry et al., 2014), team building (Sullivan et al., 2002), team working curriculum (Zemke, 2007), multidisciplinary teams (Hwang & Blandford, 2000) and team management (Mead et al., 2000). A variety of proposals for learning-teaching teamwork can be found. They include the use of communities of practice (Gates & Villa, 2014), the use of videos (Arvold & Goode, 2015), virtual environments (Sancho et al., 2011), and learning from industry experiences (Bowen et al., 2005) to mention some examples.

Intending to get an overall understanding regarding related teaching frameworks, we consulted Scopus and the Web of Sciences databases. We sought articles where the authors name their proposals as “pedagogical framework”, “learning framework”, “educational framework”, “teaching framework” or “teaching-learning framework”. We looked at these keywords together with “engineering education” and team* to be contained in the abstract, list of keywords, or title. We retrieved 58 papers.

The teaching frameworks mentioned in these papers addressed teamwork by using team-based learning combined with other approaches. Problem/project-based learning was the most often used approach (17 studies), followed by the use of ICT to enhance learning (6 studies), real-world problem solving, or any other link with industry (6 studies) and multidisciplinary teams (5 studies). Appendix 1 exemplifies the main contributions of these proposals. The aim is not to be extensive but to provide a broad view of the context.

A search over the Scopus and Web of Knowledge databases using the string ("engineering education" and cohesion and team*) showed that team cohesion has been previously addressed in engineering education, especially in engineering design. For example, while evaluating a communication framework for team effectiveness in a design and communication course, (Hoffart, et al. 2015) found that their proposal increased team cohesion.

The work of (Asio, Cross and Ekwaro-Osire 2018), found that cohesion was related to perceived student team innovation in this area. (Siggard, et al. 2014) approach industrial design with a curricular proposal that emphasizes team cohesion among the aspects they address to solve problems regarding gender. In another study addressing gender composition in engineering design, (Okudan, et al. 2002) found that cohesion values for all the female teams were considerably lower than those of other team types they studied. In two studies collective efficacy was found related to team cohesion in engineering design student project teams (R. W. Lent, J. A. Schmidt, et al. 2004) (R. W. Lent, L. Schmidt, et al. 2002).

Team cohesion has been also addressed in education enhanced by technology for engineering students. (Gupta 2018) found that perceived anxiety mediates the impact of cohesion on computer self-efficacy in self-paced technology training. In industrial management education, when studying the impact of introducing collaborative writing tools in a course, the authors found the use of these tools enhanced team cohesion among other aspects of teamwork (Uppvall, Blomkvist and Bergqvist 2017). Other authors studied the challenges of designing and implementing group-based learning activities to teach collaboration skills in engineering management education (Saunders, Jasper and Whitton 2010). They addressed the question of how academic staff takes on the logistical challenge of delivering effective group work to large cohorts of students. In this study they found the use of wikis improved team cohesion.

In SEE, in particular, previous studies on team cohesion include the work of (Wellington et al., 2005a), who found that cohesion was higher for the student teams that used agile methodologies compared with those that used plan-driven methodologies. A study of over

35 software engineering student teams, (Acuña et al., 2009, 2008) found the more the level of cohesion drops, the more conflict there was between the team members.

The study of (Rutz & Tanner, 2016) on group diversity over 44 teams of undergraduate students found that diversity was significant for the cohesion of virtual teams. (Chung-yang Chen et al., 2014) propose the Meeting-flow approach which emphasizes the role of meetings in conducting software capstone projects for undergraduate students. They evaluated their approach concerning teamwork quality and measured it in six facets: communication, coordination, balanced member contributions, mutual support, effort, and cohesion. The authors found that while their proposal significantly enhanced the other facets, it had less influence on team cohesion. (Silvestre et al., 2016) studied cohesion regarding the formation of software development teams. Their approach proposes a heuristic to form cohesive teams. However, they just focused on team design, leaving open the challenge of how to maintain the initial level of cohesion or how to raise it.

To summarise, educators around the world recognize the importance of teamwork for engineering students and mix different approaches to enhance team-based learning in their courses. Team cohesion has been found to influence the teamwork of engineering student teams. However, to date, there is a lack of teaching-learning frameworks focusing on the development of team cohesion. In SEE, in particular, only the proposal of (Silvestre et al., 2016) partially addresses this issue. Therefore, more efforts are needed to further develop the team cohesion of software engineering student teams.

1.3 Concepts

As stated by (E. Salas, 2005), as one begins to examine the team literature, it becomes clear that types of teams are as varied as the number of authors who have discussed them. However, conceptually, researchers have converged describing teams as complex, adaptive, dynamic systems (Ilgen et al., 2005).

In this doctoral dissertation, the definition of Hackman is assumed. This author states that a team can be defined as “A group *[which]* is an intact social system, complete with boundaries, interdependence for some shared purpose, and differentiated member roles” (Hackman & Katz, 2010). To limit the focus of this dissertation, team purpose and roles are related to software engineering activities in collocated educational settings. In the next sections, the conceptualization of team cohesion, team performance, and team learning are examined.

1.3.1 Team cohesion

Team cohesion is an emergent state that has been thoroughly investigated and diversely conceptualized (Beal et al., 2003; Chiocchio & Essiembre, 2009; Eduardo Salas et al., 2015). The findings of several meta-analyses on team cohesion tend to support the multidimensional view of team cohesion (Kozlowski & Bell, 2012), although some authors define cohesion as a unidimensional construct (Eduardo Salas et al., 2015).

Table 2 shows several definitions of team cohesion. This table is not meant to be exhaustive but it illustrates with a variety of team cohesion definitions that they are all related to these dimensions 1. Social cohesion: integration, closeness, and unification. 2. Task cohesion: tasks commitment. 3. Feelings: act as forces that bind people together and keep the individual in the team (e.g. group pride).

Table 2 Team cohesion definitions

Authors	Definition
(Festinger, 1950)	An attraction or bonding between group members that is based on a shared commitment to achieving the group's goals and objectives
(Seashore, 1954)	A closeness and attraction within the group that is based on social relationships within the group
(Cartwright & Zander, 1960)	Individuals' high degree of loyalty to fellow group members and their willingness to endure frustration for the group
(Cartwright, 1968)	The degree to which team members desire to remain on the team
(Lieberman et al., 1973)	A group property with individual manifestations of feelings of belongingness or attraction to the group
(Shaw, 1981)	The degree to which members of a group are attracted to each other
(A. Carron, 1982)	The extent to which the members of a group stick together and remain united in the pursuit of goals and objectives
(A. V. Carron et al., 1985)	A dynamic process that is reflected in the tendency for a group to stick together and remain united in the pursuit of its goals and objectives
(Wolfe & Box, 1987)	The degree to which team members hold an attraction for each other and a desire to remain intact as a team
(Bollen & Hoyle, 1990)	An individual's sense of belonging to a particular group and his or her feelings of morale associated with membership in the group
(Mullen & Copper, 1994)	(1) interpersonal attraction of team members, (2) commitment to the team task, and (3) group pride-team spirit
(Maznevski et al., 2000)	The extent to which team members enjoy working together and would like to continue to work together
(Okudan et al., 2002)	The average of member contribution levels
Schermerhorn, Hunt, and Osborn	The degree to which members are attracted to a group

(2002)	
(Beal et al., 2003)	The extent to which group members exhibit liking for the status or the ideologies that the group supports or represents, or the shared importance of being a member of the group
(Wellington et al., 2005a)	The degree to which the team sticks together as they pursue the team's purpose
(Wellen & Neale, 2006)	The overall attraction or bond amongst members of a group' and has 'two underlying components: (a) social cohesion, which describes the attraction amongst group members based on social relations within the group; and (b) task cohesion, viewed as the attraction that is based on a shared commitment to achieving group goals
(Karn et al., 2007)	The degree to which team members have close friendships with others in their immediate work unit and their personal attraction to members of the group
(Tesluk et al., 2009)	The tendency for a team to remain united in the pursuit of its objectives
(Curşeu & Pluut, 2013)	An emergent state that reflects the extent to which group members stick together (the group is a tight unit)
(Ralph & Shportun, 2013)	The extent to which a team acts together as a single agent toward shared goals
(Voulgari & Komis, 2015)	The forces that keep the group together and the links between group and members
(Lindsjørn et al., 2016)	Team members' motivation to maintain the team and accept that team goals are more important than individual goals

The meta-analyses on team cohesion have also shown different levels of importance for each dimension to team performance. The meta-analysis of (Beal et al., 2003) examined three dimensions of cohesion: interpersonal (social cohesion), task cohesion, and feelings of group pride. It showed that they were each equally significantly related to team performance. However, (Zaccaro & Lowe, 1988) and (Mullen & Cooper, 1994) concluded that task cohesion is the critical element when the cohesion-performance relationship is examined.

More recently, the meta-analysis of (Eduardo Salas et al., 2015) found that when cohesion is conceptualized using social and task (but not other) dimensions and when analyses are performed at a team level, its relationship with performance is more significant. Therefore, they recommend researchers to prioritize social and tasking dimensions.

Following this recommendation, the criteria of (Wellington et al., 2005a) are assumed in this dissertation. They specify that within software engineering in particular, cohesion can be seen in two ways: the social attachment within the team and the team's connection to the project itself, as defined by (A. V. Carron et al., 1985). These last authors represent cohesion at individual and group level in four constructs, adopted in this dissertation:

1) Group Integration - Task cohesion, which represents an individual’s perception about his or her group members’ tendencies to remain in the group because of the task;

2) Group Integration – Social cohesion, which describes an individual’s perception about the group members’ tendencies to remain in the group because of the social interaction;

3) Individual Attraction to the Group - Social cohesion, which means the individual’s intention to stay in the group because of the social interaction; and

4) Individual Attraction to the Group - Task cohesion, which represents the individual’s intention to stay in the group because of the task.

1.3.2 Team performance behaviors and outcomes

Team performance has been differentiated in behaviors and outcomes (Campbell, 1990) (Beal et al., 2003). Team performance outcomes are focused on results while team performance behaviors are related to the actions that drive obtaining those results. While differentiating team performance outcomes and behaviors, these authors pointed that the performance outcomes perspective does not consider potential inhibitors for the team to perform (Campbell, 1990) (Beal et al., 2003).

Team performance outcomes is the most common treated perspective in many research fields, including the literature on the cohesion-performance relationship (Beal et al., 2003). From this view, team performance has been seen “in the most restricted sense, [...] in terms of whether or not a team achieves the tasks set for it” (Senior & Swailes, 2019), or as “evaluations of the results of the teamwork” (T. Dingsøyr & Lindsjørn, 2013).

In software development, team performance outcomes have been characterized as a multidimensional construct. For instance, (S. Sawyer & Guinan, 1998) include team efficiency, product quality and team effectiveness attributes. The majority of researchers only refer to team efficiency and effectiveness dimensions (T. Dingsøyr & Lindsjørn, 2013). Efficiency is related to “the adherence to schedules and budgets” (Hoegl et al., 2003). Some authors refer to team effectiveness and quality indistinctly. They see the effectiveness as “the degree to which teams meet expectations regarding the quality of the outcome” (Hoegl et al., 2003). In addition, some authors, however, have studied other team-level outcomes like customer satisfaction and innovation (Mathieu et al., 2008).

Team performance behaviors have been studied regarding team process improvement, team learning, and cognitive task performance (Mathieu et al., 2008). Authors have proposed several team performance behaviors to be studied according to specific types of teams and contexts. However, referring to knowledge-based activity teams, researchers have included overlapping behaviors while addressing these constructs. For example, (Kirkman et al., 2004) study team process improvements by measuring seeking feedback, error discussion, and experimentation. These behaviors are addressed by (A. Edmondson, 1999) while studying team learning, although her conceptualization also includes behaviors related to exploring and reflecting. Related behaviors are analyzed by (Tindale & Vollrath, 1997) while addressing cognitive task performance concerning actions the teams exhibit in combining, integrating, and processing information.

Team learning has been recognized as important for teams to develop performance capabilities, to adapt to changes, to renew and sustain their performance over time (Bell et al., 2012), and to be able to successfully adapt and improve knowledge (Mathieu et al., 2008). In software development, in particular, team learning (as conceptualized by (A. Edmondson, 1999)) was found to influence the performance of agile teams (Torgeir Dingsøyr et al., 2016).Team learning emerged as a topic in 1990s in the management field and gained more attention from researchers in the 2000s. However, team learning conceptualizations vary considerably even within research areas (A. C. Edmondson et al., 2007). Table 3 illustrates some definitions of team learning, showing the “generative and occasionally confusing” (A. C. Edmondson et al., 2007) proliferation of this construct.

Table 3. Team learning definitions

Authors	Definition
(Dechant et al., 1993)	Processes that revolve around collective thinking and action
(Brooks, 1994)	The construction of collective new knowledge by a team
(Kasl et al., 1997)	A process through which a group creates knowledge for its members, for itself as a system, and for others
(A. Edmondson, 1999)	Activities carried out by team members through which a team obtains and processes data that allow it to adapt and improve
(Gruenfeld et al., 2000)	The acquisition, persistence, diffusion, and depreciation of group knowledge
(Argote et al., 2001)	Activities by which team members seek to acquire, share, refine, or combine task-relevant knowledge through interaction with one another
(Sole & Edmondson, 2002)	The acquisition and application of knowledge that enables a team to address team tasks and issues for which solutions were not previously obvious

(Marquardt, 2002)	The increase in knowledge, skills, and competencies accomplished by and within groups
(A. P. J. Ellis et al., 2003)	A relatively permanent change in the team's collective level of knowledge and skill produced by the shared experience of the team members
(G. S. van der Vegt & Bunderson, 2005)	Activities by which team members seek to acquire, share, refine, or combine task-relevant knowledge through interaction with one another
(Van den Bossche et al., 2006)	The creation of mutually shared cognition
(J. M. Wilson et al., 2007)	A change in the group's repertoire of potential behavior
(Mathieu et al., 2008)	An ongoing process of reflection and action, through which teams acquire, share, combine, and apply knowledge
(Van Woerkom & Croon, 2009)	The learning activities carried out by team members through which a team obtains and processes data that allow it to adapt and improve and through which outcomes such as better team performance can be achieved
(Decuyper et al., 2010)	A compilation of team-level processes that circularly generate change or improvement for teams, team members, organizations, etc
(Fisser & Browaeys, 2010)	The process of sharing individual mental models by which through collective sense making a shared mental model is created and evolved
(Sessa et al., 2011)	The process of deepening and broadening of the teams' capabilities [...] in: (re) structuring to meet changing conditions; adding and using new skills, knowledge, and behaviors; and becoming an increasingly sophisticated system
(Bresman & Zellmer-Bruhn, 2012)	Activities through which a team obtains and processes knowledge allowing it to improve
(Raes et al., 2015)	Team learning is an active, reflexive, and boundary crossing process of balancing 'co- construction' and 'constructive conflict' between team members and between team members and team external stakeholders
(Wiese & Burke, 2019)	A shift in a team's collective knowledge state

In two outstanding reviews on team learning research (A. C. Edmondson et al., 2007) (Bell et al., 2012) the authors have also differentiated between researches that address this construct as outcomes and others that focus on team learning behaviors. In their meta-analysis, Edmondson and her colleagues argue that researchers have used improvements in team performance as evidence that team learning has occurred. In this case, they conceptualize team learning as learning curves in operational settings and team member coordination of task knowledge. A second stream they found sees team learning as a process where researchers have used diverse criteria to study behaviors related to it.

Similarly, five years later in another meta-analysis of two decades of literature addressing team learning research, Bell and his colleagues identified a stream of studies that address several team learning behaviors (Bell et al., 2012). A second stream focuses on team knowledge outcomes that emerge from learning in the form of collective knowledge, transactive memory, team mental models, macrocognition, and team knowledge emergence. They argue that one problem of aligning team learning with team performance outcomes is that sometimes teams can learn yet not experience changes in their performance. Some authors have found that despite team learning has improved there were no significant changes (e.g. (Dayaram & Fung, 2012) or even the teams showed a performance decrease (Druskat & Kayes, 2000).

Considering this divergence of findings, in this dissertation, both team performance outcomes (henceforth referred to as 'team performance') and team performance behaviors (addressed as team learning) are studied. In particular, the conceptualization of (A. Edmondson, 1999) is assumed. This author defines team performance as the 'degree in which the team satisfies client needs and expectations'. Team learning is related to "activities carried out by team members through which a team obtains and processes data that allow it to adapt and improve".

In a recent review on team learning terminology (Wiese & Burke, 2019), the authors observed a mainstream of research focusing on behaviors related to the "internal processes teams engage in that build shared meaning from existing information, identify and fill in gaps in the team's collective knowledge, as well as challenge, test, and explore assumptions". The present dissertation adheres to this stream of research. Specifically, the conceptualization of (A. Edmondson, 1999) is used; it includes the following team learning behaviors: 1. exploring, 2. reflecting, 3. discussing errors and unexpected outcomes of actions, 4. seeking feedback, and 5. experimenting within and as a team.

1.4 Dissertation overview

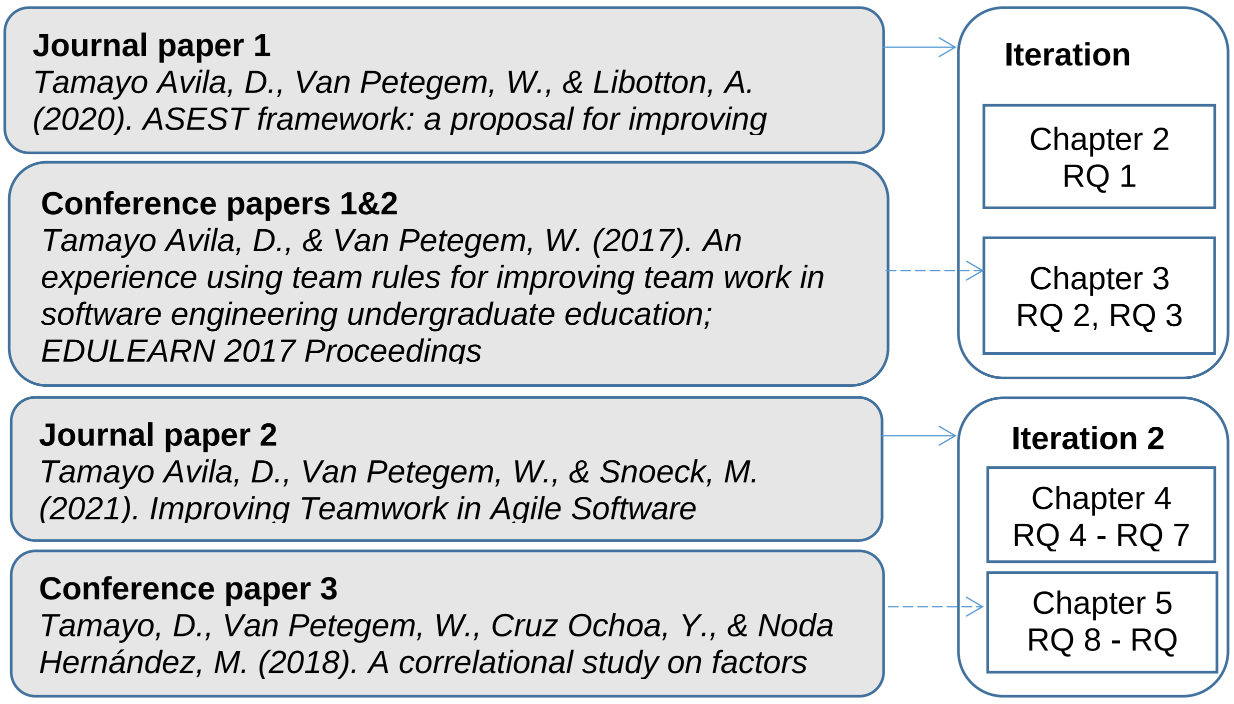
This dissertation is divided in two parts, according to the process followed to address the objective of this research (showed in Fig. 1), which included two iterations. Chapter 2 and Chapter 3 refer to iteration1. Iteration 1 aimed at determining the basis of the framework, to set up a preliminary version of this proposal (ASEST 0) and to test it.

Chapter 2 reports on the identification of the learning strategies to be included in the framework through a study over the trends on teamwork in SEE. Chapter 3 describes the construction of the preliminary framework ASEST0 guided by the IMO model (Input-

Mediator-Outcome) (Ilgen et al., 2005). Besides, it reports on two studies performed in order to test its effectiveness.

Chapters 4 and 5 focus on the second iteration. This iteration aimed at improving the proposal and validating the final framework (ASEST+). The identification of improvements for ASEST+ proceeded in two directions. The first stream of improvements aims to solve the difficulties identified from the quasi-experiments 1 and 2 and make ASEST+ more suitable for agile practice education. In doing so, ASEST0's learning strategies were further evaluated via a literature review.

The IMO model guides the second stream with the aim of identifying antecedents not yet considered in ASEST0. Two literature reviews and a correlational study were conducted. ASEST+ was then examined by means of two quasi-experiments. Chapter 6 concludes this dissertation summarizing the main findings, its implications and some final words on future research directions. Fig 2 presents the articles resulting from this research and how they relate to the contents of this dissertation.



Journal paper 1
Tamayo Avila, D., Van Petegem, W., & Libotton, A. (2020). ASEST framework: a proposal for improving
Conference papers 1&2
Tamayo Avila, D., & Van Petegem, W. (2017). An experience using team rules for improving team work in software engineering undergraduate education; EDULEARN 2017 Proceedings
Journal paper 2
Tamayo Avila, D., Van Petegem, W., & Snoeck, M. (2021). Improving Teamwork in Agile Software
Conference paper 3
Tamayo, D., Van Petegem, W., Cruz Ochoa, Y., & Noda Hernández, M. (2018). A correlational study on factors
Iteration
Chapter 2
RQ 1
Chapter 3
RQ 2, RQ 3
Iteration 2
Chapter 4
RQ 4 - RQ 7
Chapter 5
RQ 8 - RQ

Figure 2. Overview of publications related to the chapters of this dissertation

Chapter 2
Teamwork in Software Engineering Education

The content of this chapter is based on the article published in the European Journal of Engineering Education as: Tamayo Avila, D., Van Petegem, W., & Libotton, A. (2020). ASEST framework: a proposal for improving teamwork by making cohesive software engineering student teams.

In this chapter, the existing literature on SEE and teamwork is analyzed. In doing so, the first research question of this doctoral dissertation is answered with the ultimate goal of identifying the basis for our proposal:

Research question 1: What current teaching and learning approaches can be used to set up a framework for improving team cohesion leading to better team performance and team learning as perceived by software engineering student teams?

The chapter is structured as follows: Section 2.1 explains the method that was used to conduct the literature review. Section 2.2 details the data analysis and results. Section 2.3 explains the limitations of this study and section 2.4 concludes the chapter.

2.1 Method

Following the guidelines proposed by (Kitchenham et al., 2009), a systematic literature review was performed. The steps in the systematic literature review method are documented below.

2.1.1 Research question

The research question formulated in this study is: What are the trends on teamwork in software engineering education? In order to answer this research question, the research literature was first generally explored. Then the results of the systematic review were synthesized comprehensively. The process followed is shown in Fig 3.

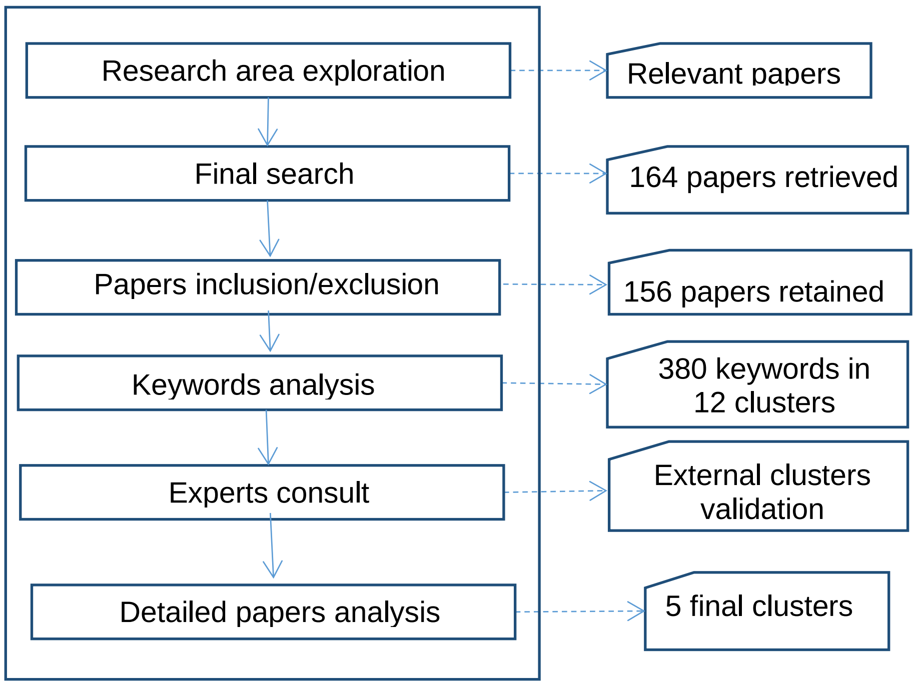


Figure 3. Literature review 1 research process and steps results

2.1.2 Search process

The process started with a manual search in specific conference proceedings and journal papers to explore the research area and focus the scope. The proceedings and journals included in this search were the following: Journal of Systems and Software, Journal of Software: Evolution and Process, Proceedings IEEE Frontiers in Education, and Proceedings Conference on Software Engineering Education & Training. This manual search led to the identification of relevant papers, which were used as a validation list to ensure the reliability and relevancy of the later search and to evaluate the search strings.

2.1.3 Inclusion and exclusion criteria

We then carried out our investigation on the Scopus and Web of Science databases. As the study aimed at determining trends, the articles published in the last ten years at that time were included (between 2006 till 2016). The following criteria were used to exclude papers: papers for which the full text was not available, full books of proceedings, and papers that mostly had a scope not relevant to improve education. A list of these excluded papers is presented in Appendix 2.

2.1.4 Data collection

The search was performed with the search string "software engineering education" AND team*. To assess the quality of the query, we checked that the studies we already knew to be relevant for our search ((Kropp et al., 2016) and (Viljan Mahnic, 2015b)) appeared in the results to ensure that this search query was able to find these papers.

The query resulted in a collection of 164 papers dating from 2006 till 2016. Many papers appeared in both databases. Based on the exclusion criteria explained above, 156 papers were retained.

Two data extraction forms were designed to collect all the information needed to address the review question. A keywords data extraction form was used to make the first analysis. A second form was used to collect data from the detailed analysis.

2.2 Data analysis and results

Five major trends were obtained: collaborative learning, games and gamification, agile methods, global and virtual teams, and real projects resolution and links with industry. The process to obtain these trends is detailed next.

2.2.1 Analysis of keywords

A first analysis of the keywords, as indicated by the authors, was conducted to identify possible points of convergence. For 17 papers keywords were not found. In these cases, the title words were used. In this way, 380 keywords were grouped in twelve clusters. Table 4 shows the 12 clusters and the most representative keywords to illustrate the classification that was made. Appendix 3 contains more detailed information about the analysis of keywords. In this appendix, all the keywords are included. The keywords are grouped in 12 clusters and a group of keywords that were not possible to classify in these clusters.

The initial classification made by the author of this dissertation was put into consideration of the doctoral research supervisor. After having made the changes as suggested, the classification was then considered by three experts to get a second external perspective of the clusters made. In case the same change was proposed by at least two experts, this change was performed.

Two aspects for inviting experts to evaluate the analysis of the keywords were considered: experience in teaching software engineering and research experience in this field. Five experts were contacted, four by email, one in person. Two out of four experts contacted by email responded in time to participate. The three experts were software engineers with 8, 14, and 30 years of teaching software engineering modelling and programming. The one with fewer years in teaching already holds a Ph.D. in computer sciences related to SEE. The other two were Ph.D. researchers in this field at that moment.

Table 4. Keywords clusters

Cluster	Most often used keywords
1. Learning-teaching process and curriculum	self-directed learning; learning by doing; creative thinking; intensive coaching; curriculum design
2. Teamwork management	teamwork; team coordination; teamwork quality
3. Project problem-based learning	project course; capstone project; project-based learning; problem-based learning; capstone course
4. Collaborative learning	cooperative learning; collaborative learning; collaboration skills; computer-supported collaborative learning
5. Real projects resolution and links with industry	real-client; industry collaboration; real-world learning; real-world problems; real-world team projects
6. Virtual and global teams	global software development; virtual teams; distributed software development; global software engineering
7. Gamification	digital game-based learning; role-playing; games for learning; simulation training games; gamification
8. Agile methods	agile methods; agile development; agile learning
9. Engineering education research field	software engineering education; engineering education; computer science education
10. Software engineering core discipline and related contents	project management; team software process; software engineering; requirements specification; software process; process and quality; software development
11. Research and experimentation	experiment; case study; empirical research; experimental study; quantitative evaluation
12. Levels of education	undergraduate education; higher education

The experts considered the classification as appropriate. Only minor changes were suggested regarding the inclusion of keywords initially not classified (marked in red between brackets in Appendix 3). In addition, as Expert 2 suggested, a cluster name changed into “games and gamification” instead of simply “gamification”, considering the Horizon report criteria on technologies in education (Becker et al., 2018) mentioned in her argumentation.

2.2.2 Detailed articles analysis

In a second iteration, a more detailed analysis of the papers in the clusters was done. Here clusters 9 to 12 were not considered since they did not refer to approaches for teaching teamwork. The other eight clusters were carefully studied. As a result, clusters 1-3 were redistributed over clusters 4-8 as will be explained below.

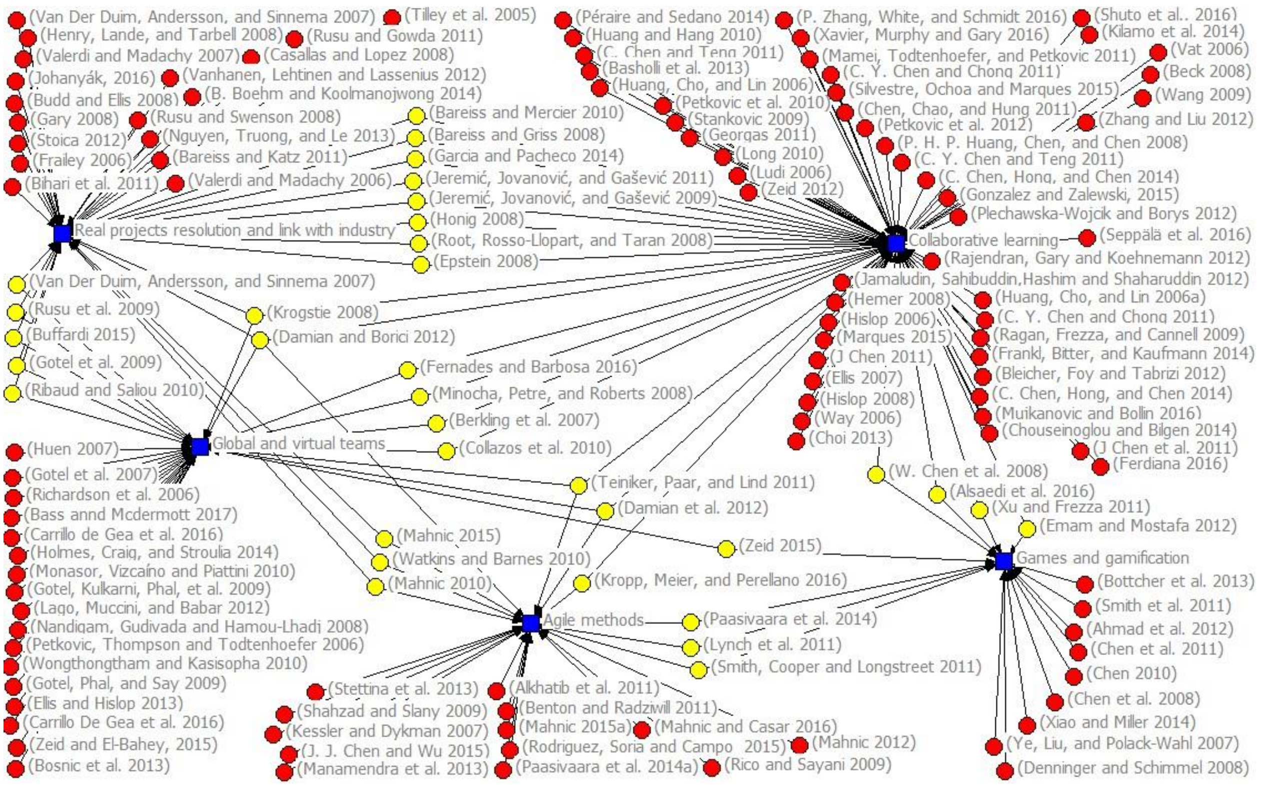
The papers initially included in the first cluster “Learning-teaching process and curriculum” were redistributed over others. For instance, (Stettina et al., 2013) discuss a graduate course design with intensive coaching for agile teams, thus it was included in the “Agile methods” cluster. The work in (Lago et al., 2012) tackles the role of learning by osmosis or the learning of complementary topics by working together with experts in global teams. It was therefore included in the “Global and virtual teams” cluster.

The “Collaborative learning” and “Teamwork management” clusters were merged. We see collaborative learning here as any team-based learning activity in which students have to cooperate to develop software. Collaboration in software teams includes teamwork aspects like communication, coordination, balance of member contributions, mutual support, effort, and cohesion, to capture the nature of team members working together (Hoegl & Gemuenden, 2001). For example, (J. Chen, 2011) proposes an approach to assess teamwork performance in SEE fostering student active collaborative learning by appropriate team management strategies. The “Collaborative learning” cluster was also nourished with some papers from the “Project-problem based learning” cluster, specifically when the authors pay more attention to collaboration and teamwork issues rather than the project-problem based learning itself. For instance, (Shuto et al., 2016) investigate the influence of team discussions on learning effectiveness in problem-based learning.

Other papers from the initial cluster “Project-problem based learning” that put more emphasis on the advantages of learning in real contexts were categorized under “Real problem resolution and links with industry”. For example, (Budd & Ellis, 2008) address project-based learning by using teaching assistants with real-world experience as team managers to maximize the learning from the real-world software development settings. (Vat, 2017) describes scenarios involved in problem-based learning to experience the real-world practice of software development.

Some papers were assigned to both, “Collaborative learning” and “Real problem resolution and links with industry” clusters. A case is the paper of (Garcia & Pacheco, 2014) that integrates TSPi (Team Software Process) student teamwork methodology and project-based learning to improve software project management skills supported by a computational tool to establish an interactive course with local software industry collaboration.

Fig 4 shows the final clusters (blue squares) that represent the five major identified trends: collaborative learning, games and gamification, agile methods, global and virtual teams, and real projects resolution and links with industry.



(Van Der Duim, Andersson, and Sinnema 2007)
(Tilley et al. 2005)
(Henry, Lande, and Tarbell 2008)
(Rusu and Gowda 2011)
(Valerdi and Madachy 2007)
(Casallas and Lopez 2008)
(Johanyák, 2016)
(Vanhanen, Lehtinen and Lassenius 2012)
(Budd and Ellis 2008)
(B. Boehm and Koolmanojwong 2014)
(Gary 2008)
(Rusu and Swenson 2008)
(Stoica 2012)
(Nguyen, Truong, and Le 2013)
(Frailey 2006)
(Bareiss and Katz 2011)
(Bihari et al. 2011)
(Valerdi and Madachy 2006)
(Bareiss and Mercier 2010)
(Bareiss and Griss 2008)
(Garcia and Pacheco 2014)
(Jeremić, Jovanović, and Gašević 2011)
(Jeremić, Jovanović, and Gašević 2009)
(Honig 2008)
(Root, Rosso-Llopart, and Taran 2008)
(Epstein 2008)
Real projects resolution and link with industry
(Van Der Duim, Andersson, and Sinnema 2007)
(Rusu et al. 2009)
(Buffardi 2015)
(Gotel et al. 2009)
(Ribaud and Saliou 2010)
(Krogstie 2008)
(Damian and Borici 2012)
(Fernades and Barbosa 2016)
(Minocha, Petre, and Roberts 2008)
(Berkling et al. 2007)
(Collazos et al. 2010)
Global and virtual teams
(Huen 2007)
(Gotel et al. 2007)
(Richardson et al. 2006)
(Bass annd Mcdermott 2017)
(Carrillo de Gea et al. 2016)
(Holmes, Craig, and Stroulia 2014)
(Monasor, Vizcaíno and Piattini 2010)
(Gotel, Kulkarni, Phal, et al. 2009)
(Lago, Muccini, and Babar 2012)
(Nandigam, Gudivada and Hamou-Lhadj 2008)
(Petkovic, Thompson and Todtenhoefer 2006)
(Wongthongtham and Kasisopha 2010)
(Gotel, Phal, and Say 2009)
(Ellis and Hislop 2013)
(Carrillo De Gea et al. 2016)
(Zeid and El-Bahey, 2015)
(Bosnic et al. 2013)
(Péraire and Sedano 2014)
(Huang and Hang 2010)
(C. Chen and Teng 2011)
(Basholli et al. 2013)
(Huang, Cho, and Lin 2006)
(Petkovic et al. 2010)
(Stankovic 2009)
(Georgas 2011)
(Long 2010)
(Ludi 2006)
(Zeid 2012)
(P. Zhang, White, and Schmidt 2016)
(Xavier, Murphy and Gary 2016)
(Mamei, Todtenhoefer, and Petkovic 2011)
(C. Y. Chen and Chong 2011)
(Silvestre, Ochoa and Marques 2015)
(Chen, Chao, and Hung 2011)
(Petkovic et al. 2012)
(P. H. P. Huang, Chen, and Chen 2008)
(C. Y. Chen and Teng 2011)
(C. Chen, Hong, and Chen 2014)
(Gonzalez and Zalewski, 2015)
(Plechawska-Wojcik and Borys 2012)
(Shuto et al.. 2016)
(Kilamo et al. 2014)
(Vat 2006)
(Beck 2008)
(Wang 2009)
(Zhang and Liu 2012)
Collaborative learning
(Seppälä et al. 2016)
(Rajendran, Gary and Koehnemann 2012)
(Jamaludin, Sahibuddin,Hashim and Shaharuddin 2012)
(Hemer 2008)
(Hislop 2006)
(Marques 2015)
(J Chen 2011)
(Ellis 2007)
(Hislop 2008)
(Way 2006)
(Choi 2013)
(Huang, Cho, and Lin 2006a)
(C. Y. Chen and Chong 2011)
(Ragan, Frezza, and Cannell 2009)
(Frankl, Bitter, and Kaufmann 2014)
(Bleicher, Foy and Tabrizi 2012)
(C. Chen, Hong, and Chen 2014)
(Muikanovic and Bollin 2016)
(Chouseinoglou and Bilgen 2014)
(J Chen et al. 2011)
(Ferdiana 2016)
(W. Chen et al. 2008)
(Alsaedi et al. 2016)
(Xu and Frezza 2011)
(Emam and Mostafa 2012)
(Teiniker, Paar, and Lind 2011)
(Damian et al. 2012)
(Zeid 2015)
Games and gamification
(Mahnic 2015)
(Watkins and Barnes 2010)
(Mahnic 2010)
(Kropp, Meier, and Perellano 2016)
Agile methods
(Paasivaara et al. 2014)
(Lynch et al. 2011)
(Smith, Cooper and Longstreet 2011)
(Stettina et al. 2013)
(Shahzad and Slany 2009)
(Kessler and Dykman 2007)
(J. J. Chen and Wu 2015)
(Manamendra et al. 2013)
(Alkhatib et al. 2011)
(Benton and Radziwill 2011)
(Mahnic 2015a)
(Mahnic and Casar 2016)
(Rodriguez, Soria and Campo 2015)
(Mahnic 2012)
(Paasivaara et al. 2014a)
(Rico and Sayani 2009)
(Bottcher et al. 2013)
(Smith et al. 2011)
(Ahmad et al. 2012)
(Chen et al. 2011)
(Chen 2010)
(Chen et al. 2008)
(Xiao and Miller 2014)
(Ye, Liu, and Polack-Wahl 2007)
(Denninger and Schimmel 2008)

Figure 4. Network of authors and trends on teamwork in SEE

Fig 4 also illustrates the authors (red circles) that tackle them. Circles in yellow are used to mark those authors that refer to several approaches in their studies. Appendix 4 summarizes the main contribution for all the papers included in the network that led deciding its inclusion in the corresponding cluster. Seventeen papers were not included since they tackle a particular aspect not addressed by any other author and are thus hard to categorize. Appendix 5 summarizes the scope of these studies.

These findings lead to answer our **Research question 1** What current teaching and learning approaches can be used to set up a framework for improving team cohesion leading to better team performance and team learning as perceived by software engineering student teams?, as summarized in Table 5. Our framework proposal will focus on four of the five trends in the literature (i.e. collaborative learning, games and gamification, agile methods, and real projects resolution and links with industry), as we address collocated teams rather than virtual or global teams, so we omit the latter. While agile software development is not restricted to collocated teams, studies have demonstrated the effectiveness of agile methods in collocated environments (Gren et al., 2017), showing better results in crucial issues such as productivity (S.D. Teasley ; L.A. Covi ; M.S. Krishnan ; J.S. Olson, 2002). Therefore, the four mentioned trends are assumed as the basis of ASEST framework.

Table 5. Approaches addressed in ASEST framework

Trends	Learning strategies	How included in the framework
1. Collaborative learning	Team-based learning	Collaboration in teams to develop the projects. Use of the win-win model to solve conflicts. Use of team rules to regulate behaviors on communication and conflict resolution
2. Real project resolution and industry links	Project-problem based learning	Development of a capstone project to solve real-world problems
3. Games and gamification	Role-playing gaming strategy	Playing software engineering and Belbin roles for conflicts resolution in teams
4. Agile methods	Agile software engineering education	Use of agile methods to develop the projects in small and self–coordinated teams

2.3 Discussion on validity

The first limitation of this study is inherent to the search for relevant papers. While the systematic search process offers some guarantee for completeness considering the use of two large databases and a sufficient and broad set of keywords, it cannot be affirmed that no relevant studies could have been missed, as unavailable papers were excluded.

Researcher bias regarding the evaluation of the studies to form the clusters is another limitation to consider. While some researchers were included to obtain an external validation of the initial clusters, the second iteration had not the same assessment due to time constraints. However, during the process of analyzing the studies, some of them were further discussed with the supervisor of this dissertation. Therefore, the reasoning behind the inclusion or exclusion of the studies was enriched with a second perspective. Nevertheless, some studies still might be classified differently by another researcher.

2.4 Conclusions

In this chapter, a literature review was conducted to identify current teaching and learning approaches to be used to set up our framework proposal. After an analysis of the keywords and a deeper study of the articles, five trends were obtained: collaborative learning, games and gamification, agile methods, global and virtual teams, and real projects resolution and links with industry.

Our framework proposal will focus on four trends, i.e. collaborative learning, games and gamification, agile methods, global and virtual teams, and real projects resolution and links with industry; leaving out virtual and global teams. In the next chapter, we will explore how these current approaches are included in the phases and steps of our framework proposal.

Chapter 3
The ASEST0 framework

The content of this chapter is based on the article published in the European Journal of Engineering Education as: Tamayo Avila, D., Van Petegem, W., & Libotton, A. (2020). ASEST framework: a proposal for improving teamwork by making cohesive software engineering student teams. Partial results are also published in the EDULEARN 2017 Proceedings as: Tamayo Avila, D., & Van Petegem, W. (2017). An experience using team rules for improving team work in software engineering undergraduate education; and in the 45th SEFI Conference Proceedings as: Tamayo Avila, D., & Van Petegem, W. (2017a). Exploring the Influence of Cohesion on Team Performance Behaviors in Software Engineering Education.

This chapter presents and tests the preliminary version of the ASEST framework (called ASEST0). In the previous chapter, the basis for the framework proposal was established by identifying teamwork trends in SEE. In this chapter, we explain how they are combined to set up the phases and steps of the framework.

The results of two quasi-experiments and a survey are reported as well. These studies sought to investigate the following research questions:

Research question 2: Does the application of the proposed preliminary framework improve team cohesion, team learning, and team performance as perceived by software engineering student teams?

Research question 3: What are the students' perceptions of the proposed preliminary framework?

The chapter is structured as follows: Section 3.1 describes the construction of the proposal. The research model guiding the construction of ASEST0 is firstly discussed. Then, the phases and steps of the framework are described. Section 3.2 discusses the effectiveness of ASEST0. It reports on two quasi-experiments that observed student teams in both academic and professional environments. The results of these studies are discussed from quantitative and qualitative perspectives. In addition, lessons learned from the studies are presented. Section 3.3 compares our proposal with existing related frameworks to assess the novelty of our proposal. Section 3.4 discusses the limitations of the reported studies. Section 3.6 concludes the chapter.

3.1 Construction of ASEST0 framework

This section firstly describes the research model that guided the construction of the ASEST0 framework. Then, the phases and steps of ASEST0 are explained in more detail.

3.1.1 Research model

The literature on factors affecting teamwork in engineering teams is quite extensive. A meta-analysis of ten years of research on team effectiveness (Mathieu et al., 2008) states that factors have been addressed using the IPO (Input Process Outcome) (Mcgrath, 1984) model and several adaptations like IMO (Input-Mediator-Outcome) (Ilgen et al., 2005). These models have been more recently used in software engineering research to study for instance productivity in agile development, team effectiveness, performance, software quality, and job satisfaction (De Melo et al., 2013) (Stewart & Gosain, 2006) (Lu, Xiang, Wang, & Wang, 2011) (S.Faraj & Sproull, 2000) (Acuna, M.Gómez, & Juristo, 2009). The IMO model states that factors are related to inputs (antecedents), mediators (emergent states), or outcomes of the teamwork process.

While studying factors affecting teamwork in engineering student teams, authors mostly focus on the effects of some antecedent or mediator on a kind of team, context, process, or outcome. For instance, (Pazos, 2012) studies the role of goal-oriented attitudes and behaviors as antecedents of conflict management and the subsequent impact of conflict management on team outcomes in virtual teams. (Hsu et al., 2007) study the relationships among computer collective efficacy, outcome expectations, and team performance in the context of collaborative learning.

(Woodley et al., 2019) analyze the role of the mediator group potency on team effectiveness over time for engineering student teams in an engineering design course. Here team rules are addressed as antecedents. The findings of the meta-analysis in (Zarraga-Rodriguez et al., 2015) are used. These authors identified team rules agreement and establishment and task and roles agreement as to the most important antecedents for student teams in higher education. This research focuses on team rules considering that software engineering methodologies establish clear roles and tasks.

The IPO model shows that inputs to the team influence team processes, which in turn lead to team outcomes. The subsequent IMO and IMOI (Input-Mediator-Outcome-Input) models emphasize the mediational role of emergent states. (Mathieu et al., 2008) state that at some stage of the team life cycle performance behaviors (like team learning) are outcomes and at other stages are antecedents or mediators driving performance outcomes. Therefore, although team learning is not seen in this dissertation as an outcome of interactions but as collective discourse activities that teams undertake to yield new insight into a problem (A. C. Edmondson et al., 2007), it is the analyzed outcome in our research model. In addition, research has shown that team members do not automatically engage in team learning (Lehmann-willenbrock, 2017). It is argued that cohesion is an important emergent state and supporting condition for team learning (Bell et al., 2012). Therefore, team cohesion is seen here as the mediator.

Fig 5 shows the research model proposed for guiding the construction of ASEST0 based on the IMO model. As mentioned, team rules agreement and establishment are seen as the input, team cohesion as a mediator whereas team performance and team learning are considered the outcome.

As can be seen in Fig 5, assessment of team member contributions was included as feedback to contribute to the establishment of team rules. Team member contribution has been found correlated with cohesion (Ohland, Carolina, et al., 2012). Besides, some studies show that students express they dislike teamwork when members are not performing equally (Vivian et al., 2016) which leads them to stop cooperating. The solid line running at the bottom shows that teams are expected to develop during the learning process. The dashed arrow at the bottom is in line with the idea of some authors who extended the initial IMO model into an IMOI model underlining the cycling nature of team development (Mathieu et al., 2008). The dashed line at the border delimits the context of the SEE and teamwork within the addressed aspects.

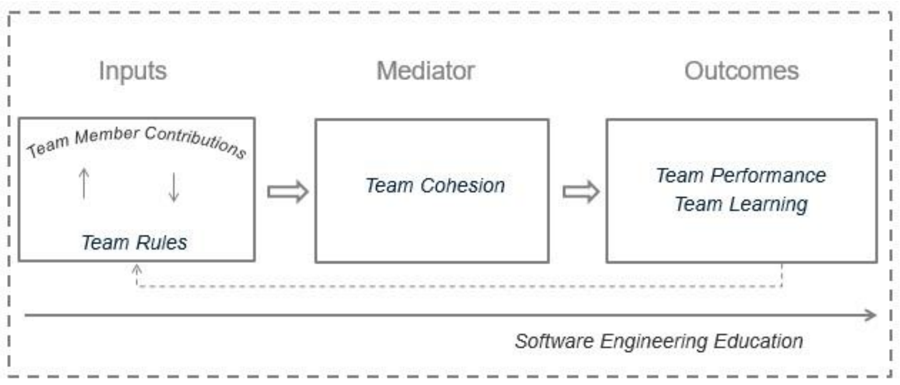


Figure 5. Research model for guiding the construction of ASEST0

To categorize our proposal we look at identifying a classification of frameworks for software engineering education. We assume the classification by (H. Ellis, 2014). In a literature review on software engineering education research (Malik & Zafar, 2012), the authors found this classification covered all the categories of the primary studies they identified. Consequently, we refer to ASEST as a teaching-learning framework, considering that our proposal is a teaching methodology containing several educational approaches to improve teamwork in SEE.

3.1.2 Phases and steps of ASEST0 framework

As mentioned in chapter 2, ASEST0 focuses on four trends, i.e. collaborative learning, games and gamification, agile methods, global and virtual teams, and real projects resolution and links with industry. Therefore, ASEST0 framework combines team-based learning, project-problem-based learning, and role-playing game learning strategies in three phases and eight steps, as can be seen in Fig 6.

The first phase is the preparation to guarantee success. It includes the establishment of the learning scenario based on agile teamwork and project-problem resolution, diagnosis, and training. During step 1 the teams (from 3 to 7 members) are formed, and the roles and projects are assigned to be solved in collaborative ways. The capstone projects are assigned based on different real-world problems. The teams are expected to complete these projects following agile methods. The students identify together in which role they want to contribute and how to self-coordinate tasks assignments.

An individual diagnosis on team working skills on communication and conflict resolution is done (step 2) utilizing the Team Knowledge Test (TKT) questionnaire (Sims-Knight et al., 2002). The results of the diagnosis are discussed in teams and used as starting point to a training (step 3) on communication and conflict resolution skills.

The training uses a role-playing game strategy. The importance of high-quality communication to reduce conflicts in software engineering teams is highlighted by (Tang, 2015). In a study with software engineering teams, (Lewis & Smith, 2008) found that communication difficulties and misunderstandings decreased cohesion and increased conflicts. In their study with IT teams (Somech et al., 2009) found that a cooperative style to solve conflicts, -seen as mutual problems that require common consideration and resolution-, was the most effective way to accomplish their task.

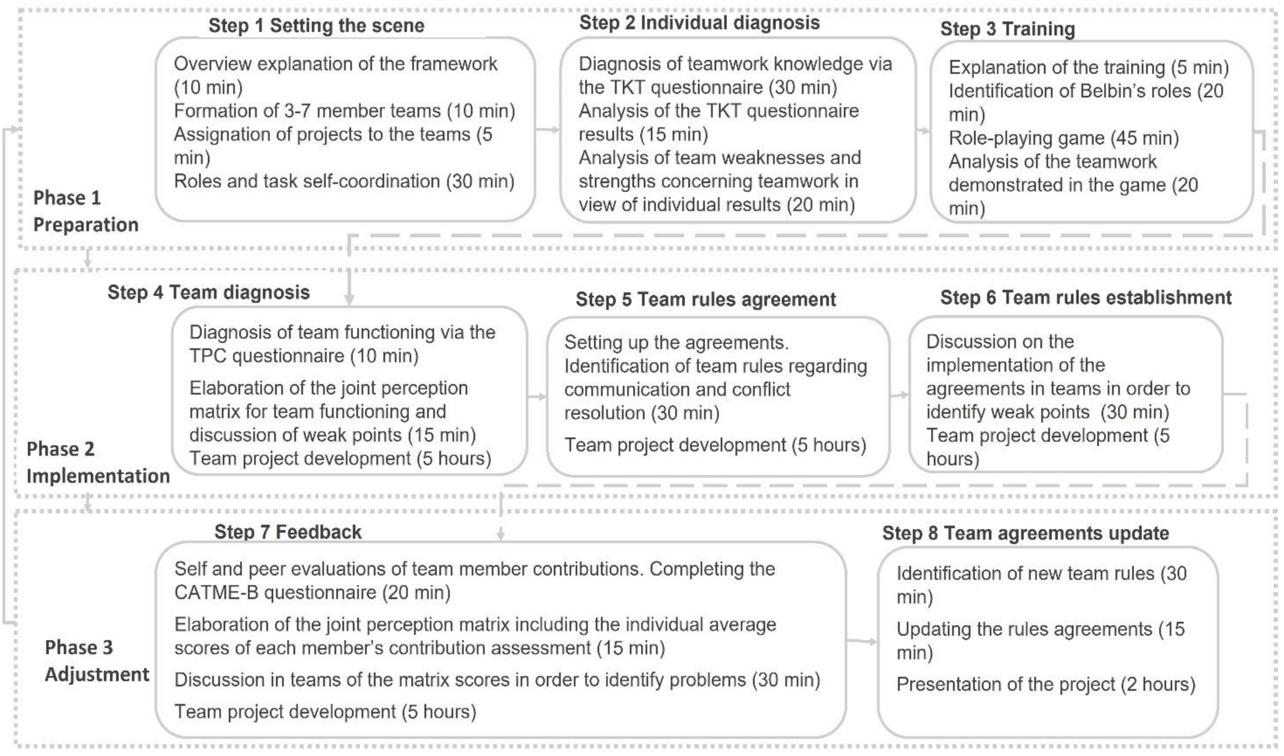

Step 1 Setting the scene
Overview explanation of the framework (10 min)
Formation of 3-7 member teams (10 min)
Assignation of projects to the teams (5 min)
Roles and task self-coordination (30 min)
Step 2 Individual diagnosis
Diagnosis of teamwork knowledge via the TKT questionnaire (30 min)
Analysis of the TKT questionnaire results (15 min)
Analysis of team weaknesses and strengths concerning teamwork in view of individual results (20 min)
Step 3 Training
Explanation of the training (5 min)
Identification of Belbin's roles (20 min)
Role-playing game (45 min)
Analysis of the teamwork demonstrated in the game (20 min)
Phase 1 Preparation
Step 4 Team diagnosis
Diagnosis of team functioning via the TPC questionnaire (10 min)
Elaboration of the joint perception matrix for team functioning and discussion of weak points (15 min)
Team project development (5 hours)
Step 5 Team rules agreement
Setting up the agreements.
Identification of team rules regarding communication and conflict resolution (30 min)
Team project development (5 hours)
Step 6 Team rules establishment
Discussion on the implementation of the agreements in teams in order to identify weak points (30 min)
Team project development (5 hours)
Phase 2 Implementation
Step 7 Feedback
Self and peer evaluations of team member contributions. Completing the CATME-B questionnaire (20 min)
Elaboration of the joint perception matrix including the individual average scores of each member's contribution assessment (15 min)
Discussion in teams of the matrix scores in order to identify problems (30 min)
Team project development (5 hours)
Step 8 Team agreements update
Identification of new team rules (30 min)
Updating the rules agreements (15 min)
Presentation of the project (2 hours)
Phase 3 Adjustment

Figure 6. ASEST0 framework phases and steps

Therefore, during the training conflicting situations are simulated and the students are asked to collaborate to solve them playing different roles in the software team. Belbin's roles are used as well. A Belbin questionnaire allows identifying team roles based on behaviors that individuals adopt when participating in a team (Kidd & Belbin, 2006). The use of Belbin's roles pursues to foster mutual trust and understanding and to build productive workplace relationships. In preparation for role-playing gaming, the students exercise nonverbal messages understanding, and emphatic communication. A sample of these communication exercises is presented in Table 6.

*Table 6. Sample of communication exercises**

Communication skill	Exercise description
Understanding nonverbal messages	A volunteer comes to the front of the room. This person acts some actions and the rest of the group interprets each action. This exercise is then done in teams. Each team member is asked to perform once. After each performance, the team discusses the nonverbal signals and their potential meanings considering if they interpreted the nonverbal signals. When all rounds are completed, the team discusses what nonverbal signals they may use in times of conflict that would adversely affect resolution and those would indicate a willingness to work toward resolution and collaboration.
Empathic listening	In peers, one student is provided with a statement and the other has to provide an empathic response. In a second round, the

	roles change and a new statement is used. Examples of statements: "*I don't know what to do. Every time I work with Maria on a project she never does her fair share. I just don't want to work with her anymore*" "*I've just had it. The work is really getting to me. We've been pushing and pushing and* *pushing to meet all of our deadlines and more work just keeps coming. I don't know if I can keep up this pace. I'm really tired.*" To reach the response, the students are asked to exercise these steps: • Repeat verbatim the content of the communication—words only, not feelings • Rephrase content—summarize their meaning in your own words • Reflect feelings—look more deeply and begin to capture feelings in your own words—look beyond words for body language and tone to indicate feelings • Rephrase content and reflect feelings—express both their words and their feelings in your own words To facilitate this, the students are provided with some useful phrases to show understanding: *What I'm hearing is*… *Your feeling now is that*… *You must have felt*… *Your message seems to be, "I*… *In other words*… *I'm sensing that you*… *As you see it*… *If I understand you correctly you*… At the end of the exercise, the students reflect on how understanding the stories of others would help to overcome conflicts in their teams.

* Adapted from (Scannell, 2010) and (KonterraGroup, 2015)

Conflicting situations in a software team are then presented to the teams (see a sample in Table 7). The students are asked to analyze the conflicting situations from their Belbin's role view while playing their software team role. They are supposed to contribute to the solution according to the criteria of (Belbin, 2013) (see a sample of a guide for students on how to contribute in Table 8.

Table 7. Sample of conflicting situations descriptions

Scenario 1 (*adapted from* (*Practical Application: Conflict Resolution Scenario*, 2016)): Wilson and Peter work as analysts in a software company and they love their job very much. They are hardworking and always on time at work. Janice and Charles are designer

engineers. They are also hard workers and very much like their work. Janice and Charles are often very critical of Wilson and Peter's work. Sometimes Wilson and Peter feel very much uncomfortable about this but they never talk about this matter. One day Wilson and Peter worked until very late. The team had to present an initial project proposal to a new client. The team leader, Georges joined them until they finished the proposal. The company has a policy that all team members have to agree on a team proposal before it is presented to the client. However, Janice and Charles said that they could not stay. Out of concern, Georges reminded Janice and Charles about the company policy and the impossibility to change the meeting to the day after. Janice and Charles said they will stick to the final proposal that Wilson, Peter and Georges agree. The day after, Janice and Charles came up with the same inquiring behavior some minutes before the meeting. Upset with Janice and Charles's behavior, Wilson, and Peter decided not to talk to them anymore
Scenario 2 (adapted from (Tikanov, 2014)): Wilson is the Scrum Master of a team that has been working for several weeks on a project that is planned to be released very soon. The team emerged from two separate teams. John and Peter are previous teams' leads –very experienced C++ developers, who know the business domain and the product very well. The problem is they clash daily mainly due to different technical opinions like we should/ should not use this pattern, code style, etc. I see communication between them is harder every new day. Each one separately is technically good and able to lead the team but they do not cooperate at all. The rest of the team is frustrated. Some guys support one or another side, others try to remain neutral. In addition, developers are repeatedly late in delivering stories for testing then the work of the team is affected.

*Table 8. Guide for students in Belbin's role contributing to conflict resolution**

Belbin's role	Contribution
shaper	move the group forward and stop complacency
completer finisher	take care of high standards
plant	provide new ideas
monitor evaluator	choose which idea would work best
coordinator	orchestrate the team effort
implementer	provides a practical view while making suggestions
resource investigator	find external resources
Team worker	stop arguments and pull the team together and improve the atmosphere
specialist	contributing from certain areas of expertise

The students are asked to solve these situations utilizing a win-win model. The criteria of (Frankl et al., 2014) are used. These authors argue on the role of successful collaboration in SEE. An example of a win-win model is the "Interest-Based Collaborative Problem Solving" approach (O'Leary & Bingham, 2007). This approach was proposed to resolve conflicts in collaborative networks and it builds on the "principled negotiation" method (Fisher & Ury, 2011). The principled negotiation states four bases: "1. People: Separate the people from the problem. 2. Interests: Focus on interests, not positions. 3. Options: Generate a variety of possibilities before deciding what to do. 4. Criteria: Insist that the result be based on some objective standard." Table 9 presents a guide of conflict resolution activities that have been adapted from (O'Leary & Bingham, 2007) as a proposal to lead the conflict resolution activities during the training.

*Table 9. Conflict resolution activities**

1. Identify the problems (e.g. interest/vision/technical/role/process/personal disagreements) and their sources (e.g. communication/ behaviors of particular members/ emotional issues/ context). Analyze the situation objectively without judging people. Then, summarize the teams' findings defining them as challenges to be solved together. In doing so, use "how-to" to phrase them
2. Understand each other interests: disclose, listen, and ask. From your team role answer what do I need and why do I need it, then identify what others need and why they need it
3. Look for ways to create value before claiming value. From your team role evaluate your priorities as well as the importance each member attributes to each issue. Look for shared interests
4. Generate a set of mutually satisfactory options. Write every idea down without judging. No idea is too dumb or silly. If the team gets stuck, go back to step 2 and review what people's interests are. Ask yourself: from my team role, how I can contribute to solving this situation? Keep in mind that options should not contradict others interest
5. Evaluate the options. Assess if the options meet most or all the essential interests of the parties involved. For each option, answer if it is doable for the team and if it is acceptable to all stakeholders Reach an agreement by selecting/modifying options that meet needs the most

*Adapted from (O'Leary & Bingham, 2007)

The second phase is the core of the framework. The aim is to set and establish team rules agreements to norm communication and conflicts resolution. The use of a cooperative team rules agreement pursues to mature the collaborative learning environment encouraging students to active learning. Other authors have previously stated the importance of team rules to achieve higher levels of performance for software engineering student teams (Monaghan et al., 2015) and to achieve a behavior change among software engineers (Lenberg et al., 2017).

To assist students to write the agreements, a diagnosis on team functioning (step 4) is done by using the (Powers et al., 2002) questionnaire. The lessons learned from the individual diagnosis in step 2 and the training in step 3 contribute to that end as well. The agreements are set in step 5. Table 10 presents the team rules agreement process based on the proposal in (Lencioni, 2005) for conflicting norming.

*Table 10. Team rules agreement activities**

1. The team reviews the lessons learned from step 2 (TKT responses, training)
2. The team identifies problems on team functioning from the TCP questionnaires
3. Team members write down their individual preferences and expectations relating to acceptable and unacceptable behaviors around discussion and debate. Areas might include use of language, tone of voice, emotional content, expectations of involvement and participation, avoidance of distractions, or timeliness of response
4. Each team member reviews their preferences with the rest of the team, while someone captures key areas of similarity and difference
5. Discuss collective preferences, paying special attention to areas of difference. Arrive at a common understanding of acceptable and unacceptable behavior that all members of the team can commit to. The team leader may have to play a key role in breaking a tie
6. Formally record and distribute behavioral expectations around conflict. Keep the agreement short and to the point Ensure that all members believe in the agreement and are willing to incorporate it into the team norm. Consider that the agreement can be revisit and updated

Step 6 refers to the establishment and implementation of the agreements. This assessment aims to contribute to students' awareness on the usefulness of team's agreement by identifying broken rules and not engaged team members.

The third phase is focused on the adjustment of the agreement. After some weeks of working together (i.e three to four weeks), the students are asked to self and peer evaluate team member contributions (step 7). This self and peer evaluation seeks to make students aware of the team collaboration and improve it. To pay attention to how members contribute to the team is important for the success of agile software teams (Lindsjørn et al., 2016) and will avoid conflict situations that can arise from social loafing (Borrego et al., 2013). Consequently, team member contributions are evaluated on five areas, according to high, medium, and low levels of team performance behaviors, as proposed by (Ohland, Loughry, et al., 2012), i.e. contributing to the team's work, interacting with team mates, keeping the team on track, expecting quality and having relevant Knowledge, Skills, and Abilities (KSAs). Students are asked to agree on rules improvements after obtaining such feedback (step 8).

3.2 Validating the ASEST0 framework

To test the ASEST0 framework, two studies were conducted. The ASEST0 framework with its three phases and eight steps is to be seen as the input of the IMO model. In study 1, the ASEST0 framework was examined through a study over a group of subjects that received the intervention compared with a group that did not receive it. In study 2, the

ASEST framework was examined by means of a study over a single group of subjects. In this case, the study aimed to know the perception of students performing in a company to enhance the proposal from the industry perspective.

The participant, context characteristics, and the quantitative and qualitative analyses of the results for both studies are described next. The variables measurement is presented first. In both cases, the SPSS software version 25 was used to perform the tests. Nvivo 12 software was used to analyze the qualitative data.

3.2.1 Variables measurement

The Group Environment Questionnaire (GEQ) (A. V. Carron et al., 1985) was used to measure cohesion. Team performance and team learning were measured using the instruments proposed by (A. Edmondson, 1999). A five-point Likert scale was used in all cases, from 'strongly disagree' to 'strongly agree' for team cohesion, from 'very inaccurate' to 'very accurate' for team performance, from 'never' to 'always' for team learning. The surveys can be found in Table 11. Since our hypotheses are formulated at the team level, individual answers were aggregated to the team level.

*Table 11. Adapted questionnaires to measure team cohesion, team learning and team performance**

Team learning (5 point scale from never to always)
We regularly take time to figure out ways to improve our teamwork processes
This team tends to handle differences of opinion privately or off-line, rather than addressing them directly as a group**
The members of this team seek all possible information they need for project development from other sources outside the team such as customers or other stakeholders, or domain experts
This team frequently seeks new information that leads us to make important changes
In this team, someone always makes sure that we stop to reflect on the teamwork process
The members of this team always express a frank opinion about the issues under discussion
We invite people from outside the team to present information or have discussions with us
Team performance (5 point scale from very inaccurate to very accurate)
Recently, this team seems to be slipping a bit in its level of performance and accomplishments **
Those who receive or use the work of this team often have complaints about our work **
The quality of work provided by this team is improving over time
Critical quality errors occur frequently in this team**
Others around us who interact with this team frequently complain about how it functions**
Team cohesion (5 point scale from strongly disagree to strongly agree)
I do not enjoy being a part of the social activities of this team**

I'm not happy with my participation in the project**
I am not going to miss the members of this team when the project ends**
I'm unhappy with my team's level of desire to successfully end the project**
Some of my best friends are on this team
This team does not give me enough opportunities to improve my personal performance**
I enjoy other parties rather than team parties**
I do not like the style of work on this team**
For me, this team is one of the most important social groups to which I belong
Our team is united in trying to reach its project goals
Members of our team would rather go out on their own than get together as a team**
We all take responsibility for any failure or poor performance by our team
Our team members rarely party together **
Our team members have conflicting aspirations for the team's performance**
Our team would like to meet sometime after the project is completed
If members of our team have problems, everyone wants to help them so we can get back together again
Team members do not like to meet after work on the project**
Our team members do not express themselves honestly about each other's responsibilities in completing the project**

*Adapted from (A. V. Carron et al., 1985) and (A. Edmondson, 1999), **Reverse scored

The instruments were adapted to software teams and translated into Spanish by a language specialist, followed by a wording revision of the questions. In addition to two software engineering teachers, a psychologist and a sociologist revised. Just a few minor changes, regarding cultural issues to adjust the language-translation were suggested. To test the reliability of the instruments, the new versions of the questionnaires were piloted and Cronbach's Alpha tests were run.

The sample for this pilot was composed of 21 volunteer students who were participants of an event of the Faculty of Informatics Engineering at the University of Holguin (Cuba). This event is organized by the students to present capstone projects in the form of posters or brief presentations to share experiences among students of different years of the program. For the three instruments, the Cronbach’s Alpha values were above the cut-off point (0.7), indicating good internal consistency (0.811 for team cohesion, 0.752 for team performance, and 0.720 for team learning).

3.2.2 Quasi-experiment 1

The experimental group consisted of thirty-four subjects performing in seven teams. The control group consisted of twenty-four students performing in five teams. These teams of three to six members were groups of individuals conveniently available to study. Table 12 describes the composition and demographics per sub-group for both the experimental and control group. All students take the program on Informatics Engineering at the University of

Holguin, in Cuba, a five-year-long program. All the students performing in the third year of the program at that time were included.

In both groups of subjects, the teams were formed at the beginning of the course by self-selection. The teams in both groups were involved in two courses at the same time, learning topics on software engineering modeling and management. The company was starting moving from RUP methodology to agile approach. They used Iconix to develop their projects, an agile use-case-driven object modeling process. The researcher conducted the intervention but two other teachers performed as main teachers for both courses. The intended learning outcomes of the course Software Engineering I related to modeling of information systems applying Iconix while performing in teams. The intended learning outcomes of the course Software Project Management related to elaborating, planning, monitor and executing a capstone project.

Table 12. Groups composition and demographics in quasi-experiment 1

	Control group	Experimental group
Subjects	24	34
Nationality	50 % Cubans, 50 % Angolans	59 % Cubans, 41 % Angolans
Teams	5	7
Gender	4F 20M	7F 27M
Age range	21 to 27	21 to 29

Table 13 shows the activities per week, linked to the ASEST0 steps, for the experimental and control groups. The courses for the experimental and control groups of students had the same contents and similar teaching and learning activities. However, in addition to the learning and teaching activities in the courses, the students in the experimental group had one workshop in those weeks the steps of ASEST0 were done (except for step 1 and 6 that were done together with the courses teaching and learning activities). The teaching in the control group did not include any team working activity beyond the capstone project development itself. The activities done during the intervention are further described below.

Table 13. Course schedule and intervention in quasi-experiment 1

Week	Teaching and learning activities of the courses	ASEST0 steps
1	Lecture/ Workshop (Introduction to SE) Lecture/ Workshop (Introduction to software project management)	1

2	Lecture/ Workshop (Domain modeling) Lecture/ Workshop (Initiation process)	-
3	Laboratory (Domain model) Team project (project definition, domain model)	-
4	Lecture/ Workshop (Use cases)	Variables measurement
5	Laboratory (Use case) Lecture/ Workshop (Planning process)	2
6	Team project (use case model, use cases description, Gantt/Pert diagram)	3
7	Lecture/ Workshop (Estimation) Laboratory (Planning automation)	4, 5
8	Team project (cost estimation, detailed project planning)	
9	First project oral presentation (Requirements review, project scope analysis, and plan)	6
10	Lecture/ Workshop (Robustness analysis)	
11	Lecture/ Workshop (Process monitoring) Laboratory (robustness diagram)	7, 8
12	Team project (Robustness model, project progress reports)	Variables measurement

3.2.2.1 Intervention

Phase 1:

During the first week step 1 was done, yet teachers just intervened to explain the course design and the activities of the framework. They also assigned the projects and explained the responsibilities of their roles. The teams were formed by selection. Team leaders were assigned to be responsible for the project management activities.

Following Iconix roles, analysts were responsible for identifying real word domain objects, defining the behavioral requirements, and performing robustness analysis to disambiguate the use cases and identify gaps in the domain model. Designers were responsible for allocating behaviors to objects in sequence diagrams as for the static model (class diagrams). According to the general course schedule, those activities for which designers were responsible were planned to be done after the intervention finished. Therefore, to not interfere with the course planning, students were told that the fact that they were assigned the role of designer throughout the intervention did not mean that they were the ones to carry out the activities, but that they were also responsible for ensuring their success. Thus, all students were expected to contribute to all activities while the teams self-organized the tasks.

The teams were asked to have a balanced number of roles (except the team leader role) with emphasis on analysts (due to the nature of the course more responsibilities for this role were needed), but the teachers did not interfere in the distribution. Teachers explained the role's responsibilities along with the projects and students voluntarily chose and agreed. Their preferences for particular projects were also considered when assigning the projects. Table 14 presents the composition of the teams and the general project scope.

Table 14. Team composition and projects scope from quasi-experiment 1

Team members and roles	Project scope
Team 1, 3 members 1 Team leader 1 Analyst 1 Designer	Information system to process patients' data at the Cancer Center in Holguin
Team 2, 6 members 1 Team leader 3 Analysts 2 Designers	Information system to process residence management data at the University of Holguin
Team 3, 5 members 1 Team leader 2 Analysts 2 Designers	Information system to process human resources data and generate related reports at the company "Ceramica Blanca"
Team 4, 6 members 1 Team leader 3 Analysts 2 Designers	Information system to process data related to indicators of scientific research and innovation at the University of Holguin
Team 5, 5 members 1 Team leader 2 Analysts 2 Designers	Information system to manage the process production data at the company "Ceramica Blanca"
Team 6, 4 members 1 Team leader 2 Analysts 1 Designer	Web site for online selling of reservations and publicity for the "Islazul" entertainments at Pernik hotel
Team 7, 5 members 1 Team leader 2 Analysts 2 Designers	Information system to manage data of research projects for the Nickel research center "CEDINIQ"

After three weeks of performing together, the variables team cohesion, team learning, and team performance were measured (using the surveys in Table 11) and the first workshop was done. The variables were measured considering the criteria of (Coultas et al., 2014), who state that emergent states, such as cohesion, require time to form.

During workshop 1 step 2 was done. After answering the TKT questionnaire (just communication and conflict resolution questions) the students were asked to check their responses to the correct ones. They were then asked to discuss in teams the results of this individual diagnosis. The students had to show their responses and identify what aspects f the questionnaire the team members more frequently answered wrong/right, to what teamwork issue were these wrong answers related to (communication or conflict resolution), and how these aspects could impact the teamwork.

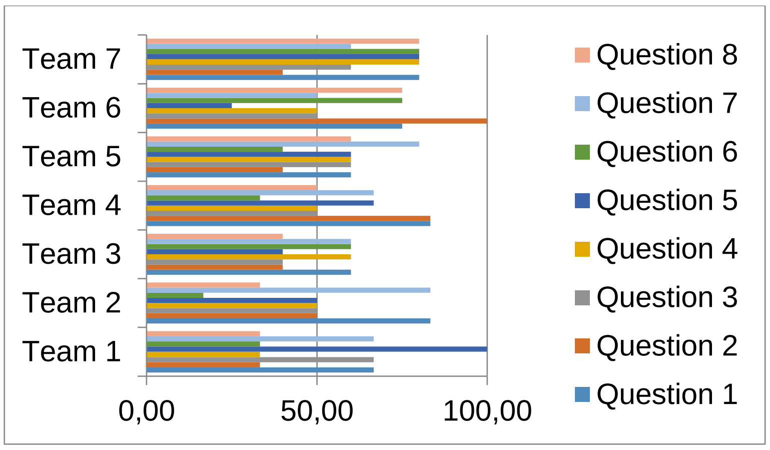
Fig 7 shows the results of these questionnaires. The percentages of right answers per question regarding communication and conflict aspects showed the majority of questions ranges between 33 and 60 % and just a few questions scored above 70 %, which denoted a low level of knowledge on communication and conflict resolution in teams.

During workshop 2 (step 3), the training on communication and conflict resolution skills was done. In preparation for the training, the teachers reminded the students of what they learned from the TKT questionnaire they answered in the previous workshop (step 2) by asking them to write down and share with their teammates what they already knew about this topic, what they expected from their teammates during the exercise and how they could contribute to the team.

The students identified their Belbin’s roles, they were provided with information on these roles and they were asked to discuss in teams what roles were present in their teams, what roles were lacking, and how the presence/ absence of these roles could affect the teamwork. Table 15 presents a sample of the ideas the students expressed.

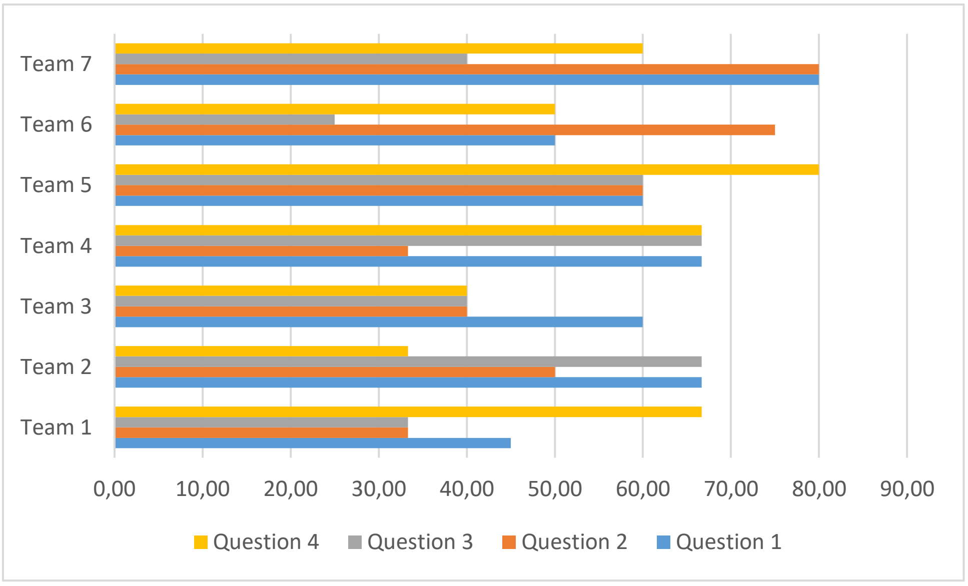
The teachers explained the definition of conflict, some negative consequences of conflicts for a software team, some positive outcomes of a conflict, and how conflicts can be resolved cooperatively. They also explained the characteristics of effective communication, barriers that can affect it, and nonverbal and empathic communication. To prepare students for better communication along with role-playing gaming, some exercises were done (a sample is presented in Table 6). The objective here was to gain an understanding of nonverbal messages and empathic listening.

Conflict resolution questions



Q1. When there is a disagreement or difference of opinion in your team, it is generally best to…
Q2. When dealing with a team member, who is not doing his/her fair share of the work, it is best to…
Q3. When you and another team member are having trouble communicating, which is the worst thing for you to do?
Q4. You have gotten quite angry in a team meeting. Which of the following is the least productive thing you could do?
Q5 In order to increase the chances of everyone doing their fair share of work, a team ought to…
Q6. Effective discussions of team business are often made difficult by people who are argumentative or dominating or disorganized. No matter what their problem, to get the meeting moving forward you need to...
Q7. If a member of your team is hostile or critical it is generally useful to…
Q8. Two members of your team have a genuine disagreement (not just miscommunication or personality conflict). Which of the following would be most likely to lead to a resolution?

Communication questions



Q1. When you are listening to other people offering their ideas, it is useful to…
Q2. When receiving feedback from your team members, it is generally useful to…
Q3. When expressing an idea or presenting some information, it is best to…
Q4. If a team member is expressing an opinion different from your own, it is generally helpful to…

Figure 7. Results of the TKT questionnaire from quasi-experiment 1

Table 15. Sample of reflection on Belbin's roles analysis from quasi-experiment 1

Belbin roles in the team	Reflections
2 Monitors evaluator 1 Teamworker	The facts that "monitor evaluator" can be overly critical and "teamworker" can be indecisive in crunch situations might lead to unsolved team issues and unreachable objectives
1 Shaper 2 Plants 1 Teamworker 1 Coordinator 1 Specialist	A varietal of Belbin roles can strengthen teamwork. Lack of a more critical and visional view could act against the quality results
2 Specialists 1 Coordinator 2 Implementer	The roles that are present in our team provide mostly a "practical style". This could help to stay on task get the work done.
2 Plants 1 Resource investigator 2 Teamworkers 1 Specialist	The presence of creative roles in our team can enrich our work. There is a lack of a more practical view
1 Monitor evaluator 1 Plant 1 Completer finisher 1 Implementer 1 Resource investigator	Belbin roles are diverse in our team. However, collaborative profiles are not well represented
2 Monitors evaluator 2 Specialists	Just two roles are present in our team. However, it can be complemented one with another as monitors think strategically and specialist people are focused "teamworkers"
3 Teamworkers 1 Resource investigator 1 Coordinator	We believe that our team is well balanced according to Belbin roles. We do not consider the roles that are not present will affect our work

The teams were then provided with descriptions of conflicting situations (similar to those in Table 7). Team members were expected to collaborate following the activities described in Table 9. They assumed the characters corresponding to their team roles and contributed to solving the situation from their Belbin's role viewpoint by following the guide in Table 8. The students were able to solve the situations and some teams addressed more than one dilemma.

Table 16 presents some dilemmas and solutions the students proposed. At the end of the training, the students were asked to debate on how they felt during the training. The students shared with their teammates if anyone felt uncomfortable and why, and what was the most important lesson learned. All teams stated they solved the conflict without any uncomfortable situation. Some lessons learned expressed by the students can be found in Table 17.

Table 16. Sample of conflicts resolution activities outcomes from quasi-experiment 1

Dilemmas or challenges	Solutions
How to improve communication between the team leader and the rest of the team members?	Communicate frequently and openly, encourage input and reciprocal feedback
How to create a shared vision for the team?	Encourage to share ideas and expectations, set clear team objectives
How to effectively communicate?	Get training, choose the best way for the team to communicate, pay attention to non-verbal communication, foment courtesy and respect
How to support members with personal problems?	Get to know each other better, do fun stuff together
How to engage all team members in a solution?	Stimulate fruitful debate, establish clear goals, clarify purposes
How to support members struggling to solve a technical task?	Get training, provide useful resources to help solve the task, pair programming
How does make engage team members take responsibility for their tasks?	Role clarity, clear progress, and tracking
How to fairly reward team member efforts?	Boost morale, recognize people achievements and outstanding ideas
How to effectively involve members to meet deadlines?	Build trust, build team spirit, clarity roles, and responsibilities

Table 17. Sample of lessons learned from the training from quasi-experiment 1

Team	Reflections
Team 1	The most important lesson for us was that we should not get passive at problems but to face them and discuss what is happening
Team 2	It is important to assist team members when needed, but also to set boundaries to prevent unacceptable situations in a team, and if someone crosses it, the team should discuss it immediately
Team 3	Team members should manage disputes within the team and solve discrepancies without involving clients. It is important to openly share ideas and understand decisions before
Team 4	The team should consider different ways of doing things. All members should

	share the work and work together to get the work done
Team 5	It is important to set team goals and every member should stick to them
Team 6	Our team agrees that it is never a good option to stop talking but to debate with respect, identify problems and get understanding. That is the most important lesson for us
Team 7	Trusting team members is essential for a team, share information and stick to the team’s decisions

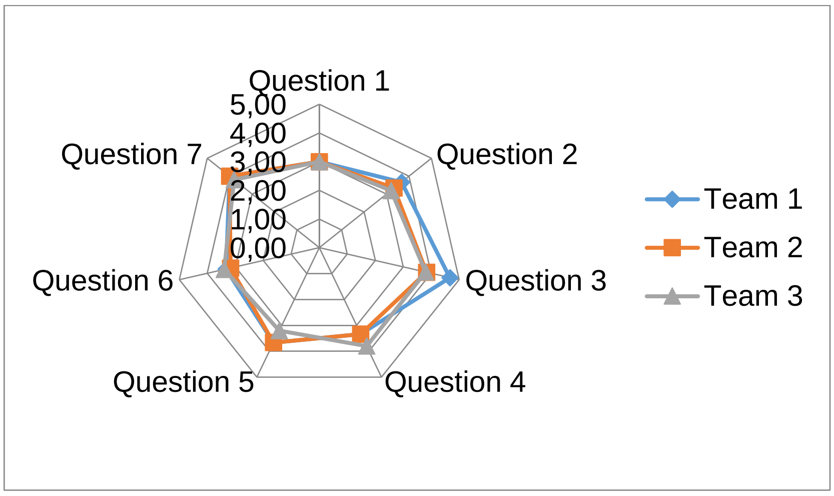
Phase 2:

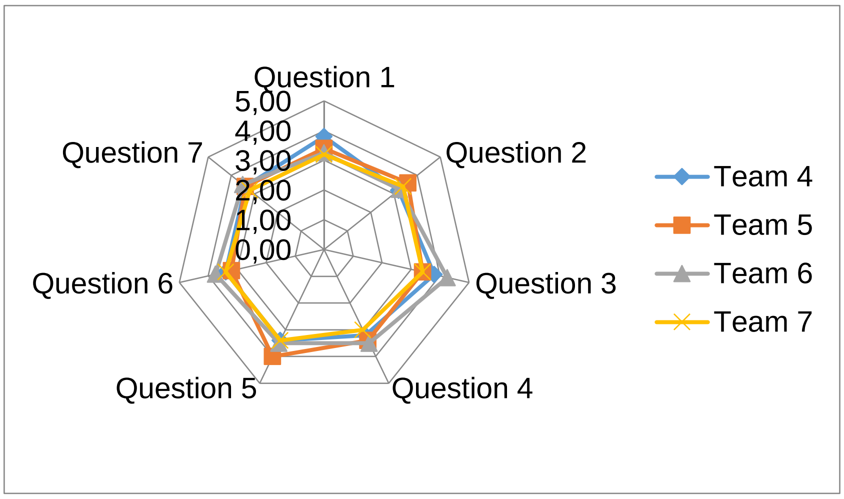
During workshop 3 (step 4) the students assessed the team functioning by using (Powers et al., 2002) questionnaire. Although the students already had taken lessons learned from the individual diagnosis and training on communication and conflict resolution, the team functioning assessment helped them to become aware of how they were doing on these matters. This served as a starting point to agree on policies to regulate communication and conflict behaviors (step 5). Responses were processed at the team level using averages (included in Fig 8). The responses were almost all above 3 points on average (out of 5), however, just question 3 (related to expressing an idea or presenting some information) scored closer to the maximum.

Considering the difficulties identified in this assessment, the students then set the team rules agreement, following the activities described in Table 10. The agreed rules can be found in Table 18.

After two weeks of performing on agreed rules, step 6 was done during the first project oral presentation of their capstone projects. The teams were asked to report if there were team members that were not fully engaged in the agreements and if there were broken rules or other incidents.

In general, team members expressed the willingness of team members to stick to the rules. However, there were some problems in applying them. Team 1 referred to the absence of members to some meetings, without major problems however as they assumed what was agreed by the rest of the team in the meetings. Team 2, 5, 6, and 7 reported problems regarding rules of duty on time. Related to this, team 4 pointed out that not all members shared project information as quickly as needed for others to complete their tasks. These issues were considered by the teams to update their agreements in the next phase.





Q1. As a team we find it difficult to accept criticism openly and non-defensively. (reverse scored)
Q2. When conflict arises in the team, it is likely to be a battle or, at best, a waste of time. (reverse scored)
Q3. My team encourages differing opinions to be expressed.
Q4. When arguments break out, my team members are able to step back, calm down, and work out our differences.
Q5. My team members criticize ideas, not each other.
Q6. My team may agree on a solution but not every member "buys into" that solution. (reverse scored)
Q7. My team ignores conflicts among team members. (reverse scored)

Figure 8. Team functioning assessment results from quasi-experiment 1

Table 18. Sample of agreed teams rules from quasi-experiment 1

Team	Rules agreement
Team 1	Discuss problems and conflicts openly and as soon as they happen Assign tasks in fairly way, according to members preferences and skills Share all project information Show up to meetings on time Everybody engages with their responsibilities Don't wait for meetings to raise impediments (update*)

	Speak for yourself, not on behalf of others (update) Bring innovative ideas to meetings (update)
Team 2	Tasks are assigned fairly, according to member skills and responsibilities Everybody does his/ her work on time Problems and conflicts are discussed immediately and a consensus should be reached to avoid bigger problems in the future Celebrate milestones accomplished on time It is ok to be wrong; It is ok to ask for help (update) Assist team members when needed (update)
Team 3	Identify problems of members to get the work done, so others can support them when needed Recognize members' strengths so the team can take advantage of that To yell or attack others is not allowed. Discrepancies are solved with understanding. We agree that as human beings emotions are inevitable, but respect should be predominant All ideas are relevant. We debate finding the best solution together (update) We peer- review all tasks done (update) Recognize members' weaknesses so the team can support them (update)
Team 4	Listen to different perspectives on how to do things Discuss conflicts respecting others perspectives Bring to debate the team agreement in our meetings Pay attention to others while expressing their opinions, keep eye contact and observe their language body Encourage members that regularly keep silent to participate Assign tasks fairly and monitor progress Support members struggling with their tasks When problems arise we quickly communicate them to the team (update) Ask questions and seek help when needed (update)
Team 5	Everybody's opinions are listening and taken into account If someone has problems that affect his or her work to be done, it should be communicated immediately and his/ her work should be fairly distributed among the members In case of disagreements, all members propose positive and constructive ideas that help to reach an agreement friendly We are open to new approaches as well as listening to new ideas (update) We share in meetings where we are stoked, so others can help us (update)
Team 6	All members should be aware of others task perspectives and progress so the team can provide the best solution Recognize progress of team members and stimulate them to continue doing a good job Each member has the responsibility to do their part of the job in time We encourage generating creative ideas (update) We recognize outstanding ideas (update)
Team 7	All members stick to the team's decisions All members should have access to all the information of the project We should carefully pay attention to the opinions of others, listening without interruptions. Then express our opinion with respect Every member is expected to do his/ her assigned work on time Conflicts should be used as opportunities to improve the teamwork Provide assistant to others (update) Seek help outside the team when needed (update)

* Update: rules included in step 7

Phase 3:

During workshop 4, steps 7 and 8 were done. The students' self- and peer-evaluate the team member contributions in five areas using a scale from 1 to 5. The results of this assessment and rules added to the agreements can be found in Fig 9. Although no area obtained the maximum, areas 1 (Contributing to the team's work) and 2 (Interacting with teammates) scored better than the rest. Thus, the teams proposed new rules for areas 3 (Keeping the team on track), 4 (Expecting quality), and 5 (Having relevant knowledge, skills, and abilities). After a week of performing with the updated agreements, the variables were measured again and the intervention concluded.

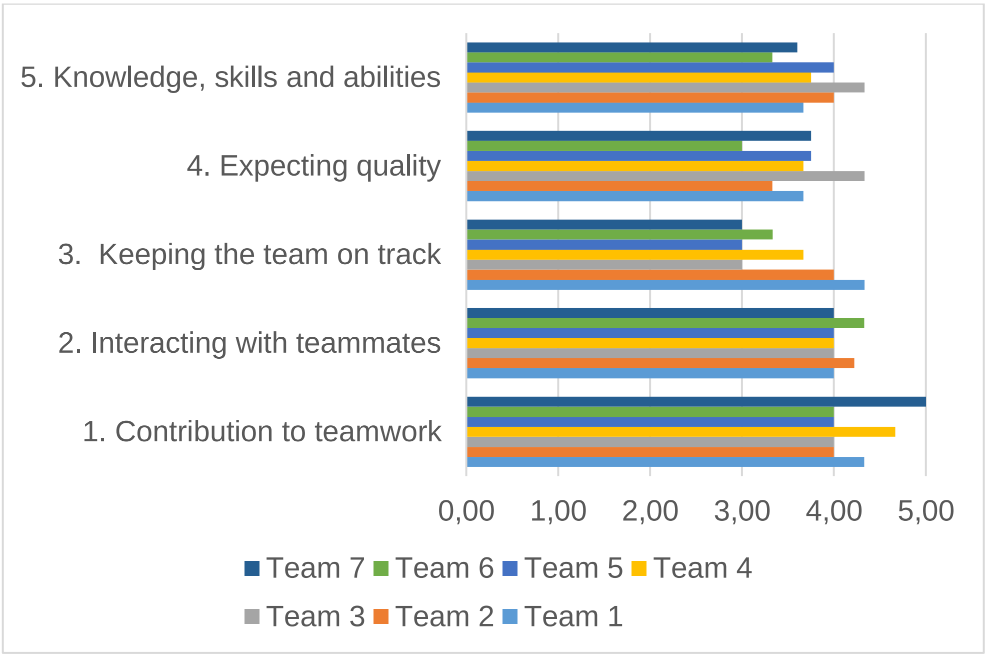


Figure 9. Results of the team member contribution assessment from quasi-experiment 1

3.2.2.2 ASEST0 effectiveness

A Shapiro-Wilk test on the difference in students' perceptions scores before and after the intervention showed p-values larger than .05, from which we could conclude that there is no evidence that the values would not correspond to a normal distribution. This test is recommended as the best choice for testing the normality of data (Razali & Wah, 2011).

To check the significance of increases we performed t-Student tests using 10000 bootstrap samples with a 95% level of confidence. As shown in Table 19, the tests revealed that the students' perceptions of team cohesion, team performance, and team learning in the intervened group significantly increased compared to the perceptions of the students in the control group. In addition, Glass's ∆ values showed large effect sizes (3.92 for team cohesion, 1.61 for team performance, and 4.84 for team learning). These results suggest that the increase can be attributed to the treatment, although some limitations should be considered, as we will discuss in section 3.4.

Table 19. T- tests results from quasi-experiment 1

	Experimental group				Control group				t	df	Sig. (2-tailed)
	Mean Before	Mean After	Diff.	Std. Deviation	Mean before	Mean After	Diff.	Std. Deviation			
Team Cohesion	55.45	76.47	21.06	2.46	56.88	57.80	0.92	4.76	8.94	10	.00
Team Performance	16.75	21.17	4.42	0.72	16.97	17.30	0.33	2.40	4.08	10	.00
Team Learning	22.39	29.94	7.55	1.36	19.71	20.02	0.31	2.05	10.14	10	.00

3.2.3 Quasi-experiment 2

The second quasi-experiment was performed with a single group of software engineering students from the University of Holguin in the period February-March 2017. They all were also developers at the software company "Datys" in Holguin, Cuba. Although this study had a weaker design (not including a control group) it allowed piloting the application of ASEST0 in a professional environment. In this way, it was possible to investigate if the framework were effective for teams composed of novice engineers working in a real industry setting.

First of all, the purpose of the study and the ASEST0 framework was presented to the company CEO and team leaders in January 2017. It was agreed to design a course including the intervention (similar to quasi-experiment 1) and topics on project management with a focus on Scrum. After presenting this proposal and making some adjustments regarding the topics as requested, the CEO and team leaders agreed with three teams to participate as part of their job.

The company was starting to move from RUP methodology to the agile approach. Table 20 describes the course schedule and the week when each step of the intervention was done. As the students were already working together in these teams for months, the cohesion of the teams before using ASEST0 was used as a baseline. Thus, the variables for both groups were measured during weeks 1 and 8.

Table 20. Course schedule and intervention in quasi-experiment 2

Week	Teaching and learning activities	Step of ASEST0
1	Workshop. (Agile management methods and teamwork) Team project	1,2, variables measurement
2	Workshop. (Communication and conflict resolution in agile teams) Team project	3
3	Workshop. (Self-regulated agile teams) Team project	4,5
4	Workshop. (Agile requirement engineering) Team project	
5	Workshop. (Planning agile projects) Team project	6
6	Laboratory. (Tools for agile project management) Team project	
7	Laboratory. (Tools for agile project management) Team project	7, 8
8	Team project (Capstone project presentation)	variables measurement

The teams were composed of five or six members. In total 16 software engineers participated, 11 were male; 5 were female. Their age levels ranged from 22 to 28 years old. They had between two and five years of experience in software development after graduation and one year on average experience in working as teams. The learning activities in the course were related to the projects they were doing at the company in order not to interfere in the organization of the teams and projects milestones. The activities done are described next.

3.2.3.1 Intervention

Phase 1

During the first workshop steps 1 and 2 were done. The researcher acted as the main teacher while other teachers assisted. The teachers explained the course design and the activities the students would be expected to do. In coordination with team leaders, the teachers agreed with the teams about the scope of the capstone projects.

Team members had been already assigned roles in these projects. The study did not include product owners as they were not available due to other assignments of the company. The team composition and project scopes are shown in Table 21.

Table 21. Team composition and project scopes in quasi-experiment 2

Team	Project scope
Team 1, 5 members 1 Scrum master 4 Developers	Traffic control system for the national police. Tickets processing, tracking, and notification of actions to take
Team 2, 6 members 1 Scrum master 5 Developers	Management Control Panel for Ministry of Justice. Statistics reports dashboard
Team 3, 5 members 1 Scrum master 4 Developers	Management System for Ministry of Justice. Legal accusations tracking and status reports

Next, the teachers presented an overview of methods for agile software project management (Kanban pool system, burndown charts, retrospective, on-site customer, daily stand-up, release planning, user stories, taskboard) and introduced students to debate on how to collaborate in teams to succeed in agile projects. Then, students were asked to answer the TKT questionnaire (communication and conflict resolution questions). They had 30 minutes to answer and then they were asked to check their responses to the correct answers and discuss the results in teams.

The students discussed their responses identifying what aspects of the questionnaire the team members more frequently answered wrong/right, to what teamwork issue (communication or conflict resolution) these wrong answers were mostly related, and how these aspects could impact agile teamwork. In discussing they were expected to consider the principles and values promoted in the Agile Manifesto (e.g. customer collaboration, responding to changes) (K. Beck et al., 2001) and the values that Scrum promotes for team members (i.e. commitment, respect, openness, focus and courage) (Schwaber & Sutherland, 2020).

The results of this assessment are included in Fig 10. The percentages of right answers per question scored between 40 and 60 % for both, communication and conflict resolution, and just a few questions scored above 80 %. These results showed that students needed better preparation on team working skills.

During week 2, step 3 - training on communication and conflict resolution skills - was done. Firstly, the teachers reminded students of what they learned from the TKT questionnaire answered in the previous workshop (step 2) by asking them to share with their teammates what they already knew about this topic, what they expected from their teammates during the exercise and how they could contribute to the team.

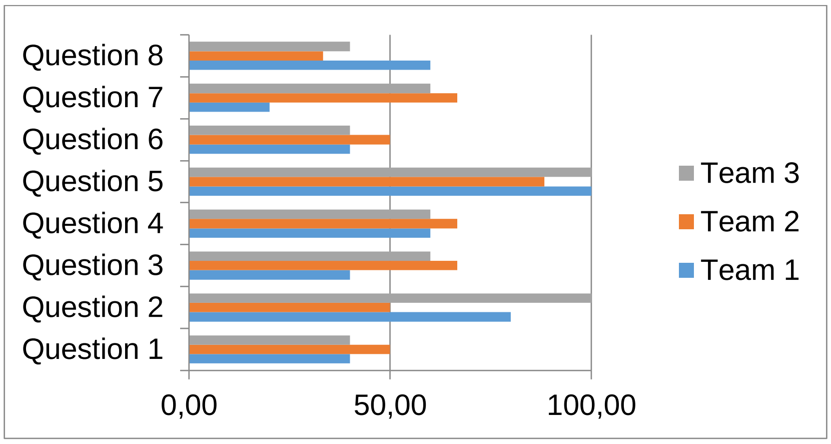
The students individually identified their Belbin’s roles, they were provided with information on these roles and they were asked to discuss in teams which roles were present in their teams, which roles were lacking, and how the presence/ absence of these roles could affect the teamwork. Table 22 includes an abstract of the roles that were present in each team and the main conclusions the teams shared.

Next, the teachers addressed aspects of communication and conflict resolution with an emphasis on agile software teams. The teachers explained some negative consequences of conflicts, types of conflicts, some positive outcomes of a conflicting situation, and how conflicts can be resolved cooperatively.

They also explained characteristics of effective communication, nonverbal and empathic communication and barriers that can affect it, channels of communication, best practices in agile development, and benefits of effective communication in agile teams. To prepare students for better communication along with role-playing gaming, the exercises in Table 6 to gain an understanding of nonverbal messages and empathic listening were done.

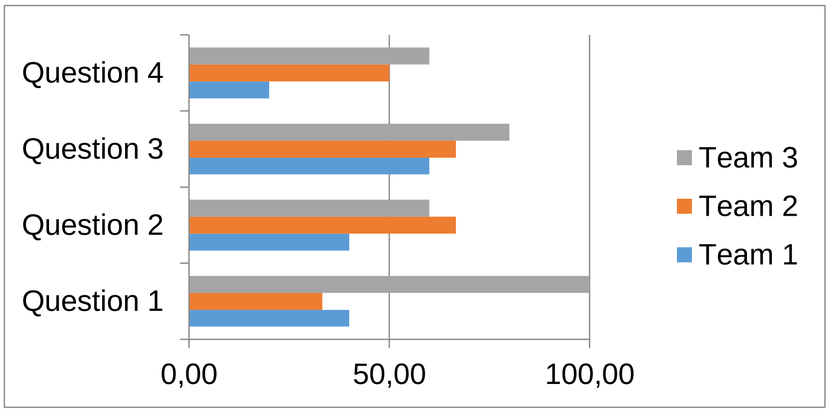
The teams were then provided with the description of conflicting situations in two rounds. At the end of each round, the teams shared their solutions. In some conflicting situations, the emphasis was on interpersonal conflicts, while in others the students had to solve conflicting requirements. In doing so, students assumed characters corresponding to their team roles (Scrum master and developers) and followed the activities described in Table 9. They were also asked to consider the situation from their Belbin’s role viewpoint following the criteria included in Table 8). Table 23 shows a sample of the outcomes from this activity.

Conflict resolution questions



Q1. When there is a disagreement or difference of opinion in your team, it is generally best to…
Q2. When dealing with a team member, who is not doing his/her fair share of the work, it is best to…
Q3. When you and another team member are having trouble communicating, which is the worst thing for you to do?
Q4. You have gotten quite angry in a team meeting. Which of the following is the least productive thing you could do?
Q5 In order to increase the chances of everyone doing their fair share of work, a team ought to…
Q6. Effective discussions of team business are often made difficult by people who are argumentative or dominating or disorganized. No matter what their problem, to get the meeting moving forward you need to...
Q7. If a member of your team is hostile or critical it is generally useful to…
Q8. Two members of your team have a genuine disagreement (not just miscommunication or personality conflict). Which of the following would be most likely to lead to a resolution?

Communication questions



Q1. When you are listening to other people offering their ideas, it is useful to…
Q2. When receiving feedback from your team members, it is generally useful to…
Q3. When expressing an idea or presenting some information, it is best to…
Q4. If a team member is expressing an opinion different from your own, it is generally helpful to…

Figure 10. TKT questionnaire results from quasi-experiment 2

Table 22. Belbin roles and sample reflections from quasi-experiment 2

Belbin role	Reflections
2 Teamworkers 1 Specialist 1 Monitor evaluator 1 Resource investigator	Our team may lose focus on action as there are no implementers, shapers, or completer finishers
1 Coordinador 1 Monitor evaluator 1 Shaper 1 Teamworker 1 Implementer 1 Specialist	Our team is well balanced in action and thinking roles. We don’t have plants, resource investigators, or completer finishers. Thus, generating ideas, searching for resources outside the team, or looking for errors are aspects that our team may need to reinforce
2 Plants 2 Implementer 1 Shaper	Our team is more focused on thinking and action and lacks social roles. We should be more careful coordinating tasks

Table 23. Sample of conflicts resolution activities outcomes from quasi-experiment 2

Dilemmas or challenges	Solutions
How to make all team members buy into a solution?	Establish a decision-making process, listen to each and everybody’s opinions, accept others point of view, listen to others feedback, make sessions for technical debate opportunely, bring external specialists to share novelties with the team, bring an external specialist to act as a mediator if the team is not able to agree on a technical discussion, recognize original ideas, identify clear objectives
How to keep healthy responsibility boundaries?	Identify overloaded members or in trouble to get their work done and support them, clear distribution of tasks and responsibilities according to the roles, respect, and open communication, get training to improve skills according to roles, effective coordination of tasks
How to enhance team members’ collaboration?	Analysis of often impediments during sprint retrospective reviews, update team members on collaborative practices, communicate openly and opportunely, foster trust and transparency, foment good relationships and nice environment, recognize personal achievements and celebrate team achievements, encourage honesty and respect

At the end of the training, the students were asked to debate how their expectations of the training were met and what was the most important lesson learned. In general, the students appreciated the training and refer to the activities followed to solve the conflicting situations as useful for the teams to solve conflicts in the future. Table 24 includes a sample of their comments.

Table 24. Sample of lessons learned from the training in quasi-experiment 2

Team	Reflections
Team 1	In the past, conflicting situations because of technical disagreements have also occurred in our team. How we review the situation described made us reflect on how we should address this kind of situation in our team. We have seen that often we ended avoiding deeper discussions and taking the easiest way to earn time. Although we have quickly solved the problem, this may not have been the best solution. We see now that some members have been feeling frustrated or worried since we have not properly discussed these matters before
Team 2	The most important lesson for us is related to role responsibilities'. In discussing the situation we have seen that conflicting situations we are currently facing in our team are related to this matter. This causes that sometimes communication does not flow as it should; maybe because selected channels maybe not be the most appropriated
Team 3	Communication is the key to success in resolving conflicting situations and avoiding them. Even when all team members know how to do great work from their roles, if they do not communicate well, the teamwork will be affected to some degree

Phase 2:

During workshop 3 (step 4) the students assessed the team functioning by using (Powers et al., 2002) questionnaire. These assessments (included in Fig 11) helped them as starting point to agree on policies to regulate communication and conflicting behaviors (step 5). The teams' responses were above 3 points on average, although just question 3 (related to expressing an idea or presenting some information) was closer to the maximum.

To set the team rules agreement, the activities described in Table 10 were done. Table 25 shows a sample of the agreed rules. After two weeks of performing on agreed rules, step 6 was done. The teams were asked to report on the rule agreements application. In general, all the teams indicated that the agreements were well accepted by all members. They reported however that some rules were not applied due to a lack of opportunity for it. For instance, team 3 indicated this was the case for the rule "we celebrate the diversity of opinions but outside the room, we present one aligned position" and team 1 experienced the same with "invite specialists to share ideas with our team". The teams expressed they all regularly brought the agreements to daily meetings and found this practice very helpful.

Table 25. Sample of agreed team rules from quasi-experiment 2

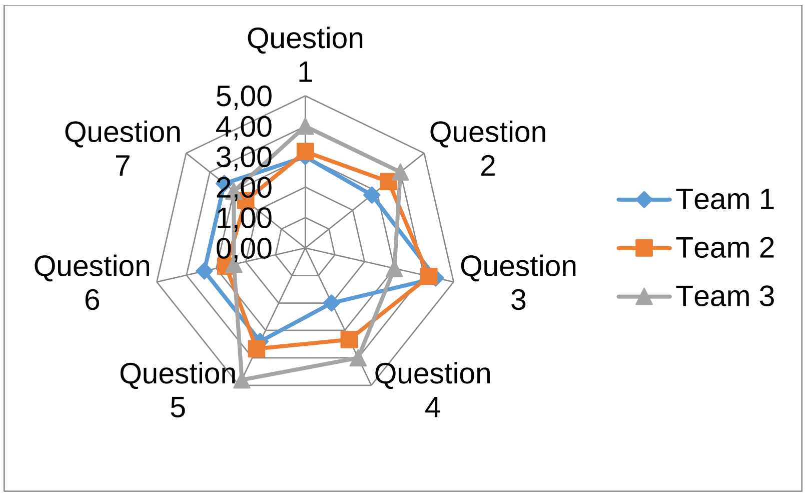
Team	Rules agreement
Team 1	Inform problems immediately Stick to team decisions Invite specialists to share ideas with our team Don't waste meeting time discussing non-relevant information for the team or nonessential details, instead, focus on meaningful work (update*) Set clear goals for each iteration (update) Keep everyone informed on events happening within the team (update)
Team 2	Share information opportunely When we debate, everybody shares their opinion Encourage introverted team members to share their opinion Pay attention to other people's updates (update) Adhere to engineering practices defined by the company and discuss infractions (update)
Team 3	Share opinions openly and honesty We celebrate the diversity of opinions but outside the room, we present one aligned position Listen first then judge, always to help! Establish consistent meetings cadence (same time, same place) (update) Updates in meetings should be short and to the point (update) Keep the team informed on interruptions (update)

*Update: Included in step 7

Phase 3:

During workshop 4, steps 7 and 8 were done. The students self- and peer-evaluated the team member contributions in five areas using a scale from 1 to 5. Fig 12 includes the results (averages per area) of this assessment.

All areas scored high. However, the students were asked to propose at least one rule to further improve areas 2 and 3, as these were the areas that obtained the lowest averages. After a week of performing with the updated agreements, the variables were again measured and the intervention concluded.



Q1. As a team we find it difficult to accept criticism openly and non-defensively. (reverse scored)
Q2. When conflict arises in the team, it is likely to be a battle or, at best, a waste of time. (reverse scored)
Q3. My team encourages differing opinions to be expressed.
Q4. When arguments break out, my team members are able to step back, calm down, and work out our differences.
Q5. My team members criticize ideas, not each other.
Q6. My team may agree on a solution but not every member "buys into" that solution. (reverse scored)
Q7. My team ignores conflicts among team members. (reverse scored)

Figure 11.Team functioning assessment results from quasi-experiment 2

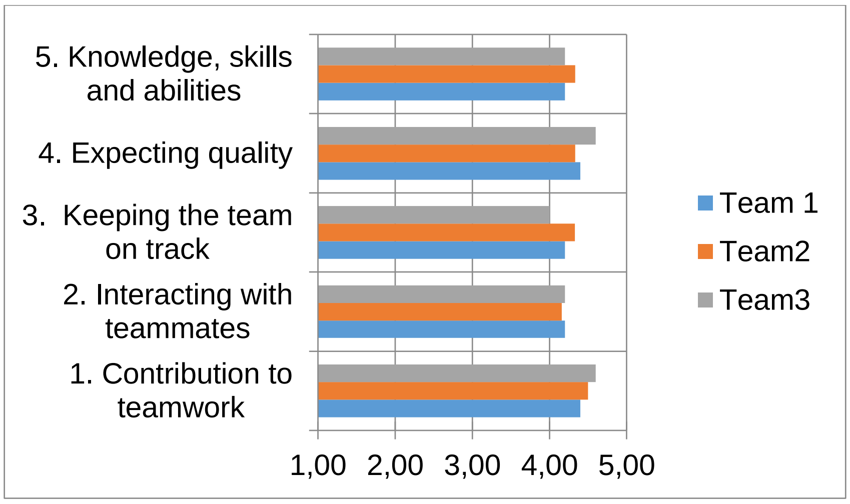


Figure 12. Team member contribution assessment results from quasi-experiment 2

3.2.3.2 ASEST0 effectiveness

A Shapiro-Wilk test on the difference in students' perceptions scores before and after the intervention showed p-values larger than .05, from which we could conclude that there is no evidence that the values would not correspond to a normal distribution. To check the significance of increases t-Student tests were performed using 5000 bootstrap samples with a 95% level of confidence.

As shown in Table 26, the tests show that the students' perceptions of team cohesion, team performance, and team learning significantly increased. In addition, Glass's ∆ values show large effect sizes (6.94 for team cohesion, 5.86 for team performance, and 6.91 for team learning). These results suggest that ASEST0 may have caused such improvements, although some important limitations of this study should be considered, as we will discuss in section 3.4.

Table 26.T- tests results from quasi-experiment 2

	Mean before	Mean after	Difference	Std. Deviation	t	df	Sig. (2-tailed)
Team Cohesion	58.94	74.50	15.56	0.95	28.43	2	.00
Team Performance	13.06	22.56	9.50	1.59	10.35	2	.00
Team Learning	21.44	30.83	9.39	1.34	12.17	2	.00

3.2.4 Students' perceptions on ASEST0

To get a better understanding of the students' perceptions, an open survey with a random sample of respondents was conducted. In total 30 students participated, 10 of them from the company. The question was "What is your opinion on the framework applied in the course(s)?". Appendix 6 shows translations of their responses as the students wrote them down.

Through summative content analysis, four major topics were identified from the students' responses. This analysis was computer-assisted by using Nvivo 12 software. The original Spanish responses were used. Firstly word frequency was analyzed, resulting in the following list of most frequently used words: team, rules, roles, game, project, conflict, experience, real, training, and contributions. Occurrences of these terms were counted including each alternative term (e.g. contribution, contributing contributed). Grouping these words, the identified topics resulted to be: the team member contributions assessment (contribution), the rules agreements (rules, conflict), the solving of a real problem (real, project, experience), and the team working skills training (roles, game, conflict).

Among the comments on the team member contributions assessment, students recognized it as important to "*put loafers in evidence*". One student pointed out that “*sometimes evaluations were unfair*”, proposing “*to do the assessment several times to overcome this kind of situation*”. They also valued the member contribution assessment to agreements' success.

In general, students valued rules agreements for “*improving teamwork*”, “*display more respect and understanding*” and “*prevent conflicts and work more united*”. They demanded more “*guidance to identify proper rules for the team*”, they asked for a way to check “*which rules are helping more*” or “*accomplishment of the rules to know how we are progressing*”, and they proposed “*the rules should focus more on project tasks*”.

Solving a problem that solved the real needs of the environment was well appreciated by the students. This was stated by the students in higher education (not in the company). Students valued the training on team working skills. They felt especially motivated about the role-playing gaming strategy. Several students suggested the training should have more sessions.

There were also some general opinions. On the positive side, the most often mentioned opinions were that they liked the experience, they would like to repeat it and they found the framework important for their development as software engineers as they learned how to better collaborate and contribute to the teamwork. As more negative comments students indicated they dislike filling in so many questionnaires, although they could also recognize the importance of the information gotten from them.

All in all, students appreciated the framework and gave some valuable feedback to improve it. Their major request was the need for more guidance to make the rule agreements and track the effectiveness of the rules.

3.2.5 Discussion and lessons learned

In this section, the results of quasi-experiments 1 and 2 are discussed along with the most important lessons learned from these studies. The discussion focuses on three aspects: team working skills preparation, team rules agreements, and team member contributions assessment. Lessons learned are described for the three phases (see Table 12).

In these studies, students of both graduate and undergraduate levels participated, though the individual diagnosis on team working skills (step 2) showed poor results in both experimental groups (most right answers scoring between 30-60 %). It should be considered however that the graduate students had on average less than five years of experience after graduation. In addition, most of them were graduated from the same program at the University of Holguin. These findings support the importance of the training (step 3) on these skills, which students appreciated, as shown by the survey.

The experimental teams in quasi-experiment 1 were able to reach agreements that addressed a broad variety of teamwork issues. The agreed rules were not just related to communication and conflict resolution, but they also included other teamwork issues like decision making. The agreements of the experimental teams in quasi-experiment 2 included fewer rules and they were more oriented to the engineering process.

Team member contributions assessment was shown to be appreciated by surveyed students in both studies. For students in quasi-experiment 1 three areas of the CATME-B questionnaire scored low (teams averages below 3 points on a scale of 0-5). These areas were “keeping the team on track”, “expecting quality” and “having relevant knowledge, skills, and abilities”. For students in quasi-experiment 2 all areas of the CATME-B questionnaire scored high (teams averages above 4 points on a scale of 0-5), nevertheless there was room for improvements. Therefore students focused the agreement updates on areas “interacting with teammates” and “keeping the team on track” that scored the lowest.

Table 27 presents a set of lessons learned from these studies to be considered to further improve ASEST0. They are grouped by the three phases of the framework.

Table 27. Lessons learned from quasi-experiments 1 and 2

ASEST0 phase	Lessons learned
Preparation	Discuss with real clients the project objectives, technologies needed, expectations, and roles beforehand. There is a need for strong commitment on their part. Discuss some case studies of similar projects before starting role-playing of simulated situations
Implementation	Include rules applicable to other teamwork functioning facets (e.g. decision making). Allow teams to learn from each other. Include rules on agile routines. If possible, engage real clients on contributing to team rules agreements. Ask teams to register the real conflict situations they may face during the project development to identify if there is a need for further skills development. If needed, come back to role-playing

Adjustment	Even when the team member contribution assessment does not show serious problems, including new rules in some areas may help to boost the teamwork. Discuss agreement updates not just regarding member contributions but any other aspect the team could be needing to regulate

3.3 Comparing ASEST0 to existing learning- teaching frameworks

In this section, we return to the literature review discussed in chapter 2 to compare the ASEST0 framework to existing related teamwork teaching-learning frameworks. We looked at validated proposals for SEE aimed at improving factors related to inputs, mediators, or outcomes of the teamwork process as their main objective. We compared frameworks in terms of trends (the basis of our proposal) and teamwork factors addressed, along with the learning scenario characteristics and main activities included. Furthermore, we looked at how our findings relate to the existing literature by comparing our results with those arrived at in previous studies in terms of student perceptions - both quantitative and qualitative. The comparison is presented in Appendix 7 and further described as follows.

A total of four teaching-learning frameworks that meet the mentioned comparison criteria were found. While these proposals focus on one or two trends, ASEST0 considers almost all of them, except for global and virtual teams. Regarding the factors, team cohesion is addressed in only one framework (Chung-yang Chen et al., 2014), where cohesion is seen as one facet of team quality. Although the proposal in (Silvestre et al., 2016) claims to design cohesive teams, these authors do not provide any evidence of cohesion improvements or how the components of their proposal relates to team cohesion. Team rules antecedent has not previously been studied at all. No studies have addressed team learning or other performance behaviors. The learning scenario characteristics are very different for all the proposals. Besides ASEST0, only the work of (Alsaedi, Toups, and Cook 2016) includes activities to prepare students for team working skills, specifically concerning communication and coordination skills. No proposal includes identification and the establishment of team rules. Only the work of (Garcia and Pacheco 2014) includes peer evaluation for contributions, though the criteria used are different.

In addition, we looked at how our findings relate to previous studies by comparing our results with those of other research work in terms of student perceptions - both quantitative and qualitative. Two studies present proposals that refer to analyzing cohesion. The study in (Chung-yang Chen et al., 2014) evaluates the effects of the Meeting-flow (MF) approach on team quality. The authors found that while the MF significantly enhanced the other five facets, it had less of an influence on team cohesion. However, this study uses a different conceptualization of cohesion, which limits a full comparison with our results.

The study in (Silvestre, Ochoa, and Marques 2016) proposes a heuristic technique to design software student teams. Although the authors claim the heuristic tries to maximize the initial level of team cohesion, the evaluation suggests their proposal is only effective concerning internal communication and coordination. The studies based on the quantitative evaluation of team performance only focus on performance outcomes such as team assignments, team presentations, and project outcomes. Some students involved in these studies express opinions that coincide with our qualitative findings. The students claim they enjoy working in real-world projects (Lago, Muccini, and Babar 2012), that team member contribution assessment helped reduce the student 'free-rider' and 'Wyatt Earp' syndromes (C. Chen and Teng 2011), and gaming helped overcome difficulties related to teamwork issues (Alsaedi, Toups, and Cook 2016).

3.4 Discussion on validity

The main limitations of quasi-experiment 1 and 2 are the small sample size along with a not random sample. In addition, quasi-experiment 2 did not include a control group. To achieve the objectives of the framework, it was necessary to select groups of students with enough programming skills to develop the capstone projects. This condition limited the number of groups available to be studied. Some educational practices of the target program, like the learning activities design, constrained the study as well. While teamwork activities might have enhanced the learning design of these courses, it was not possible to fit the intervention in the schedule due to a restriction on the amount of teaching hours. The availability of teams in the target company to participate in the study limited the possibility of including a control group. While the company decided to start moving to agile methods with the teams included in quasi-experiment 2, the other teams in the company still used waterfall methodologies during the period of the study.

To deal with the small sample size, however, the statistical analyses included bootstrapping techniques. According to (Onwuegbuzie, 2003), population validity is problematic in nearly all educational studies as the majority of researchers are forced to select a sample from the accessible population, and random samples are difficult to obtain.

Allowing students to self-select teams could have caused some bias in quasi-experiment 1. Other factors that may influence team cohesion should have been considered in both studies. For instance, the personality of team members (Acuña et al., 2009, 2008; Karn et al., 2007) or tasks characteristics (Acuña et al., 2009, 2008). The fact that the intervention in quasi-experiment 2 included some learning activities on project management (as part of the course requested by the company administration) in addition to the steps of ASEST0, might also have influenced the performance of the teams.

The quasi-experiments are also limited by the locations in which the investigation took place. The local conditions and the socio-economic status and cultural background of the participants might have influenced the studies. As the researcher acted as the main teacher, researcher bias (e.g., personality traits or pre-conceived beliefs of the researcher) might have had some impact as well.

3.5 Conclusions

This chapter presented the first version of the framework, ASEST0. The proposal was developed based on four trends of teamwork in SEE: collaborative learning, games and gamification, agile methods, and real projects resolution and links with industry. Therefore, the ASEST0 framework combines team-based learning, project-problem-based learning, and role-playing game learning strategies in three phases and eight steps.

ASEST0 was validated utilizing a study with a group of students that received the intervention compared to a control group without intervention. It was observed that the student's perceptions of team cohesion, team performance, and team learning in the intervened group significantly increased compared to the perceptions of the students in the control group. In addition, ASEST0 was tested in a second quasi-experiment that involved three teams of undergraduate students with the ultimate goal of observing ASEST0 with teams performing in a company setting. Despite this second study having a weaker design than the first quasi-experiment, it also showed that levels of team cohesion, team learning, and team performance increased after the intervention. Therefore, although the limitations of these studies should be considered, they showed the ASEST0 framework to be effective. They also contributed to identifying difficulties with the approach.

To get a better understanding of the perceptions of the students who participated in these studies, a survey with a random sample of participants was conducted. The participants expressed appreciation for ASEST0 but also pointed to some areas that required improvement. Finally, a comparison of ASEST0 framework to existing related teamwork teaching frameworks showed the novelty of this proposal.

The studies described in this chapter concluded the first iteration of this doctoral research. The next chapter reports on the second iteration that aims to improve the proposed framework considering the difficulties identified from quasi-experiment 1 and 2 and antecedents of team cohesion. In doing so, we will come back to the research model (section 3.1.1) based on IMO that led to the construction of ASEST0. This research model included team rules establishment enriched by feedback on team member contributions as the input. However, antecedents of team cohesion (the mediator in our model) were not taken into account. The next chapter will report on the identification of such antecedents.

Chapter 4
Antecedents of team cohesion

The content of this chapter is based on the article published in the IEEE Transactions on Education Journal as: Tamayo Avila, D., Van Petegem, W., & Snoeck, M. (2021). Improving Teamwork in Agile Software Engineering Education: The ASEST+ Framework. Partial results are also published in the INTED2018 Proceedings as: Tamayo, D., Van Petegem, W., & Noda Hernández, M. Cruz Ochoa, Y. (2018). A correlational study on factors that influence the cohesion of software engineering students teams.

In previous chapters, the construction and validation of the first version of the framework (ASEST0) were discussed. The framework's fundamentals were established by identifying teamwork trends in SEE, for example, collaborative learning, games, and gamification, among others, as explained in Chapter 2. The ASEST0 was developed following an IMO (input-mediator-outcome) research model adapted to SEE. In ASEST0, team rules are the input, improved by self- and peer-assessments of team member contributions. Team cohesion is the mediator, and team learning and performance are the outputs. ASEST0, however, did not include the antecedents of team cohesion. In the present chapter, this shortcoming is addressed.

This chapter reports on two literature reviews and a correlational study that were conducted during the second iteration of this doctoral research. The following research questions are answered:

Research question 4: What are the cohesion antecedents for collocated teams in software engineering education?

Research question 5: What are the most relevant identified antecedents for agile collocated student teams?

Research question 6: What are approaches in agile software development (ASD) regarding the relevant antecedents to improve the proposed framework?

The chapter is structured as follows: Section 4.1 reports on the literature review study that identified antecedents of cohesion while focusing on reported studies for collocated teams in educational settings. For those identified antecedents, a correlational study was performed over a sample of agile student teams to recognize the most relevant ones for our population. This correlational study is reported in section 4.2. The second literature review is discussed in Section 4.3. It looked at how each of the relevant antecedents was addressed specifically in the context of ASD with the ultimate goal of improving ASEST0 on these aspects for agile education. Section 4.4 discusses on limitations of these studies and section 4.5 concludes the chapter.

4.1 Antecedents of team cohesion in the context of Software Engineering Education

A systematic literature review was conducted to identify the antecedents of team cohesion addressed in the context of SEE. The literature review was conducted to identify further directions for improvement of ASEST0 regarding team cohesion antecedents. It included three stages: collecting data, analyzing data, and compiling a list of antecedents.

4.1.1 Method

The systematic literature review was performed following the guidelines proposed by (Kitchenham et al., 2009). The steps in the literature review method are documented below.

4.1.1.1 Research question

The research question formulated is **Research question 4** of this doctoral dissertation: What are the cohesion antecedents for collocated teams in software engineering education?

4.1.1.2 Search process

The process followed to answer the research question is shown in Fig 13. The process started with a manual search of specific conference proceedings and journal papers to explore the research area and limit the scope. The proceedings and journals included in this search were the following: Journal of Systems and Software, Journal of Software: Evolution and Process, Proceedings IEEE Frontiers in Education, and Proceedings Conference on Software Engineering Education & Training. This search led to the identification of relevant papers, which were used as a validation list to ensure the reliability and relevancy of the later searches and to evaluate the search strings.

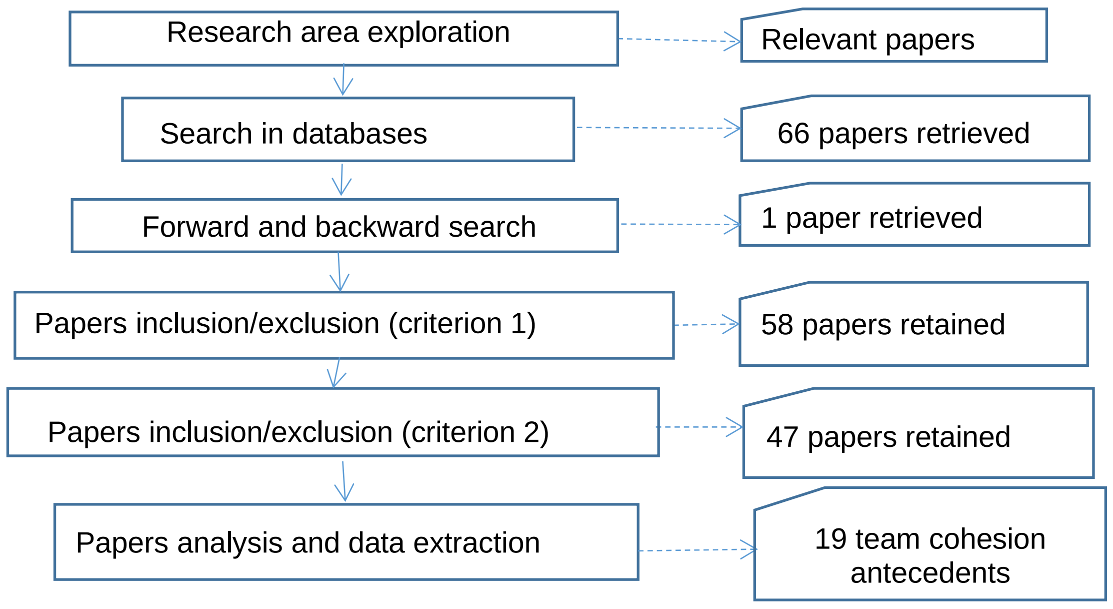

Research area exploration
Relevant papers
Search in databases
66 papers retrieved
Forward and backward search
1 paper retrieved
Papers inclusion/exclusion (criterion 1)
58 papers retained
Papers inclusion/exclusion (criterion 2)
47 papers retained
Papers analysis and data extraction
19 team cohesion antecedents

Figure 13. Literature review 2 research process and steps results

4.1.1.3 Inclusion and exclusion criteria

In the search, all articles on the Scopus and Web of Science databases published back to April 2017 were included. The following criteria were used to exclude papers: papers for which the full text was not available, full books of proceedings, and papers that had a scope not relevant to the perspective of this study. These studies are listed in Appendix 8. The papers not relevant for this study addressed team cohesion from a different conceptualization or field of application. Those papers with a different conceptualization studied cohesion concerning software code internal characteristics, as a property related to how these elements of a module belong together.

4.1.1.4 Data collection

For data collection, the search was based on the following query: TOPIC: (cohesion and team* and (“software engineering” or “software development”)). To assess the quality of the query, we checked that the studies we already knew to be relevant for our search ((Acuña et al., 2008) and (Karn et al., 2007)) appeared in the results in order to ensure that this search query was able to find these papers. Based on the exclusion criteria explained (criterion 1 in Fig 13), 58 papers were retained.

In a new iteration, 10 more papers were excluded. In these papers, the authors mentioned team cohesion (e.g to explain related work or further explain some related issue) but they did not investigate this construct (criterion 2 in Fig 13).

A deeper analysis of the rest of the articles retrieved was done. Two data extraction forms were used then to collect the information needed to address the review question. The next section discusses the information collected in these forms.

4.1.2 Results

Firstly, the main contribution of the studies and the cohesion-related findings were summarized (presented in Appendix 9). For those studies where primary data was provided to support their findings, a deeper analysis was done. The information related to these selected studies was collected in a second form (presented in Table 28).

*Table 28. Team cohesion antecedents**

Antecedent	Study	Studied in education	Team location
Software engineering methodologies	(Karn et al., 2007; Wellington et al., 2005a; Whitworth & Biddle, 2007) (Wellington et al., 2005b)(Whitworth, 2008)	Yes	Collocated
Member skill awareness	(X. Yang et al., 2015) (Xue et al., 2015)	No	Virtual
Shared governance	(X. Yang et al., 2015) (Xue et al., 2015)	No	Virtual
Personality	(Acuña et al., 2009, 2008; Karn et al., 2007)	Yes	Collocated
Conflicts	(Acuña et al., 2009, 2008; Gómez & Acuña, 2007; Rutz & Tanner, 2016)	Yes	Collocated
Temporal motifs	(Xuan et al., 2015)	No	Virtual
Proximity of team members	(Hoegl & Proserpio, 2004) (Jeanne M. Wilson et al., 2008)	No	Collocated/Virtual
Task interdependence	(Acuña et al., 2009, 2008)	Yes	Collocated

Task autonomy	(Acuña et al., 2009, 2008)	Yes	Collocated
Feedback	(Castro-hernández et al., 2015)	Yes	Virtual
Communication	(Castro-Hernandez et al., 2016; De Farias et al., 2012; A. C. Hernandez, 2017)	Yes	Virtual
Meeting-flow	(Chung-yang Chen et al., 2014)	Yes	Collocated
Collaboration	(Rutz & Tanner, 2016)	No	Virtual
Goal setting	(Rutz & Tanner, 2016)	No	Virtual
Shared understanding	(Rutz & Tanner, 2016)	No	Virtual
Diversity	(Chidambaram & Carte, 2005; Rutz & Tanner, 2016)	Yes	Virtual/ Collocated
Trust climate	(Rutz & Tanner, 2016)	No	Virtual
Leadership	(Rutz & Tanner, 2016)	No	Virtual
Team formation	(Shaikh et al., 2016)	Yes	Collocated

*The antecedents highlighted in grey were selected to be further studied

In total 19 antecedents of cohesion for software engineering teams were identified. While some of the factors might be relevant, they were not included because they pertain to virtual teams while this research focuses on collocated teams. In all, just eight of the total list of antecedents were found to have been studied for collocated teams in educational settings: software engineering methodologies, personality, conflicts, task interdependence, task autonomy, meeting-flow, diversity, and team formation.

The final list is constituted of these factors, except diversity and meeting-flow that were not included for the following reasons. The study of (Rutz & Tanner, 2016) on the diversity found that diversity was more significant for virtual teams. Meeting-flow was removed from the list because it does not refer to a specific construct but to a whole framework, including several concepts and approaches, not having enough evidence of the effectiveness of each of them by separated, therefore not included either. As a result, the following six antecedents were finally selected: software engineering methodology, team formation, personality, conflicts, task interdependence, and task autonomy.

In this dissertation, software engineering methodologies are seen as “a sequence of tasks that... [Can be controlled, measured, and improved], to produce the desired result… [In software development]” (Humphrey, 1987). Team formation refers to the methods used to form the teams (Decker, 1995). Personality is seen as "the relatively enduring styles of thinking, feeling and acting that characterize an individual" (P. T. J. Costa & McCrae, 1992). Conflicts are addressed from two perspectives, relationship, and task conflicts (Jehn, 1995). A relationship conflict exists in presence of interpersonal incompatibilities among team members, which typically lead to tension, animosity, and annoyance in the team. Task conflicts refer to disagreements among team members about the content of the tasks being performed, which includes differences in viewpoints, ideas, and opinions (Jehn, 1995).

Task interdependence refers to the situation when team members have "to share materials, information, or expertise to achieve the desired performance or output" (Gerben S van der Vegt et al., 2001). Finally, task autonomy is seen as the freedom the team members have to make decisions about objectives, work methods, delivery schedules, and distribution of work among team members (Acuña et al., 2009).

4.2 Relevant antecedents for agile collocated student teams

In this section, the relevance of the six identified antecedents for the target population is studied. The following sections explain the methodology and report the results.

4.2.1 Method

Through survey and correlational studies the **Research question 5** of this doctoral dissertation is answered: What are the most relevant identified antecedents for agile collocated student teams?

A survey addressing the selected six antecedents and team cohesion was presented to a sample of 61 software engineering students performing in 17 agile teams. They were asked to voluntarily participate. All the students filled it in the survey. Their answers were anonymous. The students were performing in the Informatics Engineering program and the Master program of Mathematics and Informatics for Management, both from the Faculty of Informatics and Mathematics at the University of Holguin. The criteria used to measure the variables and the questionnaires are presented in Table 29 and Table 30.

Table 29. Criteria to measure variables in the correlational studies

Variables	Instrument or criteria	Values
Software engineering methodology	Software engineering methodology used	Iconix, Scrum, XP
Team formation	Criteria of (Decker, 1995)	self- selection; random; by command
Personality	NEO Personality Inventory, Spanish version (P. T. J. Costa & McCrae, 1992)	openness to experience, conscientiousness, extraversion, agreeableness, neuroticism
Conflicts	Survey of (Jehn, 1995)	Likert scale
Task autonomy	Survey of (Molleman & Beukel, 2007)	Likert scale
Task interdependence	Survey of (Gerben S van der Vegt et al., 2001)	Likert scale

*Table 30. Surveys used to measure variables**

Conflicts (5 point scale 1 = "None" 5 = "A lot")
How much friction is there among members in your team**?
How much are personality conflicts evident in your team?
How much tension is there among members in your team?
How much emotional conflict is there among members in your team?
How often do people in your team disagree about opinions regarding the work being done?
How frequently are there conflicts about ideas in your team?
How much conflict about the work you do is there in your team?
To what extent are there differences of opinion in your team?
Task autonomy (5 point scale 1 = “strongly disagree” 5 = “strongly agree”)
I can determine myself how to carry out my work
I can decide myself in which order to carry out my tasks
I can decide to take a short break from my work should I want to
I have discretion in performing my work
I have influence in planning my work
I have influence on the pace of my work
I can decide myself when to complete a task
I solve problems in my work myself
Task interdependence (5 point scale 1 = “strongly disagree” 5 = “strongly agree”)
I have to obtain information and advice from my colleagues in order to complete my work
I depend on my colleagues for the completion of my work
I have a one-person job; I rarely have to check or work with others***
I have to work closely with my colleagues to do my work properly
In order to complete their work, my colleagues have to obtain information and advice from me

*In addition to these surveys, team cohesion was measured as presented in Table 11
**The word “team” was used instead of the expression “work unit” in the original survey
***Reverse scored

4.2.2 Results

Using SPSS version 25, relationships between each antecedent and team cohesion were tested. No significant associations were found for the two non-numerical variables (software engineering methodology and team formation). The analyses of variance (one-way ANOVA) indicated $\eta^2 = 0.05$, F [(58, 2)=1.37, p= 0.26] for software engineering methodologies and $\eta^2 = 0.06$, [F(56,4)= 9.46, p= 0.44] for team formation.

Table 31 summarizes the data statistics, correlations, and the results of a regression analysis performed to test relationships for the other four antecedents and team cohesion. As can be seen, Pearson correlation confirms a significant correlation of team cohesion with personality (r= 0.44; p< 0.01), task interdependence (r= 0.34; p< 0.01), conflicts (r= -0.46; p< 0.01) and cohesion. There was no significant correlation between task autonomy and cohesion (of p = 0.02, n.s).

Multiple regression analysis was used to test if these aspects significantly predicted cohesion. This technique enables to determine a correlation between a criterion variable (cohesion in this case) and the best combination of several predictor variables (Fraenkel & Wallen, 2009). The coefficient of determination indicated that 40% of the variance in cohesion was explained by the found predictors (R^2= 0.40, F(4,56)=9.42, p< 0.01). It was found that personality significantly predicted cohesion (β = 0.38, p< 0.01), as did task interdependence (β= 0.23, p< 0.05) and conflicts (β = -0.36, p< 0.01). Task autonomy was not found a good predictor as p> 0.05, consequently, this variable could be removed from the regression model.

Table 31. Data statistics, correlations, and regression weights

	Mean	Std	Pearson correlation	Regression weight
			r	β
Personality	194.69	17.52	0.44**	0.38**
Task Interdependence	18.48	3.79	0.34**	0.23*
Task Autonomy	29.63	7.24	0.02	0.05
Conflicts	31.75	3.78	-0.46**	-0.36**

**p*<.05, ***p*<.01

As a result, personality, task interdependence, and conflicts were the most relevant antecedents to be considered to improve the ASEST framework.

4.3 Relevant antecedents in the context of Agile Software Development

A systematic literature review was conducted to identify further directions for improvement of the framework regarding team cohesion antecedents. Specifically, the three relevant antecedents found through the correlational study were investigated in the context of Agile Software Development (ASD). The methodology and the discussion of the findings are discussed below.

4.3.1 Method

As in previous literature reviews conducted as part of this doctoral research, this study was performed following the guidelines proposed by (Kitchenham et al., 2009). The steps in the literature review method are documented below.

4.3.1.1 Research question

The literature review answers **Research question 6** of this doctoral dissertation: What are approaches in agile software development (ASD) regarding the relevant antecedents to improve the proposed framework?

4.3.1.2 Search process

The process followed is shown in Fig 14. The process started by searching in the Scopus and Web of Science databases. To ensure the reliability and relevancy of the searches and to evaluate the search strings, the papers we already knew to be relevant were used as a validation list. These papers were selected from the literature review that identified the antecedents.

From the initial list of articles, some of them were removed in two moments. The excluded articles are listed in appendix 10. The criteria used are explained in the next subsection. An analysis over the remaining 57 papers allowed us to identify the most addressed issues regarding the antecedents which ultimately led to directions of improvement for the ASEST0 framework.

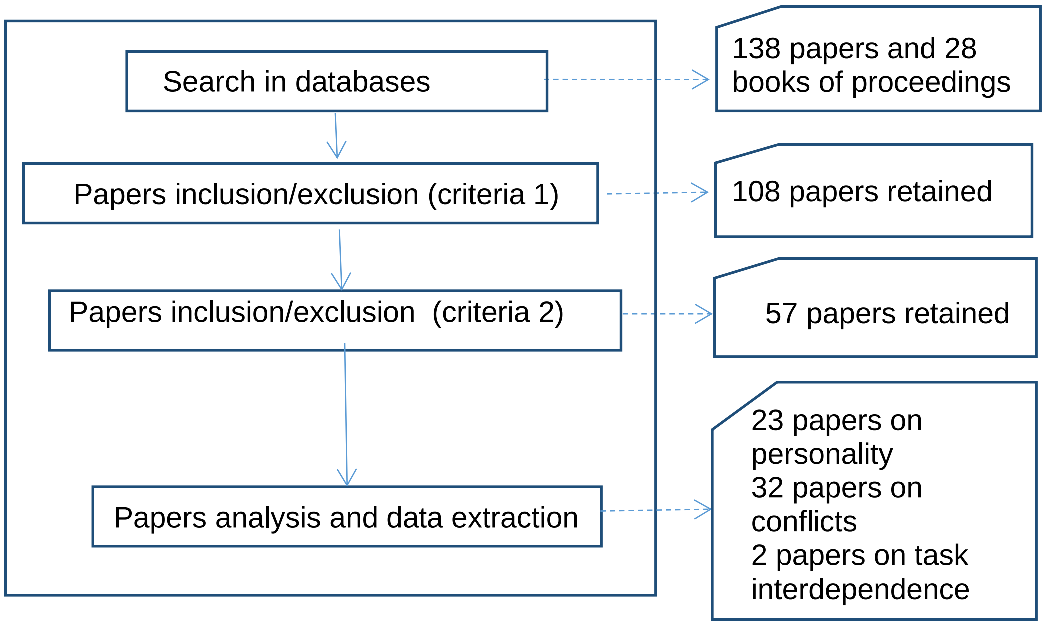

Search in databases
Papers inclusion/exclusion (criteria 1)
Papers inclusion/exclusion (criteria 2)
Papers analysis and data extraction
138 papers and 28 books of proceedings
108 papers retained
57 papers retained
23 papers on personality
32 papers on conflicts
2 papers on task interdependence

Figure 14. Literature review 3 research process and steps results

4.3.1.3 Inclusion and exclusion criteria

The following criteria were used to exclude papers: papers for which the full text was not available, full books of proceedings, and papers that had a scope not relevant to the perspective of this study (criteria 1 in Fig 14). The papers not relevant for this study were those that did not address the antecedents. Several papers were found to tackle conflicts or personalities from a too narrow or general scope (criteria 2 in Fig 14). In that case, the authors just mentioned them as related somehow to the main topic.

For example, the work reported in (Julian M Bass, 2013) that studies the role of product owners mentions that product owners should be prepared to deal with conflict resolution (e.g. to solve customer requirements conflicts), however, they did not describe how to tackle this matter. An example of an excluded paper because a too general scope is the work in (Crawford et al., 2014). While these authors discuss the role of conflicts management in agile development, they do not specify any particular approach to handle conflicts and they do not state any scientific conclusion. The excluded studies are listed in Appendix 10.

4.3.1.4 Data collection

The search string was (“software development” or “software engineering”) AND agile AND (“task interdependence” or personality or conflict). To assess the quality of the query, the researcher verified that studies they knew from the previous literature review on team cohesion antecedents to meet the search criteria for the current review (i.e., (Acuña et al., 2008), (Acuña et al., 2009)) to be present in the results. The query resulted in a collection of 138 papers published before June 2017. A total of 57 papers were retained.

4.3.2 Results

The scope of the studies related to personality, conflicts and task interdependence are presented in Appendix 11. Personality has been studied concerning its influence on job satisfaction and product quality (Acuña et al., 2008), (Acuña et al., 2009), productivity (K. S. Choi et al., 2008) (Papatheocharous et al., 2014), team performance (Mazni Omar & Syed-Abdullah, 2010), communication (S. A. Licorish & Macdonell, 2015), group development (Gren et al., 2017), work practices and beliefs (E. K. Smith et al., 2016), value system (Fagerholm & Pagels, 2014), preference for agile methods (Dave Bishop & Deokar, 2014), the outcome of Scrum teams (Branco et al., 2012), and advice networks (Keith et al., 2017).

Personality traits, characteristics, and temperaments have been considered regarding team structures (M. Yilmaz et al., 2017), learning styles (Layman & Williams, 2008), to predict persons to be suited for agile methodologies (Bhannarai & Doungsaard, 2017), to effectively allocate and rotate developers in pair (Sfetsos & Stamelos, 2011) (Venkatesan & Sankar, 2014) and for group development and maturity (Gren et al., 2017).

Personality was most often found to be addressed concerning team roles (Young et al., 2005) (Mazni et al., 2015) (Baumgart et al., 2015), team formation (S. Licorish et al., 2009; M. Omar & Khasasi, 2017; Papatheocharous et al., 2014) and pair programming practice (K. S. Choi et al., 2008; Sfetsos & Stamelos, 2011) (Venkatesan & Sankar, 2014).To improve ASEST0 framework, approaches regarding personality to allocate roles during team formation will be addressed. Personalities in regard to pair programming will be left out considering that this agile practice is more used in education to teach programming.

Two papers were found for aspects influencing conflicts in agile software development: sociocultural differences (Ozawa & Zhang, 2013) and emotional contagion (Alhubaishy & Benedicenti, 2017). Conflicting priorities were identified as obstacles to decision-making in agile software development (Drury et al., 2012). Conflicts among team members were found to influence the development of real projects in an agile academic environment (Monica Villavicencio et al., 2017). Task conflict was found to impact the results of the collaborative software development process and outcomes (Domino et al., 2004) (Crawford et al., 2014). Some studies addressed project management conflicts (Neill et al., 2017), specifically conflicts on funding processes (Cao et al., 2013), conflicts between long-term quality and short-term progress (Moe, 2013), conflicts between project and departmental tasks, conflicting and unclear projects prioritization (Salameh & Alnaji, 2014), planning, monitor and control (Pechau, 2012), results and scheduling conflicts (Rodin et al., 2011), communication (Pechau, 2012), information sharing (Pechau, 2011), and conflicts between organizational control and flexibility (Hannay & Benestad, 2010).

Conflicts have been studied concerning cultural conflicts (Ramesh et al., 2017) (Ozawa & Zhang, 2013), conflicting demands between alignment and adaptability (Ramesh et al., 2012), roles (Julian M Bass, 2015), Information Systems (IS) control alignment (Cram et al., 2016), Concurrent Versions System (CVS) (O'Reilly, C; Morrow, P; Bustard, 2003), software development methods (Wendorff, 2002), conflicts between agile and plan-driven methods (Brinker & Marcolina, 2016) (McMahon, 2004), interpersonal conflicts (Antonio Martin et al., 2013), social conflicts (Butgereit, 2017) and conflicts in trustworthy development in dynamic environments (Cao, 2012).

Conflicts between stakeholders (Abdelnour-Nocera & Sharp, 2012; R. Vijay Anand & Dinakaran, 2017; Drury et al., 2012; Khan et al., 2014) and requirement conflicts (Busetta, 2017; Chetankumar & Ramachandran, 2009b, 2008; Sachdeva & Chung, 2017) were the most addressed issues. Approaches regarding both requirements and stakeholders' conflicts will be included to improve ASEST0.

While personality and conflicts aspects were found to be widely addressed in agile software development, for task interdependence the same cannot be affirmed. Just two papers were found regarding this antecedent. One study focuses on the relationships of task interdependence with teamwork quality and project performance (Kuthyola et al., 2017). The other tackles the potential effects of trust on interdependence (Barbosa et al., 2017). As a result, no strategies addressing task interdependence for agile teams were found.

In conclusion, two major aspects for improving ASEST0 were found: the relevance of personality traits for team roles allocation and the resolution of conflicts regarding requirements and stakeholders.

4.4 Discussion on validity

The literature studies performed are limited by the possibility of missing relevant papers. Although the search process offers some guarantee for completeness considering the use of two large databases and a sufficient and broad set of keywords, it cannot be affirmed that relevant studies could have been left out. This might have occurred due to the inaccessibility of some papers and restrictions in the period in which the literature reviews were conducted.

Researcher bias regarding the selection and evaluation of the studies is another important limitation to consider. While some of the articles and results of the analysis were discussed with the supervisor and co-supervisors of this doctoral research, the majority of the papers were solely analyzed by the Ph.D. researcher. Therefore, the unconscious bias of the researcher could have led to errors and subjective interpretations during the data extraction.

Next, while according to some authors, the correlational study sample size used here is appropriate, for others, this could be considered a limitation. It has been suggested that a sample size of at least 200 is required (Kline, 2000). However, (Fraenkel & Wallen, 2009) states that the minimum acceptable sample size for a correlational study for most research is 30. In this case, the study is somewhere in between.

4.5 Conclusions

This chapter reports on improvements to the ASEST0 framework to derive the new ASEST+ framework. Therefore, antecedents not yet considered in ASEST0 were identified. Research questions 4, 5, and 6 were answered through two literature reviews and a correlational study. A total of 19 antecedents of team cohesion reported in the literature were identified, 6 of them for collocated teams in educational settings. For these six antecedents, a correlational study was performed over a sample of agile student teams to recognize the most relevant ones for our population.

The correlational study showed conflict, task interdependence, and personality to be the most important ones. In a second literature review, it was identified how these antecedents have been addressed in the context of ASD. As a result, two major aspects for improving ASEST0 were found: the relevance of personality traits for team roles and the resolution of conflicts regarding requirements and stakeholders. The next chapter will present how these aspects are included in the new version of the framework: ASEST+.

Chapter 5
The ASEST+ framework

The contents of this chapter are based on the article published in the IEEE Transactions on Education Journal as: Tamayo Avila, D., Van Petegem, W., & Snoeck, M. (2021). Improving Teamwork in Agile Software Engineering Education: The ASEST+ Framework.

The previous chapter reported on the results of the second iteration of this doctoral research. Two literature reviews and a correlational study were conducted to identify further directions of improvements for ASEST0. Antecedent factors not considered in ASEST0 were identified. After a literature review on cohesion antecedents in education, a correlational study identified the most relevant ones for agile teams. In a follow-up literature review, we searched for strategies regarding these antecedents in agile software development. Two major aspects for improving ASEST0 were found: the relevance of personality traits for team roles and the resolution of conflicts regarding requirements and stakeholders.

This chapter continues reporting on the results of the second iteration. Specifically, it addresses the improvements to the ASEST0 framework to derive the new ASEST+ framework and its validation. The improvements for ASEST+ were identified as follows: The first stream of improvements aims at solving the identified difficulties in quasi-experiments 1 and 2 (cf chapter 3) and making the framework more suitable for agile practice education (Research question 7). The second stream of improvements aims to incorporate specific approaches in ASD regarding the two major aspects of cohesion antecedents identified in the previous chapter.

The following research questions are answered:

Research question 7: What approaches regarding agile teamwork in software engineering education can be used to improve the framework learning strategies?

Research question 8: Does the application of the final framework improve team cohesion, team learning, and team performance as perceived by software engineering student teams?

Research question 9: Does team cohesion have a mediational role through the application of the final framework?

Research question 10: What are the perceptions of teachers on the final framework?

The chapter is structured as follows. Section 5.1 answers research question 7 by expanding literature review 1 reported in chapter 2. Section 5.2 informs on the strategies selected to improve the framework regarding the identified cohesion antecedents. Section 5.3 describes the improvements for the ASEST+. Section 5.4 reports on two quasi-experiments and a survey performed to answer research questions 8, 9, and 10. Section 5.5 discusses the limitations of these studies. Section 5.6 concludes the chapter.

5.1 Agile learning strategies

This section answers **Research question 7**: What approaches regarding agile teamwork in software engineering education can be used to improve the framework learning strategies?

The identification of the fundamentals of ASEST0 (i.e. collaborative learning, games and gamification, agile methods, real projects resolution, and links with industry), led to the combination of learning strategies on team-based learning, project-problem based learning, and role-playing gaming approaches in the framework through three phases and eight steps.

To answer the research question, the literature review that led to finding a set of learning strategies to build ASEST0 (chapter 2), was expanded. In doing so, it was considered that from the original literature review Scrum was observed as the most widespread methodology used to teach agile software development.

The industrial report on the latest state of Agile at that time (VersionOne, 2016) showed the use of Scrum to be a trend for companies as well. According to this report, Scrum and its variants (Scrum/XP Hybrid and Scrumban) are used by 68% of the respondents (VersionOne, 2016). Scrum puts more emphasis on project management and does not specifically address technical details for building software, allowing teams to use it together with other agile methodologies like XP. Therefore, Scrum was chosen to develop the improved ASEST+ framework along with some XP practices.

Studies on teamwork in SEE focused on Scrum published after the date of the original review were sought (see search process of the literature review in section 2.1.2). Then, all publications containing the words Scrum, SEE and team in their title, abstract, or keyword list published during 2017 were retrieved. Three more studies were found, all related to the use of Lego games (Paasivaara et al., 2017; Steghöfer et al., 2017; (Monica Villavicencio et al., 2017). Appendix 12 reports the methodologies that were found to be used in those studies addressing agile methods. In all, a total of fifteen studies focusing on Scrum were further analyzed, twelve of which from the original literature review.

Appendix 12 also summarizes the core elements of the fifteen studies focusing on Scrum (i.e. capstone course, Lego blocks game, and agile collaboration) concerning the learning strategies of ASEST0 that were further analyzed to select approaches to improve the framework. The study in (Viljan Mahnic, 2015b) on the use of Scrum in education found the use of capstone projects developed in teams as the most widely adopted strategy, together with the use of simulation games as an alternative to practical project work. These findings are in line with the learning strategies used in ASEST0.

In this literature review, the authors reported four studies that proposed games for teaching Scrum, two of them using Lego blocks (Paasivaara et al., 2014) using plasticine instead as a lower-cost alternative (Ramingwong & Ramingwong, 2015). Moreover, Lego blocks have been used in professional training courses (Krivitsky, 2009). According to (Steghöfer et al., 2017) among the reasons many authors state to use games for teaching Scrum are that games are better at dealing with the time constraints of lectures. Games let students apply Scrum in practice in a short timeframe and enable students to evaluate their process and outcomes (Steghöfer et al., 2017).

Table 32 summarizes the selected studies' main contribution to improving ASEST+, the criteria leading to their selection, and how the identified improvements are included in ASEST+. Overall, the team-based strategy in ASEST+ is focused on using Scrum teams to collaborate in an agile environment. A project-based learning strategy is sustained throughout the capstone course on Scrum and agile practices for developing real-world projects. Also, a role-playing strategy is employed based on the use of Lego games.

Table 32. Selected studies and improvements on learning strategies

Study	Contribution/ Criteria	Improvements
(Kropp et al., 2016)	Presents curriculum design and a set of required competencies for agile developers in a three-level “Agile Competency Pyramid.” This proposal is close to industry needs as it is built based on findings of quantitative and qualitative studies that involved more than 100 software companies	The levels “technical practices” and “collaboration practices” of the pyramid are included in the capstone course. The third level, “agile values,” was not included considering that, as the authors noted, its

		development requires change at the individual level and ASEST+ focuses on behavioral changes at the team level
(Monica Villavicencio et al., 2017)	Presents an adapted educational framework to teach Scrum. The original framework (Monica Villavicencio, 2014), resulting from a doctoral research study, focuses on software measurement education. The author describes its flexibility to be used in developing similar proposals in other domains	The fundamental components of the adapted educational framework are used to formulate the capstone course design
(Paasivaara et al., 2014)	Presents a Lego-based Scrum simulation game. This approach was initially developed as internal training in a Finnish security software company to support their adoption of agile practices, making it coincide with the industry perspective. Its flow covers the main activities included in other papers that report on the use of Lego games (Lynch et al., 2011a), (Paasivaara et al., 2017), (Steghöfer et al., 2017)	The phases of this Lego game are used to set up the main flow of the game during training (Step 3)
(Steghöfer et al., 2017)	Reports on the use of a Lego game to teach Scrum. The teacher's role in this game is closer to the agile coach trainer who acts in industry settings	The teacher's role as coach and the planning poker activity are used in the ASEST+ Lego game. Coaching is included in the course design as well

5.2 Team cohesion antecedents strategies

Intending to select strategies on cohesion antecedents to improve our framework proposal, we returned to the literature review discussed in section 4.3. The studies identified from this literature review related to the relevance of personality traits for team roles and the resolution of conflicts regarding requirements and stakeholders were further studied to identify those studies that focused on Scrum.

Appendix 13 summarizes the methodologies addressed in these studies and lists those studies that were not selected and the main reason leading to this decision. In all, seven studies were found to focus on Scrum. Some approaches did not focus on any methodology in particular.

Table 33 shows the selected studies' main contribution toward improving ASEST+ concerning cohesion antecedents, the criteria leading to these selections, and how the identified improvements are included in ASEST+.

Table 33. Selected studies and improvements on cohesion antecedents

Study	Contribution/ Criteria	Improvements
(Baumgart et al., 2015)	The findings show how personality traits based on the Five-Factor Model (P. T. J. Costa & McCrae, 1992) are relevant for Scrum teams. This model provides a better understanding than the Myers-Briggs Type Indicator questionnaire (MBTI), the other widely used approach for studying personality traits of software developers (Balijepally et al., 2006)	Their proposal on the relevance of personality traits for Scrum teams is used for role allocation (step 1)
(Khan et al., 2014)	This study focuses on Scrum and tackles both requirements and stakeholders conflicts (the most addressed conflict types). In a Scrum project, the decomposition of requirements drives the technical tasks and normally all team members are involved in the selection and prioritization of requirements while the Scrum master coordinates its decomposition. Including this strategy aims to address task interdependence in ASEST+ as well	The study's negotiation model is used for conflict resolution during the Win-Win Lego game to train students in Scrum and team working skills (step 3)

5.3 ASEST+ improvements

This section describes the improvements of ASEST+. The selected studies on the framework learning strategies and team cohesion antecedents described in tables 14 and 15 (sections 5.1 and 5.2) led to improvements regarding three aspects: course design, role allocation, and a Win-Win Lego game. In addition, the identified difficulties from quasi-experiment 1 ad 2 through the students' survey (section 3.2.4) led to improvements regarding the team rules agreements.

Course design: The course aims to apply Scrum along with agile practices to develop real projects while students learn how to work as a team. Scrum puts more emphasis on project management and does not specifically address technical details for building software, allowing teams to use it together with other agile methodologies (Schwaber & Sutherland, 2020). Therefore, Scrum is used as a management framework, and some development practices from extreme programming (XP) are included.

The students need programming and modelling skills as prerequisites. The course design is based on an educational framework that contains inputs, guidelines, and outputs (Monica Villavicencio, 2014), based on and adapted from proposals in (Steghöfer et al., 2017), (Paasivaara et al., 2014) and (Kropp et al., 2016). The inputs refer to the resources needed for developing the activities. In contrast to the proposal in (Monica Villavicencio, 2014), the software tools here are open to being selected by the teachers. In addition to the teaching materials these authors propose (e.g., slides, websites, and books), Lego blocks are used in the Scrum workshop and didactic guides are needed to direct students concerning assignments. In addition, video tutorials are used to prepare students for solving practical exercises during the hands-on laboratories. Reading and watching assignments and short quizzes after lectures or before the laboratories allows the students to expand their knowledge and reinforce their understanding. Scrum templates and a template to record conflicts arising in the project are necessary inputs for project development.

The course curriculum guidelines (shown in Table 34) include the content to be taught, the intended learning outcomes, the teaching and learning activities, and the assessment tasks. The topics can be adapted to introduce new concepts, practices, and tools. In addition to Scrum and ASD introductions, as proposed in (Monica Villavicencio, 2014), technical and collaboration practices are included based on the proposal in (Kropp et al., 2016).

Technical practices concern testing, continuous integration, and clean code. Collaboration practices refer to communication (i.e., communication with customers in order to have a good understanding of the requirements, and intensive and open communication among all stakeholders). Team working skills for conflict resolution are included as well. Scrum is taught through a workshop based on (Steghöfer et al., 2017) and (Paasivaara et al., 2014). The teachers have to ensure that each sprint (a short period during which the Scrum team works to complete a specific work product) of the capstone project is doable in 2-3 weeks and provide coaching sessions.

Finally, the outputs relate to the developed project and the learning gained during the course. Therefore, in addition to the real-world project and experience applying Scrum (Monica Villavicencio et al., 2017), this course's output includes skills in teamwork and agile practices.

Table 34. Course curriculum guidelines

Topics	Intended Learning Outcomes	Teaching & Learning Activities	Assessment Tasks
ASD and teamwork	Describe the values and principles of the Agile Manifesto. Explain concepts of ASD. Characterize agile teams. Explain	Lecture	Reading assignments, Short quizzes

	factors affecting communication and conflicts in agile teams. Give examples of requirements and stakeholders conflicts		
Scrum process	Apply Scrum ceremonies to a simulated project	Workshop	Reading/watching assignments, Short quizzes, Simulation project
Agile practices	Explain the main concepts and aims of clean code, refactoring, continuous integration, and testing in ASD	Lecture	Reading assignments, Short quizzes
	Set up an automated build and test environment	Hands-on laboratories	Reading/watching assignments, Practical exercises
	Apply code formatting, consistent use of language features and naming conventions, and use of meaningful names	Hands-on laboratories	Reading/watching assignments, Practical exercises
	Perform code reviews and solve problems by immediate refactoring	Hands-on laboratories	Reading/watching assignments, Practical exercises
	Perform tests (integration, unit, system, acceptance)	Hands-on laboratories	Reading/watching assignments, Practical exercises
Capstone project	Apply the Scrum process and agile practices to develop a real project. Demonstrate how to effectively communicate with teammates and clients and manage information to get the work done and improve existing implementations. Illustrate conflicts that can occur during the project and its solution	Coaching Project work Team project presentations	Project solution Oral presentations Project reports

Role allocation: During team formation, role allocation is based on personality traits. To that end, the criteria set out in (Baumgart et al., 2015) are used. Their findings state that agreeableness is relevant for filling the roles of product owner and developer, while conscientiousness is important for the role of Scrum Master.

These authors define agreeableness as the individual's extent of friendliness and the degree of trustworthiness. Conscientiousness is related to the extent of organization, commitment, and persistence. Agreeableness is manifested in qualities such as trust, altruism, and compliance, while dutifulness, self-discipline, and striving for achievement are related to conscientiousness (P. T. Costa & Mccrae, 1995).

In ASEST+, the NEO Personality Inventory (P. T. J. Costa & McCrae, 1992) is used to assess the presence and extent of these traits. To assign the roles individuals will play on the team, for each team member, the highest-scoring traits are considered in the following order: the Scrum Masters are assigned first, followed by product owners and then developers.

Win-Win Lego game: In the one day of Scrum workshop training, the teams have to build Lego city projects incrementally following the Scrum process. The Lego game flow described in (Paasivaara et al., 2014) was adopted, as can be seen in Fig 15.

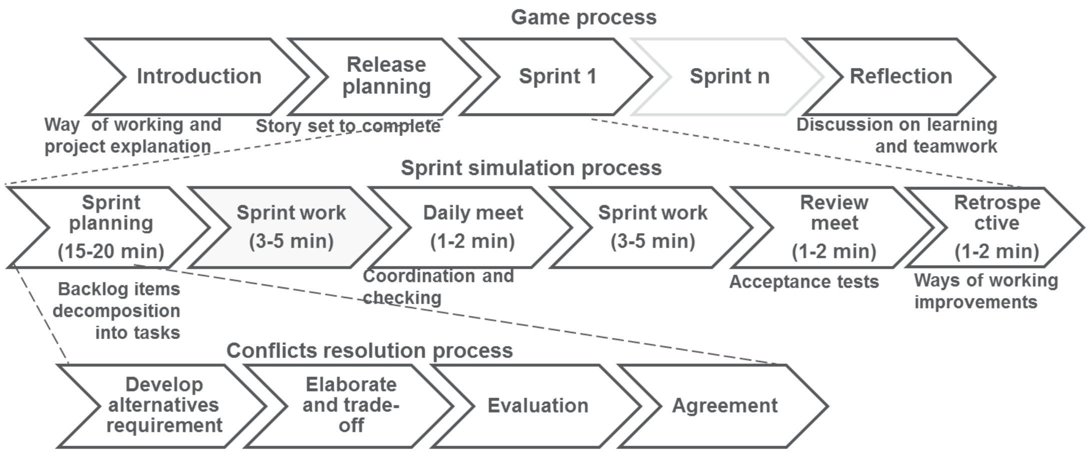

Game process
Introduction
Release planning
Sprint 1
Sprint n
Reflection
Way of working and project explanation
Story set to complete
Discussion on learning and teamwork
Sprint simulation process
Sprint planning (15-20 min)
Sprint work (3-5 min)
Daily meet (1-2 min)
Sprint work (3-5 min)
Review meet (1-2 min)
Retrospe ctive (1-2 min)
Backlog items decomposition into tasks
Coordination and checking
Acceptance tests
Ways of working improvements
Conflicts resolution process
Develop alternatives requirement
Elaborate and trade-off
Evaluation
Agreement

Figure 15. Win-win Lego game flow

One teacher conducts the workshop while other teachers play the role of product owners. As product owners, they answer questions about the product but do not intervene in the project work. The game includes a conflict resolution process based on (Khan et al., 2014). The release planning process incorporates the proposal of (Steghöfer et al., 2017). The teachers explain the prepared backlog of work in descending order of priority.

One planning poker round with randomly selected participants is conducted while the others observe. One user story is chosen, and the students offer their estimation and explain their reasoning. Teachers introduce conflict situations during the sprint planning phase in their role as product owners, and the teams have to collaborate to resolve conflicts. The teachers introduce conflicts regarding requirements priorities, introducing new requirements conflicting with the sprint backlog as well.

Through a win-win negotiation process, team members negotiate by identifying the win conditions from the perspectives of all roles involved. They develop a set of options, evaluate them, iterate on some, reject others and finally converge to a mutually satisfactory agreement. During the review meeting, the product owner again introduces some conflicts by rejecting some requirements and asking to change others. This leads to requirements conflicts that the team has to resolve during the next sprint planning phase.

Team rules agreements: To facilitate reaching an agreement, an evaluation of how the teams communicate and resolve conflicts is conducted through the TPC questionnaire (Heesen et al., 2002). The teams have to agree on rules to improve the problems identified through this evaluation by addressing agile practices. A set of agile practices is provided based on the proposal in (So & Scholl, 2009) to assist students (listed in Table 35). These authors propose a representative set of agile practices for eight common areas: iteration planning, iterative development, continuous integration and testing, stand-up meetings, customer acceptance tests, customer access, retrospectives, and collocation. This set includes well-established agile practices proposed by experienced practitioners. However, it could be enriched over time with updated practices from the industry and other issues that might arise during retrospective meetings. The students are allowed to propose general rules to their agreements as well.

*Table 35. Agile practices set**

Iteration Planning: participation of all team members
All members of the technical team actively participate during iteration planning meetings
All technical team members take part in defining the effort estimates for requirements of the current iteration
When effort estimates differed, the technical team members discuss their underlying assumption
All concerns from team members about reaching the iteration goals are considered

The effort estimates for the iteration scope items are modified only by the technical team members Each developer sign up for tasks on a completely voluntary basis The customer picks the priority of the requirements in the iteration
Iterative Development: short iterations, time-boxing, working software
We implement our code in short iterations The team rather reduces the scope than delay the deadline When the scope cannot be implemented due to constraints, the team holds active discussions on re-prioritization with the customer on what to finish within the iteration At the end of an iteration, we deliver a potentially shippable product We keep the iteration deadlines The software delivers at the iteration end always meet the quality requirements of the production code Working software is the primary measure for project progress
Continuous Integration & Testing: continuous integration, test-driven development
The team integrates continuously Developers have the most recent version of code available Code is checking in quickly to avoid code synchronization/integration hassles... The implemented code is written to pass the test case New code is written with unit tests covering its main functionality Automated unit tests sufficiently cover all critical parts of the production code For detecting bugs, test reports from automated unit tests are systematically used to capture the bugs All unit tests are run and passed when a task is finished and before checking in and integrating There are enough unit tests and automated system tests to allow developers to safely change any code
Stand-Up Meetings: short, regular, focused
Stand up meetings are extremely short (max. 15 minutes) Stand up meetings are to the point, focusing only on what has been done and needed to be done on that day All relevant technical issues or organizational impediments come up in the stand-up meetings Stand up meetings provides the quickest way to notify other team members about problems When people reported problems in the stand-up meetings, team members offered to help instantly
Customer Access: ease of contact to the customer, useful feedback
The customer is reachable The developers can contact the customer directly or through a customer contact person without any bureaucratical hurdles The feedback from the customer is clear and clarifies their requirements or open issues to the developers
Customer Acceptance Tests: frequent, requirements verification by the customer
We apply customer acceptance tests frequently A requirement is not regarded as finished until its acceptance tests (with the customer) have passed Customer acceptance tests are used as the ultimate way to verify system functionality and customer requirements The customer provides a comprehensive set of test criteria for customer acceptance

The customer focuses primarily on customer acceptance tests to determine what have been accomplished at the end of an iteration
Retrospectives: identification and implementation of improvement points
We apply retrospectives frequently
All team members actively participate in gathering lessons learned in the retrospectives
The retrospectives help us become aware of what we did well in the past iteration/s
The retrospectives help us become aware of what we should improve in the upcoming iteration/s
In the retrospectives (or shortly afterward), we systematically assign all-important points for improvement to responsible individuals
Our team follows up intensively on the progress of each improvement point elaborated in a retrospective
Collocation: degree of physical proximity
Developers are located majorly in ...
All members of the technical team (including QA engineers, DB admins) are located in ...
Requirements engineers are located with developers in ...
The project/release manager works with the developers in ...
The customer is located with the developers in …

*Taken from (So & Scholl, 2009)

An example conflict resolution item in the TPC questionnaire is “My team may agree on a solution even though not every member ‘buys into’ that solution.” A rule to address this could be “All concerns from team members about reaching the iteration goals are considered,” addressing the area of “iteration planning.” Another rule could be “When the scope cannot be implemented due to constraints, the team holds active discussions with the customer on re-prioritization and what to finish within the iteration,” addressing the “iterative development” area.

As mentioned earlier, such evaluation was mentioned in the survey of students using the ASEST0 framework as sometimes being unfair. The particular student who made such a mention did not explain the causes of this comment. However, factors such as interpersonal conflicts or poor teamwork participation might have led to less truthful judgments in the peer evaluation.

As it is expected that teamwork improves with time, repeated opportunities to assess members’ contributions could lead to fairer feedback and continuous improvement of the agreements. Therefore, the last two phases in ASEST+ should be repeated in cycles (coinciding with the project sprints) to repeatedly evaluate the effectiveness of rules and team members’ contributions. Occasionally, some regression to phase 1 might be useful, for example, when a student is late in joining the course/team. In this case, time constraints should be carefully considered. Table 36 shows the final version of ASEST+ with the new activities underlined.

Table 36. The ASEST+ framework phases and steps[1]

Phase 1 Preparation: Setting of the learning environment, teams, projects, and introduction activities
Step 1. Setting the scene. Introduction to ASD
Input: Lesson contents, Assignment guides, Project descriptions, NEO Personality Inventory (PI) questionnaire, Short quizzes 1.1 Overview explanation of the framework (10 min) 1.2 Formation of 3-7 member teams (10 min) 1.3 Personality traits identification through the NEO PI questionnaire (30 min) 1.4 Assignation of roles based on personality traits (15 min) 1.5 Assignation of projects to the teams (20 min) 1.6 Lesson on introduction to ASD and agile teamwork (60 min) 1.7 Reading/watching assignments (Individual assignments) (30 min) 1.8 Short quizzes on ASD and agile teamwork (20 min) Output: Team structure; Assignments completed
Step 2. Diagnosis of teamwork skills and conflict-handling styles
Input: TKT and Thomas-Kilman Instrument (TKI), also known as Thomas-Kilmann Conflict Mode Inventory, questionnaires, Assignment guides 2.1 Diagnosis of teamwork knowledge via the TKT questionnaire (30 min, Individual assignment) 2.2 Analysis of the TKT questionnaire results (15 min, Team assignment) 2.3 Diagnosis of conflict-handling style through the TKI questionnaire. (20 min, Individual assignment) 2.4 Analysis of team weaknesses and strengths concerning handling conflicts considering individual styles. (20 min, Team assignment) Output: Individual diagnosis and team lessons learned, Team member conflict-handling styles; Team conflict-handling styles analysis
Step 3. Training on Scrum and team working skills. Introduction to agile practices
Input: Lesson contents, Assignment guides, Short quizzes, Lego blocks, Planning poker cards, Lego project backlog, conflicting situations description, Belbin's roles questionnaire 3.1 Reading/watching assignments on Scrum (60 min, Individual assignment) 3.2 Short quiz on Scrum (20 min, Individual assignment) 3.3 Explanation of the training (10 min) 3.4 Identification of Belbin's roles (optional, 20 min, Individual assignment) 3.5 Scrum process simulation via Lego projects. Resolution of conflicting situations (120 min) 3.6 Analysis of the teamwork demonstrated in the game (20 min, Team assignment) 3.7 Lesson on introduction on agile practices (60 min) 3.8 Reading/watching assignment (30 min, Individual assignment) 3.9 Short quiz on agile practices (20 min, Individual assignment)

[1] The ASEST0 framework can be derived by eliminating the underlined issues

Output: Lego projects developed; Report on the game; Quizzes answered

Phase 2 Implementation: Team rules agreement establishment, sprint project deployment, and agile practices application

Step 4. Team functioning diagnosis. Introduction to automated build and test environments

Input: TPC questionnaire, Lesson contents, Assignment guides, Software tools, Computers, and collocated environment
4.1 Diagnosis of team functioning via the TPC questionnaire (10 min, Individual assignment)
4.2 Elaboration of the joint perception matrix for team functioning (15 min, Team assignment)
4.3 Reading/watching assignment (60 min)
4.4 Hands-on laboratory on automated build and test environments (90 min)
4.5 Team project. Set up/update the automated build and test environment (3 hours, Team assignment)
Output: Joint perception matrix on team functioning; Practical exercises solved; Automated environment setup/updated

Step 5. Team rules agreement establishment. Team project beginning

Input: Joint perception matrix on team functioning, Automated build and test environment, Computers and collocated environment, Scrum templates, Conflicts template, List of systematic agile practices, Assignment guides
5.1 Setting up the agreements. Identification of team rules regarding communication and conflict resolution linked to systematic agile practices. (30 min, Team assignment)
5.2 Team project. User Stories are written and estimated. First version of Product Backlog is written. Sprint and Release Planning is done (5 hours, Team assignment)
Output: Team rules agreement, Product Backlog, Sprint, and Release Planning, Conflicts record

Step 6. Team rules agreement assessment. Agile practices application

Input: Team rules agreement, Assignment guides, Lesson contents, Scrum artifacts, Automated build and test environment, Computers and collocated environment, Scrum templates, Conflicts template
6.1 Assessment of team rules to track their effectiveness and improve the agreements. (e.g. using a scale from 1 to 5) (15 min, Individual assignment)
6.2 Discussion in teams of total scores based on the member's assessments averages, to identify problems and achievements on rules effectiveness. (30 min, Team assignment)
6.3 Reading/ watching assignment (60 min)
6.4 Hands-on laboratory on agile practices (45 min)
6.5 Team project. Sprint execution. (5 hours)
Output: Team agreement assessment report, Practical exercises solved, Product increment, Unit/ Integration/ Acceptance tests, Conflicts record, Scrum artefacts

Phase 3 Adjustment: Agreements adjustment and sprint /project conclusion

Step 7. Feedback

Input: CATME-B questionnaire, Assignment guides, Lesson contents, Scrum artifacts, Automated build, and test environments, Computers and collocated environment, Scrum templates, Conflicts template

7.1 Self and peer evaluations of team member contributions. Completing the CATME-B questionnaire (20 min, Individual assignment)
7.2 Elaboration of the joint perception matrix including the individual average scores of each member's contribution assessment (15 min, Team assignment)
7.3 Discussion in teams of the matrix scores to identify problems (30 min, Team assignment)
7.4 Team project. Development tasks. Sprint Review & Retrospective (5 hours)
7.5 Reading/watching assignments (60 min)
7.6 Hands-on laboratory on agile practices (45 min)
Output: Team member contributions matrix, Practical exercises solved, Scrum artifacts, Unit/Integration/Acceptance tests, Sprint lessons learned, Conflicts record

Step 8. Team agreement update

Input: Team member contributions matrix, team rules agreement, Scrum artifacts, and reports
8.1 Identification of new team rules (30 min)
8.2 Updating the rules agreements (15 min)
8.3 Presentation of the project (2 hours)
Output: Team agreement updated, Project solution

5.4 Framework applicability

This section discusses general framework applicability issues and exemplifies some teaching materials developed to apply the framework. A separate document has been elaborated as an instrument to guide teachers while applying the framework[2]. This guide contains all the questionnaires required to perform the activities, templates that were developed to support the information generated along the process as well as additional resources and further methodological guidelines and examples.

To apply ASEST, time constraints, resource availability, and participants' characteristics should be carefully taken into account, including their fit with limitations dictated by the program/course schedules. Availability of real-world clients and projects and commitment by all parties to participate should be guaranteed beforehand. Cultural issues that might influence students' willingness to use team rules should be investigated as well.

The course curriculum guidelines (presented in Table 34) should be adapted considering the characteristics of the students and target program/course. Therefore, the topics and the intended learning outcomes might vary to make the framework coherent with the aims of the target program/course. The students' level of programming and modeling skills, as well as team working skills expected to be developed within the program up until the point

[2] It can be accessed through the following link:
http://merode.econ.kuleuven.be/publications/Appendix/ASEST_TechnicalReport.pdf

while ASEST will be deployed, should be analyzed as well. The format of teaching and learning activities and assessment tasks of the course curriculum guidelines might also need some adaptations considering time constraints and resource availability. Taking into account these issues, the framework could be applied in a semester-long course, or along several semesters. However, a minimum of eight weeks is required.

To make ASEST fit in the target program/course, time constraints should be considered beforehand. For each activity, the estimated time indicated in Table 36 should be adapted. Some changes in the sequence of activities might be needed as well. The availability of locations where the activities could take place, as well as other resource constraints related to real-world clients, or information required, might lead to some changes in activity formats (e.g. online activity, written report). In doing so the sequence of activities should be considered, as well as the fact that some of them should be done individually and others are required to be performed in teams.

For example, the written report to be written as a team assignment shown in Table 37 has been designed to support activity 1.5 in Table 36 "Assignation of projects to the teams". A similar set of questions could be used to guide the debate of teams in a face-to-face meeting. However, the planned time of the target course/program should allow doing so. In this case, information about the projects should be provided by clients beforehand (See table 38). An alternative format for this activity could be through an online platform which would provide a more flexible timing.

Table 37. Sample of a written report activity

Team assignment: "Feasibility analysis" (questions adapted from (Nguyen, Truong, & Le, 2017)) Elaborate a written report containing the team structure and an analysis of the feasibility of the capstone project. To perform the feasibility analysis, answer the questions below. This analysis will help you to reach a clear understanding of the project and identify issues that require further clarification. • Does the team estimate is the project likely to be a success? (consider similar projects, expertise, knowledge on the dominie where the software will be applied) • Does the team estimate that it is likely to complete the project within the expected schedule? • What are the risks or difficulties of the project? • Do team members have adequate knowledge and skills to complete this project? • Does the team have a clear vision of the product concept that allows them to plan the project? • Has the team defined its project goals, outcomes, and timelines? • Do team members understand well their roles and responsibilities? • Do team members hold each other accountable for the project timelines,

commitments, and results? • Which tools are needed to complete this project?

Table 38. Project scope description template

Company	*Name of the company and its mission*
Project	*Name of the project*
End-users of the project	*Brief description of the end-users*
Project leading	*Person, department, or organization*
Antecedent of the project	*Why is this project running? Factors that led to the need of the project. Person or organization that started the project (or the idea of the project).*
Stakeholders	*Other people/ organizations involved in the project*
Description of the project	*What problem is this project solving? Fundamental purpose/outcome of the project*
Context	*The organizational, cultural context where the project could impact. E.g. Communities, associations, groups*
Resources/ Technologies	*Depending on if it is a new project or an extension of existing, information on required technologies will be needed: e.g Data flow diagrams, architectural diagrams, the definition of done, checklist, security policies, developing frameworks, version control, project structure, 3rd party libraries and packages used*

Considering that the course curriculum guidelines (presented in Table 34) target agile practices with a starting level of complexity, the lesson contents, individual assignments, and short quizzes included in these guidelines, should be developed by teachers to adapt the framework to specific courses objectives. The component of the curriculum guidelines can be combined in the format of didactic guides including content (text documents, lecture notes, videos, images, websites, and other resource links), specific reading/watching activities associated with the content, and the corresponding assessment activities in the form of short quizzes or practical exercises. Table 39 shows an example of a didactic guide related to topic 1 "Agile software development and teamwork".

Table 39. Sample of a didactic guide

• **Content** *(Brief introduction to the topic, objectives, and further bibliography resources)* Requirement engineering is a complex activity that often involves conflicting situations mostly due to different stakeholders' perspectives and priorities. A conflict in requirements engineering exists, if the needs and wishes of different stakeholders regarding the system

contradict each other, or if needs and wishes cannot be considered. Examples of these situations for a software project on a driving system can be:

- A group of stakeholders demands the use of radar sensors for distance measurement. Another group of stakeholders asks, instead, for ultrasound sensors
- A stakeholder demands to display safety-relevant information for the driver on a head-up display. Other stakeholders argue this would detract the driver and hence reject this requirement

In any set of requirements, it is likely to find conflicts, overlaps, and omissions. Conflicts in requirements are a problem that occurs when a requirement is inconsistent with another requirement. Technical reasons are caused by the following difficulties:

- A massive quantity of requirements can lead to conflicts between them.
- Changes in requirements during system development phases. These changes may occur after the addition of new requirements or the update of old ones.
- Complex system domains can lead to misunderstanding of requirements, and therefore, conflicts between them.

Conflicting requirements can be classified as follows:

- Data conflict: Wrong, incomplete information about requirements, different interpretation, different views, and different assessment
- Interest conflict: Interests or goals concerning the system contradict each other
- Value conflict: Different ways of life, ideology, or religion resulting in each stakeholder considering the importance of a requirement differently
- Relationship conflict: Strong emotions, deficient communication, and negative interpersonal behavior between stakeholders
- Structural conflict: Unequal balance of authority or power, destructive patterns of interaction, unequal control, ownership or distribution of resources, and time constraints

While resolving conflicts, new ideas and innovative requirements arise. Thus, conflicts should be seen as opportunities for innovation and expansion.

At the end of this assignment you should be able to:

- Discuss different stakeholders' perspectives, questions, and prioritization categories that are useful to prevent requirement and stakeholders conflicts during requirement elicitation
- Explain how to identify and track requirement conflicts by the Mandatory-Essential-Optional Strategy

Activities *(Orientation about the activities the student are expected to perform to complete the individual assignment)*

This activity aims to give you some guidance for you to be better prepared to prevent conflicting situations during requirement elicitation along with your project. Please, read the document "Agile requirements elicitation". From this reading identify (And keep this information at hand along with your project work!):

- How do users and developers usually view each other?
- What are useful questions to tease out different types of requirements?
- What are the categories that are helpful to ask the customer prioritizing

requirements to early prevent possible conflicts?

Now, put yourself in the shoes of the customer and solve the quiz below.

This activity aims to prepare you to identify and track conflicting requirements from your set of user histories. Please, read the document "Identifying conflicting requirements". Explain how to elaborate a requirement matrix for managing conflicts. Keep the example at hand to help your team project work!

Quiz *(Self-evaluation)*

- Considering the categories you learned from your reading, choose how would you prioritize the following requirements? (Mandatory, Essential, Optional)

Case1: A credit card billing system:

- The system separates the charges by purchase type, to assist the purchaser in understanding buying patterns
- The system lists current charges, sum them, and request payment by a certain date; these are essential requirements
- The system prints the credits in black and the debits in red

Case 2: A hostel management system for a university:

- The system should allow the warden to shuffle multiple students seats
- The system should allow the warden to assign a student a seat in his hostel
- The system should maintain a log of all allotments and vacations in his hostel

*The contents are based on (Pfleeger & Atlee, 2005) and (Aldekhail et al., 2016)

5.5 Validating the ASEST+ framework

In this section, we explore the effects of the new ASEST+ framework on team cohesion, team performance, and team learning as perceived by students. Two quasi-experiments were done with the participation of other teachers in addition to the researcher. In quasi-experiment 3 they performed as assistant teachers while the researcher acted as the main teacher. The assistant teachers conducted quasi-experiment 4 without researcher intervention.

5.5.1 Quasi-experiment 3

This section reports on a quasi-experiment set up to observe the effects of ASEST+ on students' perception of team cohesion, team performance, and team learning. The experimental group was observed during eight weeks in the period Sept.-Nov. 2017. A control group from the same program was observed in the period of Sept.-Nov. 2019. The students were involved in the program on Informatics Engineering at the University of Holguin, in Cuba. Convenience sampling was used. All the students performing in the third and fourth years of the program at that time were included. Table 40 describes the composition and demographics per sub-group for both the experimental and control group.

Table 40. Groups composition and demographics in quasi-experiment 3

	Control group		Experimental group	
	Subgroup 1	Subgroup 2	Subgroup 1	Subgroup 2
Subjects	9	13	12	16
Teams	3	3	4	4
Gender	2F 6M	6F 7M	6F 6M	5F 13M
Age range	21-24	21-25	21-23	21-25

The participants were performing in 2 courses simultaneously (Soft. Eng. II for subgroups 1, and Soft. Eng. III for subgroups 2), performing in teams of 3–5 members. The researcher acted as the main teacher for both courses, and assistant teachers were coaches. In the experimental group the teachers assigned real projects to the teams whereas in the control group the students presented their proposals.

The projects were modules of larger projects in progress (new for the students in the experimental group). The students in the experimental group followed Scrum with some minor changes because students could not work on the project every day due to other duties. The students in the control group used Scrum and Iconix as they were already following these methodologies in these projects. The teachers did not interfere in the distribution of tasks among team members in any group.

Both the experimental and control group students in Soft. Eng. II learned about integration and acceptance tests while in Soft. Eng. III they learned about clean code and unit test practices. Table 41 shows the activities per week, linked to the ASEST+ steps, for the experimental and control groups. In the experimental group phases 2 and 3 were deployed two times, concurring with two project sprints.

The study only covers 8 weeks (out of the 16-course weeks in total), not to interfere with the overall course organization. The courses for the experimental and control groups had different designs. For both groups the students learned the same agile practices; however, the teaching and learning activities were different. In the control group, the teaching materials were slides and books. Their assessment tasks included just practical exercises in the classroom in addition to the capstone project activities. The teaching in this group did not include any team working activity beyond the capstone project development itself. These students were not coached.

Table 41. Courses schedule and intervention in quasi-experiment 3

Week	Control group	Experimental group	
	Activity	Activity	Step
1	Lecture (integration tests/ clean code) Team project (Sprint 1)	Lecture (ASD, teamwork)	1, 2
2	Laboratory (integration tests/ clean code) Team project (Sprint 1)	Scrum Workshop, Lecture (Introduction to technical practices)	3
3	Workshop. Team project (Sprint 1)	Laboratory (Automated environments), Team project (Sprint 1)	4,5
4	Lecture (acceptance tests/ unit tests) Team project (Sprint 2)	Laboratory (clean code/integration tests),Team project (Sprint 1)	6
5	Laboratory (acceptance tests/ unit tests) Team project (Sprint 2)	Laboratory (clean code/integration tests),Team project (Sprint 1)	7, 8
6	Workshop. Team project (Sprint 2)	Laboratory (unit tests/ acceptance tests), Team project (Sprint 2)	4,5
7	Team project (Sprint 3)	Laboratory (unit tests/ acceptance tests), Team project (Sprint 2)	6
8	Team project (Sprint 3)	Team project (Sprint 2)	7,8

5.5.1.1 Intervention

Phase 1:

Phase 1 was done in two weeks. During the first session, the activities of step 1 were done. The teachers explained the course design and the activities of the ASEST+ framework (activity 1.1[3]). The teams were formed (activity 1.2) and the students filled in the NEO PI questionnaire (activity 1.3) to assign the roles based on their personality traits (activity 1.4). As a result, the teams were composed of a scrum master, a product owner, and 1-3 developers (shown in Table 42).

Table 42. Team composition and projects scope from quasi-experiment 3

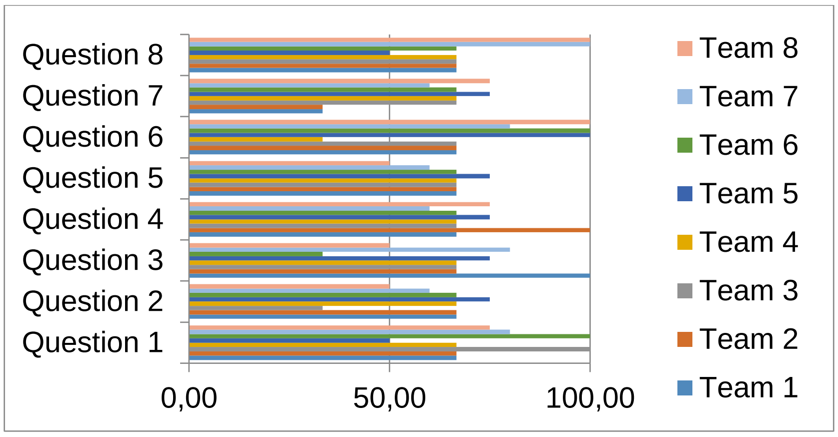
Team members and roles	Project scope

[3] The numbered activities can be found in Table 36

Team 1, 3 members 1 Scrum Master 1 Product Owner 1 Developer	Information management system for commerce. Module for tracking low-selling merchandise. Client: Moa's municipal company of Commerce and gastronomy in Holguin
Team 2, 3 members 1 Scrum Master 1 Product Owner 1 Developer	Information management system for banking. Module for managing mortgage loan applications. Client: BPA bank branch office in Holguin
Team 3, 3 members 1 Scrum Master 1 Product Owner 1 Developer	Information management system for health care supply chain. Module to manage information regarding health care equipment reception and distribution process to the health care units in the municipality. Client: the Municipal Health Care Directorate in Holguin
Team 4, 3 members 1 Scrum Master 1 Product Owner 1 Developer	Information management system for university services. Module to manage information of the reservations in the university students' restaurant. Client: Computerization department at the University of Holguin
Team 5, 4 members 1 Scrum Master 1 Product Owner 2 Developers	Information management system for agricultural products commercialization. Module for managing contracts with producers to purchase and sale of grains (corn, beans, soybean). Client: Agro-industrial grain company of Gibara, Holguin
Team 6, 3 members 1 Scrum Master 1 Product Owner 1 Developer	Real-time software for weighing sugar cane production in situ and updating this information online. Client: Group AZCUBA Holguin
Team 7, 5 members 1 Scrum Master 1 Product Owner 3 Developers	Information management system for university services. Module to manage information related to requests of technical support services. Client: Computerization department at the University of Holguin
Team 8, 4 members 1 Scrum Master 1 Product Owner 2 Developers	Information management system for cultural goods commercialization. Module for assigning sale locations. Client: Marketing department of FCBC institution in Holguin

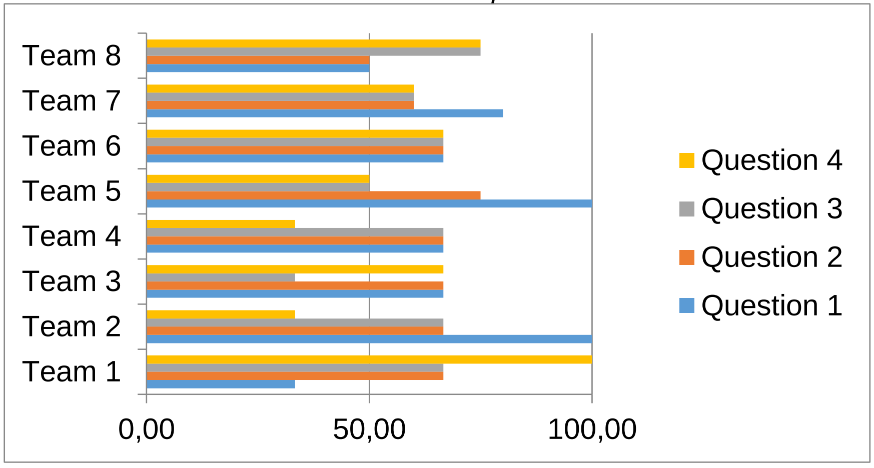
After assigning the projects, the students were oriented to do the project feasibility analysis outside the classroom (activity 1.5). In this way, they could contact clients, investigate, get clarifications and further discuss issues to reach a clear understanding of the project in addition to identifying risks and problems. Then, the introduction lesson to ASD and agile teamwork was given (activity 1.6) and the individual assignments were oriented to be done after the lesson (activity 1.7 and 1.8). During this lesson, the activities included in ASEST0 to improve communication skills by gaining an understanding of nonverbal messages and empathic listening were included (see Table 6). A workshop for step 2 was done two days later, during the same week. After answering the TKT questionnaire (communication and conflict resolution questions) (activity 2.1) the students were asked to check their responses to the correct ones and discuss in teams the results (activity 2.2).

Conflict resolution questions



Q1. When there is a disagreement or difference of opinion in your team, it is generally best to…
Q2. When dealing with a team member, who is not doing his/her fair share of the work, it is best to…
Q3. When you and another team member are having trouble communicating, which is the worst thing for you to do?
Q4. You have gotten quite angry in a team meeting. Which of the following is the least productive thing you could do?
Q5 In order to increase the chances of everyone doing their fair share of work, a team ought to…
Q6. Effective discussions of team business are often made difficult by people who are argumentative or dominating or disorganized. No matter what their problem, to get the meeting moving forward you need to...
Q7. If a member of your team is hostile or critical it is generally useful to…
Q8. Two members of your team have a genuine disagreement (not just miscommunication or personality conflict). Which of the following would be most likely to lead to a resolution?

Communication questions



Q1. When you are listening to other people offering their ideas, it is useful to…
Q2. When receiving feedback from your team members, it is generally useful to…
Q3. When expressing an idea or presenting some information, it is best to…
Q4. If a team member is expressing an opinion different from your own, it is generally helpful to…

Figure 16. TKT questionnaire results from quasi-experiment 3

The results per team (included in Fig 16) showed the majority of questions ranged above 60 % and just a few questions below 40 % which indicates an adequate overall level of knowledge on communication and conflict resolution in the teams. Comparing these results with the ones obtained during the application of ASEST0 (quasi-experiment 1 and 2), it can be noticed that the results were higher. This suggests that the activities on agile teamwork during step 1 (lecture and related assignments) might have contributed to reaching a better preparation.

The students then filled in the TKI questionnaire (activity 2.3) and discussed their conflict-handling styles (activity 2.4). A sample of the ideas the students expressed are shown in Table 43. The teams discussed their profiles and how their styles could influence the teams reaching agreements. Then the teams shared with the whole group their main conclusions. As a reading assignment, the students were provided with the conflicting situation descriptions for Scrum teams used in the team working skills training of ASEST0 along with the solutions the teams proposed for those situations. They were asked to analyze this information as study cases in preparation for the next step.

Table 43. Sample of reflections on team conflict-handling styles from quasi-experiment 3

Team 1 The team has more cooperative and assertive styles (2 collaborating and 1 compromising). We consider this as such a strength for us as we do not have avoiders or competitors; therefore it should be easier to agree on solutions. Collaboration is one of the most important characteristics of successful agile teams
Team 2 The team has 2 compromising and 1 competing. As it lacks more cooperative styles, the team should find a way to effectively reach agreements. Compromising might help in team self-organization. Competing style might be useful for striving to improve, although we should look to share a common vision
Team 3 The team has 1 competing, 1 collaborating, and 1 avoiding. The member who holds collaborating style could mediate in conflicting situations. A commitment of all members to achieve the sprint goal should be in place
Team 4 The team has 1 compromising, 1 accommodating and 1 collaborating. The style of our team is more cooperative. We do not have avoiders or competitors. This should help the team to keep a unified outlook
Team 5 The team has 1 compromising, 1 avoiding, and 2 collaborating. The team does not have competitors or accommodators. We think our team has a balanced style that should help us to collaborate and coordinate tasks

Team 6 The team has 1 compromising, 1 collaborating, and 1 avoiding. Our team does not have competitors or accommodators. The collaborating and compromising styles should be helpful to reach agreements and move forward efficiently
Team 7 The team has 2 compromising, 1 accommodating, and 2 avoiding. We might have problems agreeing on a solution. Compromiser should take the lead to coordinate debates, otherwise, communication problems could arise
Team 8: The team has a collaborative style: 1 collaborating, 2 compromising, and 1 accommodating. We think our team is prepared to reach an understanding which should help to achieve our project goals

During week 2 step 3 was done. The teachers started the week with the assignments on Scrum (activities 3.1 and 3.2). Three sprints for developing the Lego construction projects were conducted (activity 3.3) two days after. A brief discussion on the conflicting situation study cases served as a starting point to explain the activities of the win-win Lego game. The students showed being highly motivated to the game and all the teams completed the Lego project. They also were able to solve the majority of the conflicts that were generated. Table 44 shows a sample of conflicting requirements solved. Table 45 includes a sample of students' reflections on the game.

Table 44. Sample of conflicting requirement solved during the Lego game from quasi-experiment 3

Conflicting requirement	One bus stop per priority building: the shop, school, church, hospital, and kindergarten
Win-conditions	Product owner: Increment of bus stops Developer: Keep the quality of the city design Scrum master: Guarantee the materials needed to fulfill all the requirements
Alternatives requirement	A1 (developer): build one bus stop between priority buildings: one between the shop and the school, a second stop between the church and hospital, and place the kindergarten close to the school A2 (Scrum master): build just a story building instead of two, to get some Lego blocks that can be assigned to the new bus stops
Issues	A1: There is not enough space to build the kindergarten close to the school A2: The product owner does not agree on eliminating one story building as he argues it is vital to have enough place for people to live in this city
Options	A1.1: Reduce the space for the school sports area to gain a place to build the kindergarten A2.1 Eliminate two levels of the story buildings
Evaluation	With option A2.1 the design of the city would be less affected

	(developer's win-condition), the materials would be guaranteed (Scrum master's win-condition), and the bus stops can be incremented (Product owner's win-condition)
Agreement	Eliminate two levels of the story buildings and one bus stop per priority building

Reflecting on the game at the end of the workshop it was repeatedly mentioned by students that time pressure and overlapping roles were among the factors they considered having negatively influenced their work. These criteria suggest that it might be helpful to prolong the first sprint to allow students to better understand the activities they are expected to do and practice them from their roles with more ease.

Table 45. Sample of students reflections on the Lego game from quasi-experiment 3

Team	Reflections
Team 1	Our team was able to finish the project despite the first sprint was not completed in time. All members contributed as expected. All conflicts were solved. The main challenge was that sometimes the roles were overlapping functions
Team 2	The team enjoyed developing the project and worked according to the planning. The scrum master did superb work coordinating activities. The main challenge was to find solutions solving the conflicts mostly because of time pressure
Team 3	Our team completed the project. Everybody collaborated although sometimes the scrum master participated in developing with Legos and others the developers tried to decide on planning. Despite this, the team managed to communicate and coordinate well the sprint activities and reach a good result. The main challenge was the planning
Team 4	The team was not able to finish the last sprint in time as one member was absent for some minutes causing a delay. However, the team did a good job solving the conflicts and planning the project. The two first sprints flowed with ease and all members collaborated according to their roles. The main challenge was to be able to solve the conflicts keeping the quality of the project
Team 5	Our team did a great job. All the conflicts were solved; all the stories were completed in time. The planning was challenging at first but in the end, we succeeded. It was also challenging to contribute just from the team role as everybody wanted to put their hands on the Legos and the planning was difficult in the first place
Team 6	For our team was difficult to make decisions in a short time. This was particularly challenging while solving conflicts. We were able to find solutions but we needed extra time. Despite this, the team was able to complete the project with very short delays
Team 7	Our team enjoyed very much the project and we all collaborate to get the goals for each sprint. We had some delays in sprints 1 and 2, mostly because everybody had so many ideas that it was difficult to reach an agreement under time pressure. This was the main challenge for us, to be able to succeed under time pressure. But the team succeeded and we managed to solve the conflicts
Team 8	The team is happy with the overall results of our work and we think are now better prepared as a team. The first sprint was challenging, mostly the conflict resolution, the team could not reach an agreement in the period we had to solve

	the conflict, mostly because not all members were actively participating in proposing solutions. Nevertheless, during the two other sprints, all members cooperate in a better way and the team succeeded

The day after the workshop, the introduction lesson to agile practices was given (activity 3.4). The related assignments and quizzes were introduced (activity 3.5) and the students were expected to solve them to conclude the first phase.

Phase 2:

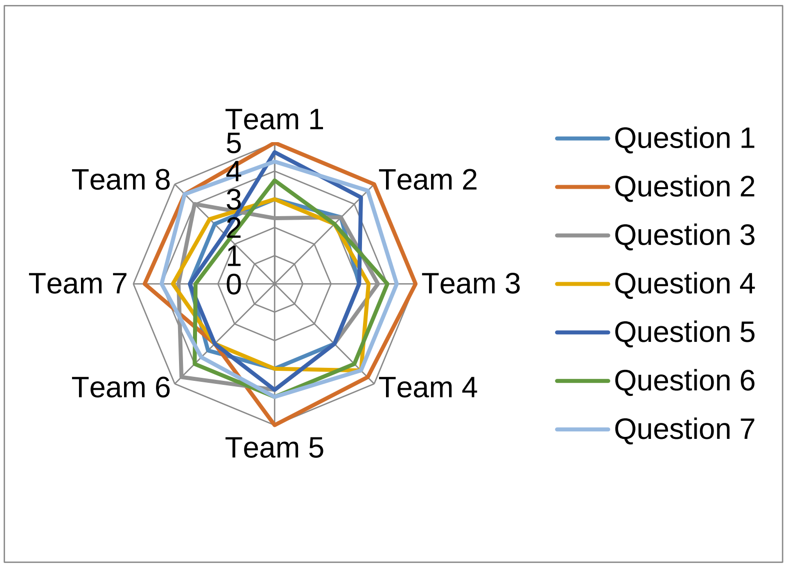
Phase 2 was done two times, i.e. during weeks 3-4 (sprint 1) and 6-7 (sprint 2). Some activities of steps 4 and 5 were reorganized to adapt the framework to the educational activities schedule of the target program. Starting week 3 (and repeated in week 6) the teams were oriented to diagnose team functioning, analyze the results of this evaluation, and elaborate the team rules agreements (activities 4.1, 4.2, and 5.1).

The individual assignment on automated build and test environments was introduced (activity 4.4) and the teams started to elaborate the product backlog and release planning (5.2). The results of the team functioning assessment are shown in Fig 17.

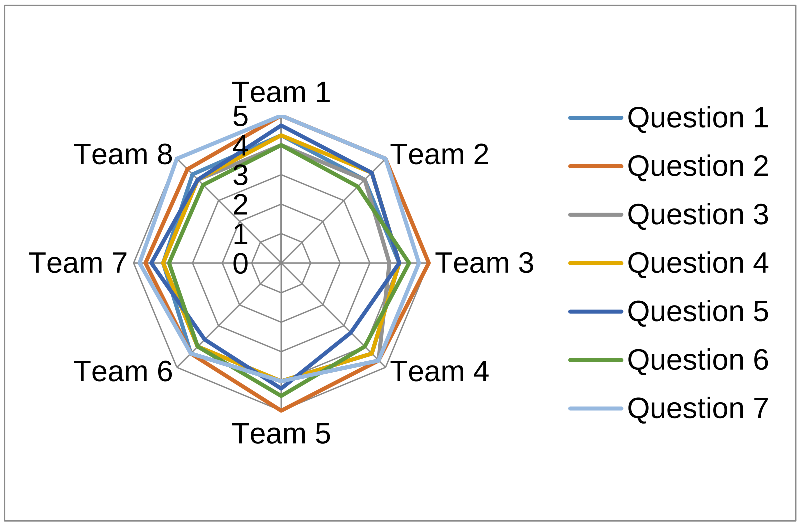
The teams chose agile practices to elaborate the agreements related to areas more familiar to them until that moment: Iteration Planning, Iterative Development, Stand-Up Meetings, Customer Access and Collocation. They were asked to propose no more than 2 rules based on practices to contribute solving the difficulties found from the TPC questionnaire evaluation (items scoring equal or lower than 3 points in teams' averages). In addition, they also included general rules (not more than 3). Table 46 shows a sample of the agreed rules.

The first hands-on laboratory (activity 4.4) was done on day 3, week 3 (and repeated in week 6). Then, the students were instructed to set up the environment to develop their projects (activity 4.5). The teachers coached them during a whole session to guarantee the students' success. Starting the next week of this phase, the students were oriented to individually prepare themselves for the hands-on laboratory on agile practices (activity 6.3) and work in teams on the execution of sprint 1. The individual assignments in preparation for all laboratories on agile practices were exercises with a basic level of complexity.

After a week of performing on agreed rules, an analysis on the effectiveness of the team rules agreement was done (activities 6.1 and 6.2). The analysis was followed by a hands-on laboratory (activity 6.4) in a final session to conclude phase 2. The rules' effectiveness helped to make students aware of difficulties and achievements up to that point.



Assessment Sprint 1



Assessment Sprint 2

Q1. As a team we find it difficult to accept criticism openly and non-defensively (reverse scored)
Q2. When conflict arises in the team, it is likely to be a battle or, at best, a waste of time (reverse scored)
Q3. My team encourages differing opinions to be expressed
Q4. When arguments break out, my team members are able to step back, calm down, and work out our differences
Q5. My team members criticize ideas, not each other
Q6. My team may agree on a solution but not every member "buys into" that solution (reverse scored)
Q7. My team ignores conflicts among team members (reverse scored)

Figure 17. Team functioning results from quasi-experiment 3

Table 46. Sample of agreed team rules and its assessment from quasi-experiment 3

Team	Rules agreement
Team 1	All concerns from team members about reaching the iteration goals are considered (Q1, Iteration Planning)** Decisions are made by consensus (General rule) The retrospectives help us become aware of what we should improve in the upcoming iteration/s (A5, Retrospectives)
Team 2	Stand up meetings are to the point, focusing only on what has been done and needed to be done on that day (Q4, Stand-Up Meetings) Each developer sign up for tasks on a completely voluntary basis (Q6, Iteration Planning) All technical team members take part in defining the effort estimates for requirements of the current iteration (Q6, Iteration Planning) Developers have the most recent version of code available (A3, A4, Continuous Integration & Testing)
Team 3	When effort estimates differ, the technical team members discuss their underlying assumption (Q3, A2 Iteration Planning) All team members actively participate in gathering lessons learned in the retrospectives (Q3, A2, Retrospectives) Respect others' opinions, everybody's opinion matters! (General rule) Working software is the primary measure for project progress (A3, Iterative Development)
Team 4	Stand up meetings are extremely short (max. 15 minutes) (Q1, Stand-Up Meetings) When people report problems in the stand-up meetings, team members offer to help instantly (Q1, Q5, Stand-Up Meetings) Working software is the primary measure for project progress (A3, Iterative Development) Code is checking in quickly to avoid code synchronization/integration hassles (A4, Continuous Integration & Testing) Celebrate team achievements (General rule) Our team follows up intensively on the progress of each improvement point elaborated in a retrospective (A3, Retrospectives)
Team 5	When people report problems in the stand-up meetings, team members offer to help instantly (Q1, Q4, Stand-Up Meetings) All team members actively participate in gathering lessons learned in the retrospectives (Q1, Retrospectives) Discuss with professionalism and respect to avoid personal conflicts (General rule) Code is checking in quickly to avoid code synchronization/integration hassles (A4, Continuous Integration & Testing)
Team 6	Stand up meetings are extremely short (max. 15 minutes) (Q4, Q5, Stand-Up Meetings) Stand up meetings are to the point, focusing only on what has been done and needed to be done on that day (Q4, Q5, Stand-Up Meetings) The retrospectives help us become aware of what we should improve in the upcoming iteration/s (Q5, A3, Retrospectives) Our team follows up intensively on the progress of each improvement point elaborated in a retrospective (Q5, A2, Retrospectives) Respect everybody perspective, all ideas are valid (General rule) Recognize outstanding contributions (General rule)

Team 7	Developers have the most recent version of code available (Q6, Continuous Integration & Testing) The feedback from the customer is clear and clarifies their requirements or open issues to the developers (Q5, Customer Access) Each developer sign up for tasks on a completely voluntary basis (Q6, Iteration Planning) In a conflicting situation, we bring innovative solutions to debate We together see conflicts as an opportunity to grow (General rule) Speak openly, listen carefully, respect always (General rule) Our team follows up intensively on the progress of each improvement point elaborated in a retrospective (A5, Retrospectives)
Team 8	All technical team members take part in defining the effort estimates for requirements of the current iteration (Q6, Iteration Planning) In the retrospectives (or shortly afterward), we systematically assign all-important points for improvement to responsible individuals (Q6, A1, Retrospectives) The team integrates continuously (A1, A4, Continuous Integration & Testing) Everybody’s opinions are heard and considered (General rule) Working software is the primary measure for project progress (A3, Iterative Development)

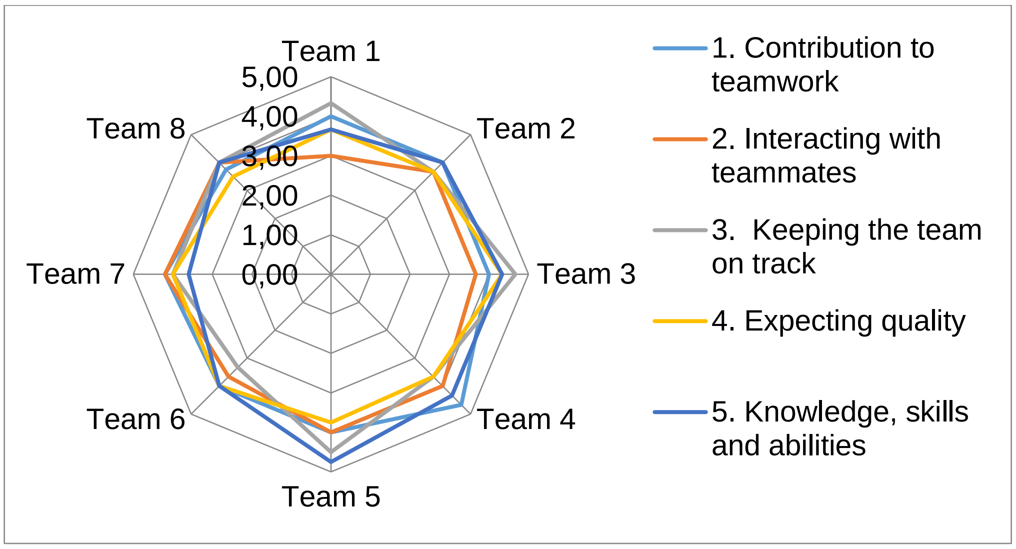
Question (Q) of the TPC questionnaire, Area (A) of the CATME-B questionnaire and Agile practice area target by the rule

Phase 3:

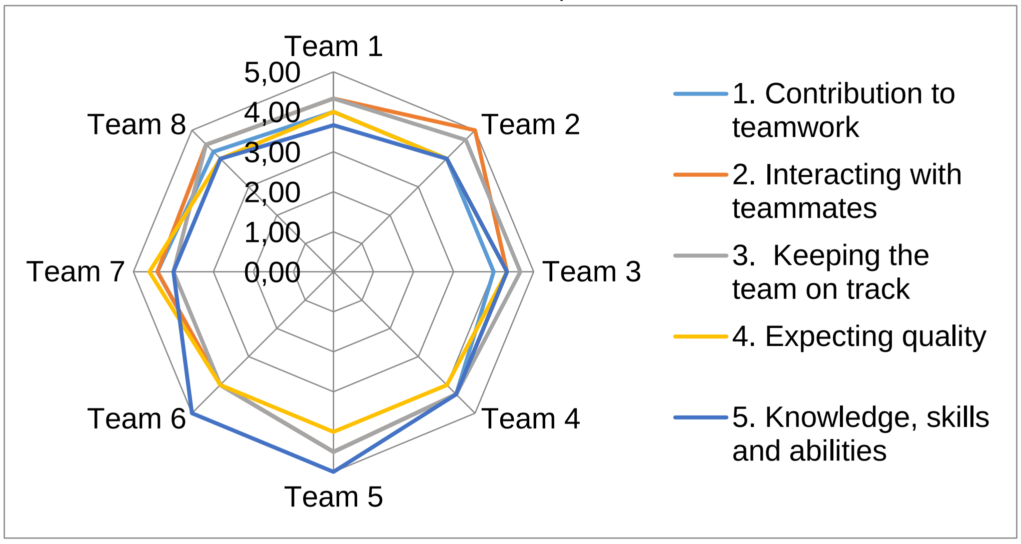
Phase 3 was done two times respectively during weeks 5 (sprint 1) and 8 (sprint 2). The activities were done in two face-to-face sessions along with the self-preparation and team project activities outside the classroom. Starting the week, the students individually prepared for the hands-on laboratory (activity 7.5).

During session 1 (day 3), the teams analyzed how members were contributing to the teamwork (activities 7.1, 7.2, and 7.3) and updated the agreements to improve member contributions (activities 8.1 and 8.2). The results of the team member contribution assessment are shown in Fig 19.

Next, the hands-on laboratory was given (activity 7.6). The assessment done during sprint 1 showed all areas of member contribution scored above 3 points (out of 5). Thus, the students added rules to their agreements for those areas scoring below 4 points. During sprint 2 all areas scored above 4. Then, the students were asked to choose 1 area to be further improved. During session 2 (day 5) the teams presented their work (activity 8.3) to conclude the phase (or the intervention, after the post-measurement of variables).



Assessment Sprint 1



Assessment Sprint 2

Figure 18. Results of the team member contribution assessment from quasi-experiment 3

5.5.1.2 ASEST+ effectiveness

SPSS software version 25 was used to perform the statistical tests. A Shapiro-Wilk test on the students' perceptions for the experimental and control group before the intervention and at the end of the intervention period showed p-values larger than .05 for all variables, from which we could conclude that there is no evidence that the values do not correspond to a normal distribution for team cohesion, team learning, and team performance. Therefore, to check the significance of increases the non-parametric test Mann-Whitney U was used. Table 47 shows the results of the tests. These tests showed that there was a significant difference between the students' perceptions of team cohesion, team performance, and team learning in the experimental group compared to the perceptions of the students in the control group.

Table 47. Mann-Whitney test results from quasi-experiment 3

	Experimental group	Control group	U	p-value (2-tailed)	r
	Mean rank	Mean rank			
Team Cohesion	10.50	3.50	0	.002	0.83
Team Performance	10.50	3.50	0	.002	0.83
Team learning	10.38	3.67	1	.003	0.79

Table 48 shows the means for the pre/post measurements and their deltas. As noted, for both the experimental and control groups team cohesion, performance and team learning were not significantly different for the measurement during the first week. By the last week, the means increased for both groups. However, the increments were only significant for the experimental group.

Table 48. Means and differences of team cohesion, team performance, and team learning for the experimental and control groups in quasi-experiment 3

	Experimental group			Control group		
	Mean before	Mean after	Difference	Mean before	Mean after	Difference
Team Cohesion	59.86	75.09	15.19	57.74	59.26	1.53
Team Performance	14.64	21.62	6.98	14.21	17.90	3.69
Team learning	17.52	29.96	12.44	17.48	21.53	4.05

Finally, Mann-Whitney U tests showed the initial means were not significantly different between the two groups, p values > 0.05, 2-tailed (p = 0.12 for team cohesion, p = 0.64 for team performance, p = 0.70 for team learning). In addition, Hedges' g values showed large effect sizes (5.70 for team cohesion, 2.21 for team performance and 3.05 for team learning).The results of the tests thus suggest that the increase can be attributed to the treatment, although some limitations of this study should be considered (section 5.5).

5.5.2 Quasi-experiment 4

This section reports on a study that replicates the intervention with a few minor changes in the activities order of ASEST+ not to interfere with the overall project aims and course organization. The assistant teachers acting in quasi-experiment 3 conducted quasi-experiment 4 while the researcher did not intervene at all in the study. The quasi-experiment 4 examined ASEST+ framework through a study of eight weeks during the period October - December 2017. The convenience sampling method was used. Table 49 summarizes the characteristics of the group.

Table 49. Groups composition and demographics in quasi-experiment 4

Total of students	16
Total of teams	4
Percent of female	38%
Percent of male	62%
Age range	21-26

As the students were already working in the same teams for some weeks before (3-6), measuring the cohesion of the teams before using ASEST+ could serve as a baseline measurement. Thus, the variables were measured during the first week and right after the last session. The participants were performing in 4 teams of 3–5 members each. The students worked in a software production center of the faculty that coordinated software projects with local industry and departments of the university. Thus, the students worked on the resolution of real problems. The teams were composed of students of the 3rd and 4th years of the program. The teams followed an Iconix methodology to design the solutions and Scrum process to manage their projects as closely as possible with some minor changes because students could not work on the project every day due to other duties. The students were already using Iconix before the intervention started and learned Scrum as a result of the ASEST+ activities. The teachers did not interfere in the distribution of tasks among team members.

The participants of this study were performing in 2 courses simultaneously (Soft. Eng. II for students in 3rd year, and Soft. Eng. III for students in 4th year). The students in Soft. Eng. II learned about integration and acceptance tests while in Soft. Eng. III they learned about clean code and unit test practices. Table 50 shows the teaching-learning activities per week, linked to the ASEST+ steps. The study only covers 8 weeks (out of the 16-course weeks in total) and one sprint of the Scrum project, not to interfere with the overall course and program organization and project schedule.

Table 50. Course schedule and intervention in quasi-experiment 4

Week	Teaching-learning activities	ASEST+ Steps
1	Lecture (ASD, teamwork)	1, 2 (pre-measurement)
2	Scrum Workshop, Lecture (Introduction to technical practices)	3
3	Laboratory (Automated environments), Team project	4,5
4	Laboratory (clean code/integration tests),Team project	4, 5 (cont.)
5	Laboratory (clean code/integration tests),Team project	6
6	Laboratory (unit tests/ acceptance tests), Team project	7,8
7	Laboratory (unit tests/ acceptance tests), Team project	7,8 (cont.)
8	Team project presentation	Post measurement

5.5.2.1 Intervention

Considering that quasi-experiment 4 replicates the intervention described in section 5.4.1.1 of quasi-experiment 3, the phases below will not be described in detail. The activities were performed similarly. Therefore, the description below only contains the order in which the activities were done as it was slightly changed (to not interfere with the overall project and course organization). In addition, it refers to a few differences compared with quasi-experiment 3 that were derived from experiences from this previous intervention or the characteristics of the intervened teams.

Phase 1:

Phase 1 was done in two weeks. During the first session, the activities of step 1 were done. The teachers explained the course design and the activities of the ASEST+ framework (activity 1.1). The teams were already formed, thus, activity 1.2 was not necessary. However, the teams were using just Iconix before the study started. Therefore, the roles were re-organized as proposed in ASEST+. The students filled in the NEO PI questionnaire (activity 1.3) to assign the Scrum roles based on their personality traits (activity 1.4). As a result, the teams were composed of a scrum master, a product owner, and 1-3 developers. The composition of the teams and project scopes can be found in Table 51.

Table 51. Team composition and projects scope in quasi-experiment 4

Team members and roles	Project scope
Team 1, 3 members 1 Scrum Master 1 Product Owner 1 Developer	Information management system for resources planning. Client: "UEB Ómnibus" – Bus transport company, Holguin
Team 2, 4 members 1 Scrum Master 1 Product Owner 2 Developers	Data analytics on students' preferences from the university Virtual environment (Moodle platform). Academic vice rectorate, University of Holguin
Team 3, 5 members 1 Scrum Master 1 Product Owner 3 Developers	Information management system for production process control. Client: "Cerámica Blanca Holguín" Ceramics factory, Holguin
Team 4, 4 members 1 Scrum Master 1 Product Owner 2 Developers	Information management system for controlling capacitation process. Client: "Ceproniquel" - Nickel company, Holguin

After the project assignment, the students were oriented to do the project feasibility analysis (activity 1.5). Then, the introduction lesson to ASD and agile teamwork was given (activity 1.6) and the individual assignments were oriented to be done after the lesson (activity 1.7 and 1.8). A workshop for step 2 was done two days later, during the same week. After answering the TKT questionnaire (communication and conflict resolution questions) (activity 2.1) the students were asked to check their responses to the correct ones and discuss in teams the results (activity 2.2). The results per team showed that the majority of questions ranged above 60 % and just a few questions below 40 %. These results (shown in Fig 19) indicate an adequate overall level of knowledge on communication and conflict resolution in teams.

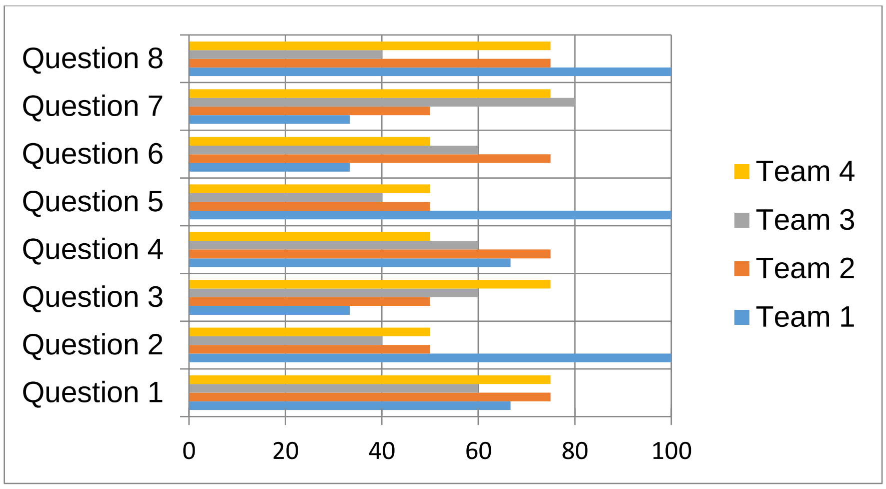
Table 52. Sample of reflections on team conflict handling styles from quasi-experiment 4

Team 1 The team has 1 competing and 2 compromising. The team is lacking collaborative styles which can affect looking for solutions with a unified look. However, the team could benefit from the styles that are present. Compromising style can help the team to get practical solutions to the problems but the team should be able to focus on fairness. Competing should be willing to "fight" fairly, be more persuasive, and not exaggerate his position. Competing can help our team when a more assertive perspective is needed, for example when it is needed something outside the team to get a team goal. The most important thing for our team is the desire to learn how to reach solutions as fast as is needed by agile teams, thus everybody is willing to listen to each other and grow both individually and

as a team
Team 2 The team has 1 avoider, 2 compromising, and 1 collaborator. The avoider style could help the team in avoiding involved emotions during conflicts, which can be a benefit for the trustworthy relationships needed in agile teams. As the team has 2 compromising styles, we should keep in mind that in reaching agreements the quality of the outcomes could be affected. Agile teams are those in which preference for team success is over individual achievements. The collaborator style could help our team to reach understanding and have collaborative discussions for the greatest good of the team
Team 3 The team has 1 competing, 2 collaborating, and 2 avoiding. The main characteristics of an agile team are the desires of all members to collaborate and continue to improve. We think that the collaborating styles could take the lead in solving disagreements to get cooperative solutions and truly work as a team like is required for agile development. The avoiding styles could help avoid emotions while discussing conflicting situations. Competing could help the team in seeing a solution that benefits the team as a whole
Team 4 The team has 2 compromising, 1 accommodating and 1 collaborating. While discussing, our team considers it important to keep in mind that small sacrifices for the greater good of the team are worthy. This can help us to reach agreements faster, as is required by self-organized teams. Accommodating style can help in making a concession, but it is important that everybody understand all points of view and why the concession is granted to avoid resentment. Compromising can help to make concessions without loose too much while collaborating can help to find the best solution for all

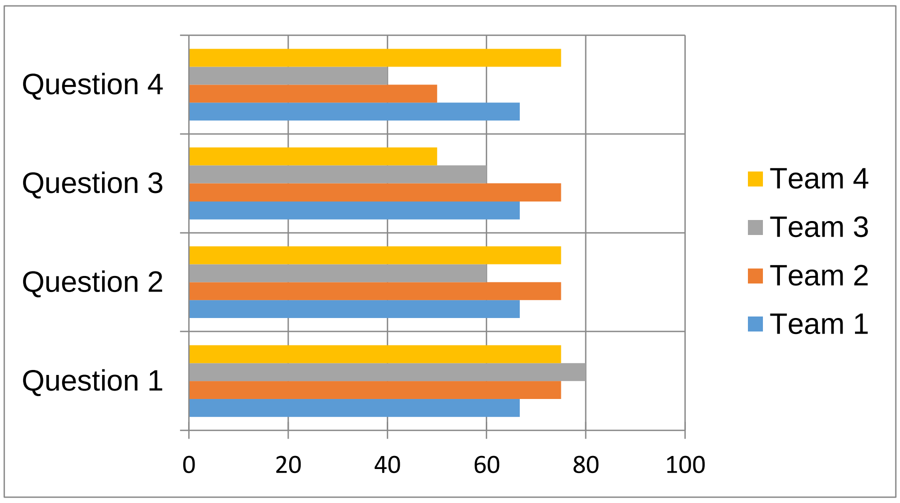
At the end of week 1 the students filled in the TKI questionnaire (activity 2.3) and discussed their conflict-handling styles (activity 2.4). Table 52 includes a sample of the students' reflections during the discussion. During week 2 step 3 was done. The teachers started the week with the assignments on Scrum (activities 3.1 and 3.2). Two days after the workshop on simulation of Scrum process (activity 3.3). Assistant teachers were invited to participate to play the role of product owners. Three sprints for developing the Lego projects construction were conducted. Considering the reflections of the students participating in quasi-experiment 3 on the Lego game, the first sprint lasted twice the time planned for the other 2 sprints to let students fully understand the activities without too much time pressure initially. Another concern of the students in quasi-experiment 3 referred to not enough preparation to perform project planning by using the planning poker technique. Thus, in addition to the planning poker round conducted by selected students to illustrate this technique, all students also performed a simulated round in teams before the Lego project started. This helped students to start the game better prepared to plan the sprints.

Conflict resolution questions



Q1. When there is a disagreement or difference of opinion in your team, it is generally best to…
Q2. When dealing with a team member, who is not doing his/her fair share of the work, it is best to…
Q3. When you and another team member are having trouble communicating, which is the worst thing for you to do?
Q4. You have gotten quite angry in a team meeting. Which of the following is the least productive thing you could do?
Q5 In order to increase the chances of everyone doing their fair share of work, a team ought to…
Q6. Effective discussions of team business are often made difficult by people who are argumentative or dominating or disorganized. No matter what their problem, to get the meeting moving forward you need to...
Q7. If a member of your team is hostile or critical it is generally useful to…
Q8. Two members of your team have a genuine disagreement (not just miscommunication or personality conflict). Which of the following would be most likely to lead to a resolution?

Communication questions



Q1. When you are listening to other people offering their ideas, it is useful to…
Q2. When receiving feedback from your team members, it is generally useful to…
Q3. When expressing an idea or presenting some information, it is best to…
Q4. If a team member is expressing an opinion different from your own, it is generally helpful to…

Figure 19. TKT questionnaire results from quasi-experiment 4

The students showed to be highly motivated to play the game and all the teams completed the Lego project. They were able to solve all the conflicts that were generated by the assistant teachers. Reflecting on the game at the end of the workshop the students still mentioned issues related to roles overlapping. This is, however, a problem that even experienced teams sometimes face. In addition, they asked for more sessions to further develop their skills. The day after the workshop, the introduction lesson to agile practices was given (activity 3.4). Some orientation was given to the students concerning the related assignments and quizzes (activity 3.5), which they were expected to solve in the conclusion of the first phase.

Phase 2:

Phase 2 was done during weeks 3-5. Starting week 3 the teams were instructed to diagnose team functioning (the results are shown in Fig 20), analyze the results of this evaluation and elaborate the team rules agreements (activities 4.1, 4.2, and 5.1). Table 53 includes a sample of the agreed rules.

Table 53. Sample of the agreed team rules from quasi-experiment 4

Team	Rules agreement
Team 1	Stand up meetings are extremely short (max. 15 minutes) (Q1, Stand-Up Meetings)* When people report problems in the stand-up meetings, team members offer to help instantly (Q1, Q5, Stand-Up Meetings) All ideas are heard, we respect others perspectives, we agree the best for the team (General rule) All team members actively participate in gathering lessons learned in the retrospectives (A4, Retrospectives) Code is checking in quickly to avoid code synchronization/integration hassles (A4, Customer Acceptance Tests)
Team 2	Stand up meetings are to the point, focusing only on what has been done and needed to be done on that day (Q4, Stand-Up Meetings) When effort estimates differ the technical team members discuss their underlying assumption (Q5, Iteration Planning) We respect others' opinions, all ideas are valid (General rule) Developers are located majorly in the laboratory (A2, collocation)
Team 3	Stand up meetings provides the quickest way to notify other team members about problems (Q1, Stand-Up Meetings) The feedback from the customer is clear and clarifies their requirements or open issues to the developers (Q3, Customer Access) Listen without interruptions, show respect and understanding (General rule) Developers have the most recent version of code available (A3, Continuous Integration & Testing)
Team	All concerns from team members about reaching the iteration goals are considered

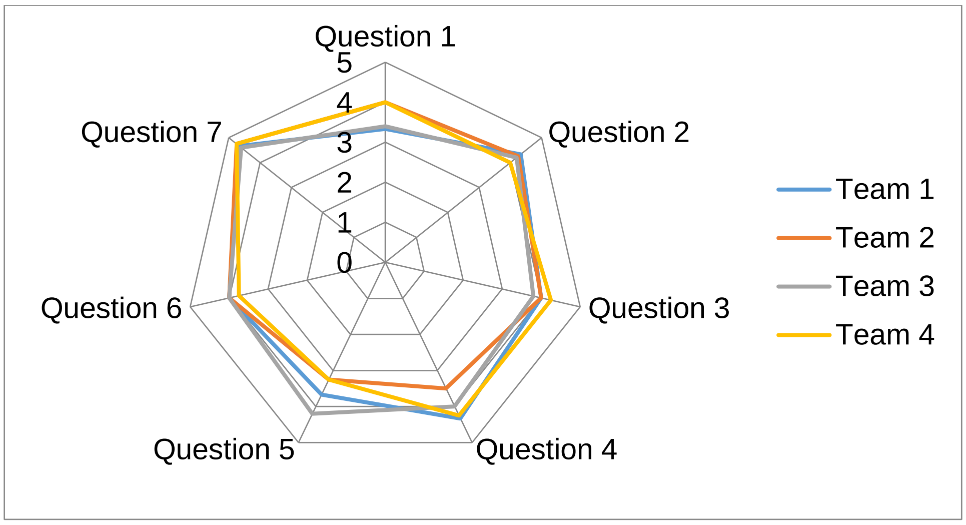
4	(Q5, Iteration Planning) Each developer sign up for tasks on a completely voluntary basis (Q6, Iteration Planning) All technical team members take part in defining the effort estimates for requirements of the current iteration (Q6, Iteration Planning) We are open to new approaches as well as to listening to new ideas (General rule)

*Question (Q) of the TPC questionnaire, Area (A) of the CATME-B questionnaire and Agile practice area target by the rule

The individual assignment on automated build and test environments was oriented (activity 4.3) and the teams started to elaborate the product backlog and release planning (5.2). The first hands-on laboratory (activity 4.4) was done on day 3 in week 3. As the students had already a development environment set up before the intervention started, the students were asked to update the environment according to the particular needs of each team to develop their projects (activity 4.5). Starting week 4, the students were oriented to individually prepare themselves for the hands-on laboratory on agile practices (activity 6.3) and work in teams on the execution of sprint 1 (activity 6.5). Week 5 started with the analysis on the effectiveness of the team rules agreement they set in week 3 (activities 6.1 and 6.2). The analysis was followed by the hands-on laboratory (activity 6.6) in a final session to conclude phase 2.

Phase 3:

Phase 3 was done during weeks 6-8. Starting week 6, the students individually prepared for the hands-on laboratory (activity 7.5). During session 1 (day 3), the teams analyzed how members were contributing to the teamwork (activities 7.1, 7.2, and 7.3) (the results are shown in Fig 21). They also updated the agreements to improve member contributions (activities 8.1 and 8.2). The assessment showed all areas of member contribution scored above 3 points (out of 5). Thus, the students added rules to their agreements for those areas scoring below 4 points. Starting week 7, the hands-on laboratory was given (activity 7.6). The laboratory was followed by a second assessment of the effectiveness of the agreed rule (activities 6.1 and 6.2). The regression to these activities had the intention of making the students aware of the achievements and remaining difficulties regarding the team agreements while they still had two weeks ahead. During the rest of week 7, the teams worked on the project development. The sprint review and retrospective were done starting week 8 (activity 7.4). During the last session (weeks 8, day 5) the teams presented their work (activity 8.3) to conclude the intervention, after post-measurement of the variables.



Q1. As a team we find it difficult to accept criticism openly and non-defensively (reverse scored)
Q2. When conflict arises in the team, it is likely to be a battle or, at best, a waste of time (reverse scored)
Q3. My team encourages differing opinions to be expressed
Q4. When arguments break out, my team members are able to step back, calm down, and work out our differences
Q5. My team members criticize ideas, not each other
Q6. My team may agree on a solution but not every member “buys into” that solution (reverse scored)
Q7. My team ignores conflicts among team members (reverse scored)

Figure 20. Team functioning diagnosis results from quasi-experiment 4

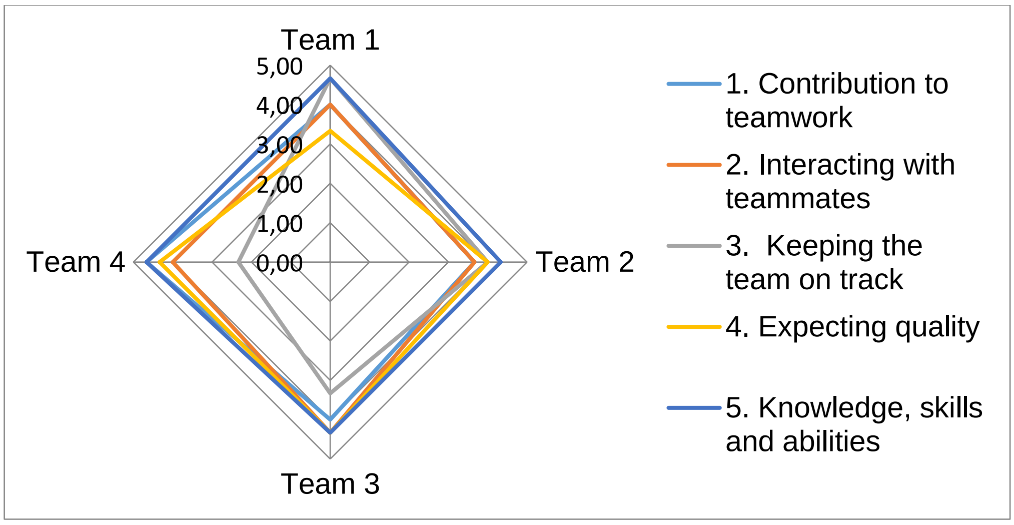


Figure 21. Results of the team member contribution assessment from quasi-experiment 4

5.5.2.2 ASEST+ effectiveness

The SPSS software (version 25) was used to perform the tests. A Shapiro-Wilk test on the students' perceptions for the experimental and control group before the intervention and at the end of the intervention period showed p-values larger than .05 for all variables, from which we could conclude that there is no evidence that the values do not correspond to a normal distribution for team cohesion, team learning, and team performance.

To check the significance of increases t-Student tests were performed using 5000 bootstrap samples with a 95% level of confidence. As shown in Table 54, the tests revealed that the students' perceptions of team cohesion, team performance, and team learning significantly increased by the end of the intervention. In addition, Glass's ∆ values showed large effect sizes (0.65 for team cohesion, 3.07 for team performance, and 2.32 for team learning). These results suggest that ASEST+ might have been responsible for such an increase, although some important limitations of this study should be considered, as we will discuss in section 5.5.

Table 54. T-tests results from quasi-experiment 4

	Mean before	Mean after	Difference	Std. Deviation	t	df	Sig. (2-tailed)
Team Cohesion	48.52	80.23	31.71	5.59	22.99	16	.00
Team Performance	13.12	22.76	9.64	3.14	12.66	16	.00
Team Learning	18.52	31.74	13.22	3.03	18.00	16	.00

5.5.2.3 Mediation analysis

This section reports on a mediation analysis performed to answer **Research question 9**: Does team cohesion have a mediational role through the application of the final framework? Through a mediation analysis, it is possible to investigate to what extent X (antecedents) exerts its effect on Y (outcomes) through M (mediator) within the examination of the process (A. Hayes, 2017). Thus, we investigated whether cohesion mediated the antecedents personality, conflicts, and task interdependence on the outcomes of team performance and team learning, through the application of the ASEST+ framework during quasi-experiment 4.

The criteria of (Maxwell et al., 2011) on the collection of data for mediation analysis was assumed to measure the variables. These authors state that for mediation analyses, data gathered all at one time could introduce biased estimations of the mediators' effects. Therefore, these authors recommend that mediation hypotheses should be tested only with longitudinal data where the antecedents, mediators, and outcomes should be measured separately in time. Therefore, the antecedents personality, conflicts, and task interdependence were measured during the first week. Team cohesion was measured 4 weeks later (in the middle of the intervention) and the outcomes team learning and team performance were measured during week 8 (same used to test the effectiveness of ASEST+).

The procedure outlined by Andrew F. Hayes for testing mediation (Hayes, 2013) and Process macro version 3.1 for SPSS (Hayes, 2012) was used. Specifically, the model 4 of this procedure was selected as it corresponds with the IMO model used to set up our proposal. This approach uses bootstrapping for assessing the effect of X (antecedent) on Y (outcome) through M (mediator), also called *indirect effects of X on Y through M*. Researches have shown that bootstrapping is a more powerful approach to test mediation than other tests such as the Sobel test and the causal steps approach (Mackinnon et al., 2002). While using bootstrapping techniques, no assumptions about the shape of the sampling distribution of the statistic are necessary when conducting inferential tests (Preacher et al., 2007). Hence, the analyses were performed using 5000 bootstrap samples with 95% confidence interval.

Table 55 shows the coefficients and 95% confidence intervals (CI) for the *indirect effects of X on Y through M*. Since zero is not in any confidence interval, it can be concluded that the indirect effect is indeed significantly different from zero at p<.05 (two-tailed) (Hayes, 2009). Thus, it can be affirmed that team cohesion mediated the relationships between the antecedents and outcomes through the application of ASEST+. The indirect effect coefficients in Table 55 specify the amount of mediation. For instance, the values in the first row in table 55 indicates that the indirect effect between task interdependence and team learning was 0.37, suggesting that as task interdependence increases by one unit, team learning increases by 0.37 through the task interdependence effect on team cohesion, which in turn affects team learning.

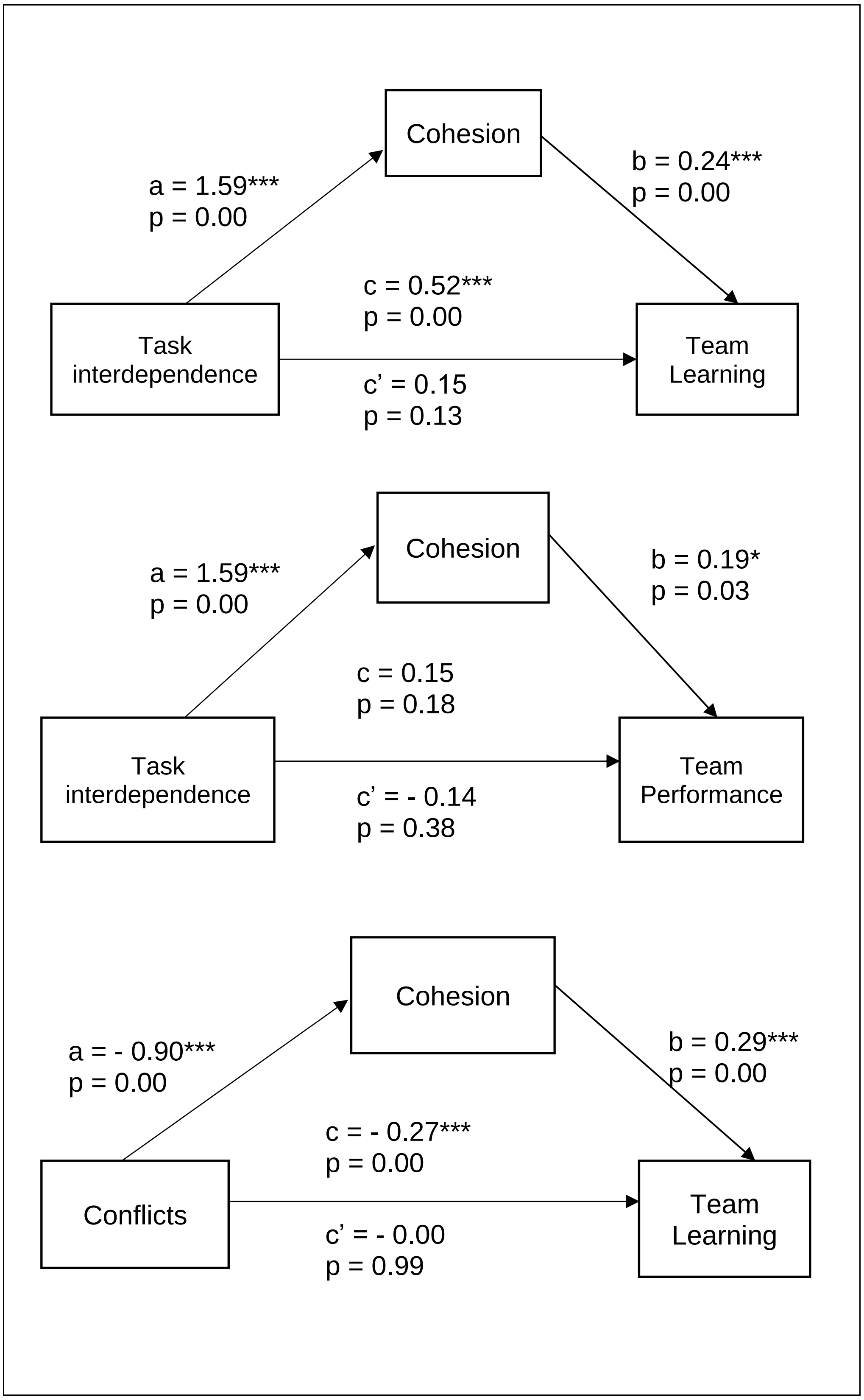
Table 55. Regression coefficients and confidence intervals for the indirect effects of the antecedents personality, conflicts, and task interdependence on the outcomes team learning and team performance through team cohesion

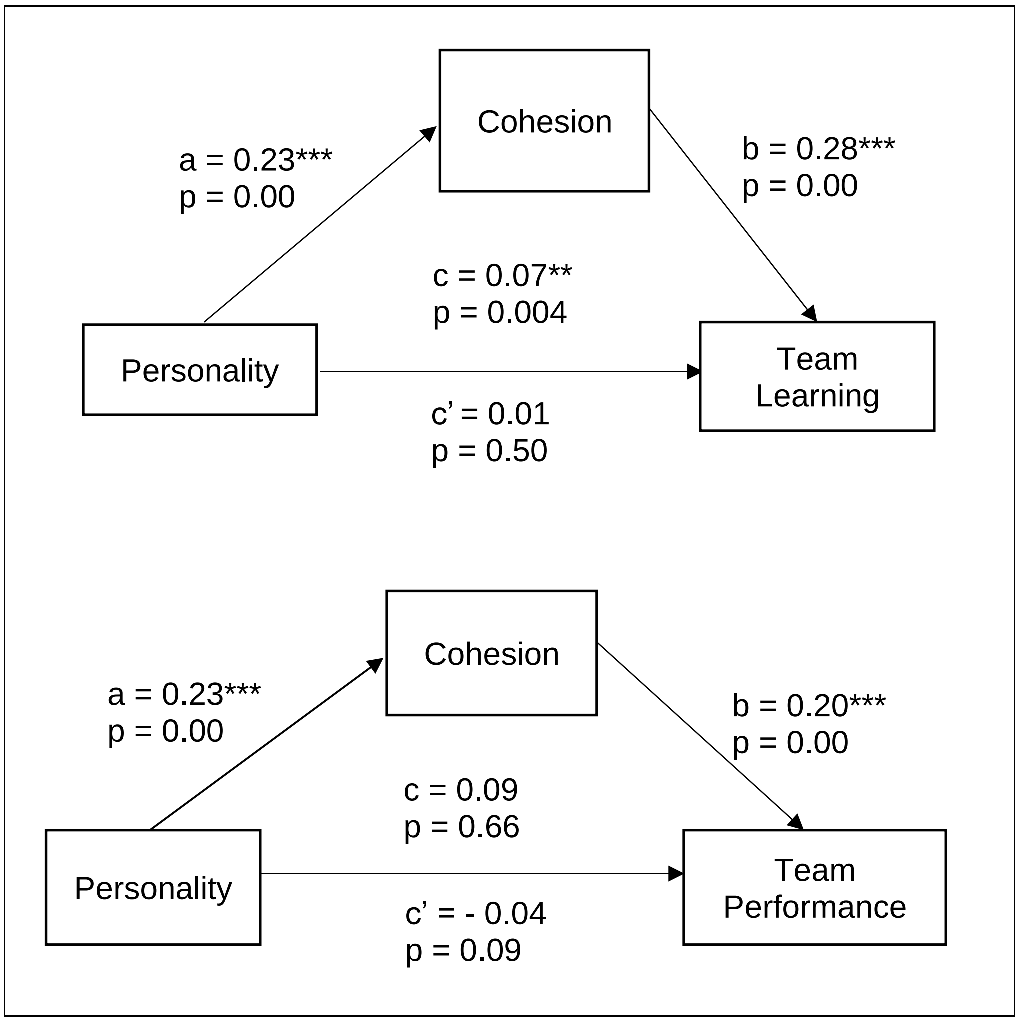
Relationship	B	CI
Task interdependence – Cohesion - Team Learning	0. 37	[0.16, 0.81]
Task interdependence – Cohesion - Team Performance	0. 29	[0.02, 0.75]
Conflicts – Cohesion - Team Learning	- 0.27	[- 0.42, - 0.14]
Conflicts – Cohesion - Team Performance	- 0.22	[- 0.36, - 0.04]
Personality – Cohesion - Team Learning	0.06	[0.04, 0.10]
Personality – Cohesion - Team Performance	0.04	[0.01, 0.08]

Fig 22 shows path diagrams to illustrate the mediational relationships, indicating the regression coefficients and their significance in each path. The regression coefficients are described as follows, according to (Hayes, 2013): coefficient *a* is the coefficient for X in the model predicting M from X. Coefficient *b* is the coefficient for M in the model predicting Y from M. Coefficient c' (called direct effect) is the part of the effect of X on Y that is independent of the pathway through M. Coefficient c (called total effect) is the total extent to which Y is changed by X, that may come to be through a variety of forces both direct and indirect.

As can be seen in Fig 22 the regression coefficients for paths *a* and *b* indicate that task interdependence, conflicts, and personality were significant predictors of team cohesion, and that in all cases team cohesion was a significant predictor of team learning and team performance. These results support the mediational hypothesis. They can be interpreted as: greater values for task interdependence have a positive influence on team cohesion, which in turn leads to greater team learning and team performance. Furthermore, lower values for conflicts can be associated with greater team cohesion, which in turn lead to greater team learning and team performance. In addition, greater values for personality traits of agreeableness, consciousness, extraversion, and openness and less neuroticism have a positive influence on team cohesion, which in turn lead to greater team learning and team performance.

The results in Fig 22 also show that not any of the direct effects (paths *c*’) were significant. The total effects (paths *c*) on team performance were not significant in any case either. However, the coefficients for total effects showed task interdependence and personality to have a significant positive effect on team learning. Also, they indicated conflicts to have a significant negative effect on team learning. Therefore, the total effect of task interdependence on team learning suggests that as task interdependence and personality increase by one unit, team learning increases by 0.52 and 0.07 units, respectively. In addition, the total effect of the antecedent conflicts on team learning suggests that as conflicts decrease by one unit, team learning increases 0.27 units.





*p<.05, **p<.01, ***p<.001

Figure 22. Standardized regression coefficients for the relationship between the antecedents personality, conflicts, and task interdependence and the outcomes team learning and team performance mediated by team cohesion

Although it cannot be affirmed that ASEST+ was responsible for the mediation (as a control group was not included in this study), our findings confirm the mediational role of team cohesion between the antecedents personality, conflicts, and task interdependence and the outcomes team learning and team performance. Considering that ASEST+ has been shown to be effective in quasi-experiments 3 and 4 and that the framework includes strategies regarding the antecedents personality, conflicts, and task interdependence to influence team cohesion, it could be assumed that ASEST+ might have played a positive role to enable this mediation.

5.5.2.4 Teachers perceptions on ASEST+ framework

After the quasi-experiment 4 concluded, the assistant teachers were asked to express their opinion regarding the ASEST+ framework. They were asked to fill in a survey adapted from (Barksdale et al., 2009). This survey was originally developed to evaluate the application of new technologies in agile environments. The teachers were asked to assess each item on practicability and acceptability using a Likert scale in five points: (1) Strongly disagree; (2) Disagree; (3) Neither agree or disagree; (4) Agree; (5) Strongly agree. Then they were asked to share their general impressions and concerns. Practicality refers to the likelihood that the approach could be effectively used in agile SEE. Acceptability questions whether the teachers thought the approach would be accepted by their colleagues.

The teachers' scores on practicality and acceptability of the ASEST+ framework and their overall impressions and concerns (translated from Spanish) are included in Appendix 14. In all, the teachers showed appreciation regarding the ASEST+ framework. They found the approach easy to learn and implement, with a reasonable level of effort. They showed confidence that the approach would fit in with the program and considered that other teachers will adopt the use of it, thus they would encourage the use of ASEST+.

In addition, the teachers expressed positive impressions and a few concerns. Among the general thoughts about the use of ASEST+, the teachers expressed appreciation for the framework and willingness to further use it. In particular, one teacher mentioned his positive opinion on the use of rules to regulate team behaviors as he observed students to be keen on applying such agreements.

The teachers recognized as potential benefits the novelty and usefulness to better prepare software engineers in a crucial area for them. Among the potential costs to implement the proposal, the teachers pointed two main concerns: the time needed to prepare some materials and apply all the activities, and the Lego blocks necessary for the training, which is scarce in Cuba. However, they pointed out that the potential benefits will outweigh the potential costs. Related to the potential costs they recommended sharing experiences about the application of the framework, assigning activities to be done outside of the classroom in case needed, and changing Lego blocks for a more available resource.

The teachers valued the proposal as strategic. They appreciated its novelty and considered ASEST+ flexible to adapt. Regarding the aspects of the framework that might cause them to hesitate using the approach, a teacher mentioned difficulties in tracking the information generated with its application. The teacher proposed the implementation of a management information system to support the activities of ASEST+. He proposed applying artificial intelligence to extend the possibilities by analyzing the information gathered. For example, it was mentioned to recognize patterns of behaviors and anticipate necessary changes.

Finally, a teacher recommended highlighting rules that were assessed as the most effective during phase 2 in such a way that it allows students to hold them as team achievements. It could contribute to building a positive environment. In addition, the teacher suggested providing some guidelines to teachers to facilitate the application of the framework in different scenarios and to evaluate its effectiveness.

5.6 Discussion on validity

In previous chapters and sections above it is already mentioned that in the literature reviews relevant papers might have been missed. Researcher bias regarding the selection and evaluation of the studies is another important limitation of the literature reviews. As the results reported in this chapter (sections 5.1 and 5.2) expand on the previous literature reviews, their limitations should be considered here as well. The constraints described in section 3.4 concerning the sample selection and other conditions of the context leading to choosing for a quasi-experiment design (in quasi-experiment 1) also influenced the research design in quasi-experiment 3 and 4.The sample size is an important limitation of both quasi-experiments and mediational analysis. The results are limited by the fact of not having a random sample. To deal with the small sample size, however, the statistical analysis included bootstrapping techniques.

The sample size is an important limitation of both quasi-experiments and the mediational analysis. In order to deal with the small sample size however, the statistical analysis included bootstrapping techniques. The results of quasi-experiment 4 are limited by the fact of not having a random sample. As mentioned before, population validity is a threat in nearly all educational studies as the majority of researchers are forced to select a sample from the accessible population and random samples are difficult to obtain (Onwuegbuzie, 2003). Both quasi-experiments are also limited by the location in which the investigation took place. The local conditions, socioeconomic status, and cultural background might have influenced these studies.

In quasi-experiment 3, researcher bias like personality traits or attributes of the researcher might have had some impact, as well as the fact that the researcher acted as the main teacher. In this quasi-experiment, the fact that teachers were assigning projects to the students in the experimental group while the students in the control group presented their proposals might have had an influence too. The intervention of teachers was however necessary as the students had to develop extra new features for an existing modular source project. For students to extend these projects they had to dive into existing code, which increased the difficulty of their team projects. Thus, the teachers had to intervene to guarantee the assigned work was feasible during the available time. Another factor to consider in quasi-experiment 3 is students' different levels of expertise on the software methodologies used. While the students in the control group already knew Scrum and Iconix since they were already applying these methodologies in the projects before the beginning of the study, and the students in the experimental group had to learn Scrum before starting the project. This favoured however the control group students. The better performance of the treatment group would therefore be all the more significant.

An issue in quasi-experiment 4 is that even when students have the same curricula, they are normally selected to work on projects because they have shown excellent results in previous courses and are highly motivated to learn. Thus, these conditions might have had a camouflaging impact on our study. However, despite the limitations of this study, the fact that the researcher did not conduct any intervention increases its value to external validity.

In addition, the mediation analyses (quasi-experiment 4) do not address the interactions that can occur between several domains and facets that describe the personality traits (as conceptualized in the five-factor model (P. T. Costa & Mccrae, 1995)). Thus, although our study provides preliminary evidence that suggests a relationship between personality and the outcomes team performance and team learning mediated by team cohesion, further studies that deeper consider the multifaceted nature of human behaviors are required to grasp richer conclusions. Another issue concerning personalities is that while assigning roles according to traits is employed to enhance team cohesion, this strategy might limit the development of students' skills with regard to the different functions members play in a Scrum team. Therefore, allowing students to rotate roles might benefit them in the long term. In this case, further studies that investigate the influence of roles rotation on team cohesion could lead to an improved strategy in ASEST concerning roles allocation.

Finally, the teachers' survey is based on self-perception and the responses may therefore not accurately represent the possible value of the framework. Also, the perceptions of only two teachers are not sufficient to make valid generalizable conclusions. However, they show preliminary evidence of general acceptance and appreciation from teachers that applied this framework proposal.

5.7 Conclusions

This charter reports on the final version of the proposal of this doctoral dissertation: the ASEST+ framework, a proposal to improve teamwork in terms of team learning and team performance, along with its validation. The improvements that led to ASEST+ framework focus on four aspects: course design, roles allocation, win-win Lego game, and team rules agreements. The final proposal focuses on Scrum teams. Approaches regarding team-based learning, project-problem based learning, and role-play gaming were combined in ASEST+ to train the teams on collaborative and technical practices. In addition, ASEST+ establishes policies for roles allocation considering personality traits for Scrum teams. The rules agreements have a more dynamic nature and they are established regarding communication and conflict management linked to agile practices.

The results of a study (quasi-experiment 3) over two groups of students applying ASEST+ indicated their perceptions on team cohesion, team performance, and team learning significantly increased compared with the perceptions of the students in the groups that did not receive this intervention. Another study (quasi-experiment 4) over one group of students applying ASEST+ also indicated their perceptions on team cohesion, team performance, and team learning significantly increased after the intervention.

In addition, quasi-experiment 4 showed team cohesion to mediate the relationships between the antecedents personality, conflicts, and task interdependence and the outcomes team learning and team performance. The results showed that greater values for personality traits of agreeableness, conscientiousness, extraversion, and openness to experience and less neuroticism, as well as greater task interdependence, have a positive influence on team cohesion, which in turn lead to greater team learning and team performance. In addition, lower values for conflicts can be associated with greater team cohesion, which in turn lead to greater team learning and team performance. The results of a survey showed positive perceptions of the teachers that participated in both quasi-experiments on the ASEST+ practicability and acceptability. Further, they offered general impressions and concerns that could be helpful to future improvements of the framework and its application. The next chapter will further discuss the implications of applying this proposal and some directions for future research.

Chapter 6
General discussion

This dissertation presented a teaching and learning framework called ASEST (Agile Software Engineers Stick Together) for the development of team cohesion aimed at improving team performance and team learning of software engineering student teams. The research process included two iterations. Iteration 1 aimed at determining the basis of the framework, setting up a preliminary version of this proposal (ASEST 0), and testing it. The second iteration aimed at improving the proposal and validating the final framework (ASEST+). This chapter concludes the dissertation with a summary of the main contributions of this research and its possible impact in the field of SEE. In addition, some open questions are discussed as possible paths for future research.

6.1 Main contributions

6.1.1 Identification of trends on teamwork in software engineering education

From an analysis of the existing literature on teamwork in SEE, five major trends were obtained: collaborative learning, games and gamification, agile methods, global and virtual teams, and real projects resolution and links with industry. Two iterations over articles from Scopus and Web of Science databases dating from 2006 till 2016 were done. In a first analysis, the keywords were classified to identify possible points of convergence. Then, the resulting classification was put into consideration by other researchers to get an external perspective of the obtained clusters.

The identification of trends allowed obtaining a picture of the strategies and points in common that educators have used to teach teamwork in software engineering. Reviewing the evolution of this domain led to the establishment of the ASEST framework foundations on current teaching and learning approaches. In addition, this compilation provided perspectives for further developments to coherently integrate different approaches in a novel proposal to educate software engineers.

Despite the limitations of the analyses done to obtain the five trends (discussed in section 2.3), the explicitness of these perspectives could serve as a starting point for other researchers and teachers pursuing to improve teamwork in SEE. In addition, it could contribute to highlighting global coherence in this field.

6.1.2 Identification of the relevant aspects affecting cohesion of software engineering student teams

The antecedents of cohesion in SEE reported in the literature were identified. The set of antecedents for collocated teams was narrowed down to the most relevant for agile collocated student teams through a correlational study. From an initial list of 19 antecedents of team cohesion reported in the literature, 6 were identified for collocated teams in educational settings. For these six antecedents, the correlational study was performed showing conflicts, task interdependence, and personality to be the most relevant.

Considering that team cohesion has been thoroughly investigated (Mathieu et al., 2008), a compilation of antecedents of team cohesion in SEE contributes to clarifying and consolidating this specific corpus of knowledge. By identifying how antecedents of team cohesion have been addressed in software engineering education it is also possible to determine research gaps and future research opportunities.

Although the correlational study was limited to Cuban software student teams, its findings contribute to shed some light on the connotation of the addressed antecedents for agile software development teams specifically. The fact that the antecedents task autonomy, team formation, and software engineering methodologies were found not significant for agile software engineering student teams should be considered by future researchers. In particular, the relationship between task autonomy and team formation regarding cohesion should be further investigated.

According to the principles articulated in the agile manifesto, autonomous, self-organized teams are required for the success of agile software development. More recently evidence has been found that task autonomy influences team cohesion for industrial agile software development teams (A. Kumar & Kakar, 2018). However, our result showed it to be different for student teams. Thus, other factors related to educational contexts should be considered while examining such a relationship.

For instance, (Wielenga-Meijer et al., 2012) found that having autonomy when learning a task is crucial, yet having too much of it may lead to adverse learning outcomes when cognitive demands are high. Learning is an ongoing process, while cohesion is an emergent state. Consequently, one could hypothesize that events happening along the learning process (like changes in cognitive demands) might moderate the relationship between task autonomy and cohesion, making this relationship occur only under particular conditions.

The temporary nature of cohesion as an emergent state should be also considered regarding team formation. Our finding shows that the fact that a particular method can influence an initial level of cohesion, does not necessarily mean that it continues being significant concerning cohesion in further development states of the team.

6.1.3 Foundation of ASEST framework to develop cohesion for software engineering student teams

The main contribution of this dissertation is the framework called Agile Software Engineers Stick Together (ASEST). This proposal aims to develop team cohesion, leading to better team learning and team performance of software engineering student teams. This framework's fundamentals were established by identifying teamwork trends in SEE, for example, collaborative learning, games, and gamification, among others (discussed in chapter 2).

The first version of ASEST (presented in chapter 3 as ASEST0) was developed following an IMO (Input-Mediator-Outcome) research model (Mathieu et al., 2008) adapted to software engineering education. In ASEST0, team rules are the input, improved by self and peer assessments of team member contributions. Team cohesion is the mediator, and team learning and performance are the outcomes. The core of ASEST0 is establishing agreed-upon rules related to communication and conflict resolution to regulate team behavior. The ASEST0 framework combines team-based learning, project-problem-based learning, and role-playing game learning strategies in three phases and eight steps.

The novelty of ASEST0 was established by comparing our proposal to existing related teamwork teaching-learning frameworks (section 3.3). The comparison was made in terms of the basis of our proposal (trends reported in chapter 2) and teamwork factors addressed, along with the learning scenario characteristics and main activities included. It was found that while these frameworks focused on one or two trends, ASEST0 considers almost all of them, except for global and virtual teams.

Regarding the factors, team cohesion was found to be addressed in only one framework proposal (Chung-yang Chen et al., 2014), team rules were found not previously studied at all. No studies were found to address team learning or other performance behaviors. Regarding the learning scenario characteristics, besides ASEST0, only the work of

(Alsaedi, Toups, and Cook 2016) included activities to prepare students for team working skills. No proposal included identification and the establishment of team rules and only one framework (Garcia and Pacheco 2014) included peer evaluation for team member contributions, though the criteria used are different.

Although ASEST0 focuses on software engineering students, it might benefit engineering education in general. As the framework does not target specific task routines, its activities could be easily adapted to other engineering teams. Besides, its basis is in alignment with the most widespread approaches used in engineering education, as was discussed in the introduction of this dissertation.

An improved version of the ASEST0 framework called ASEST+ was developed in the second iteration of this doctoral research to address identified difficulties of the preliminary version and the lack of cohesion antecedents. The first stream of improvements aimed to solve these difficulties while making ASEST+ more suitable for agile practice education. In doing so, ASEST0's learning strategies were further evaluated via a literature review. The IMO model guided the second stream of improvements to identify antecedents not yet considered in ASEST0. Two literature reviews and a correlational study were conducted. As a result, two major aspects for improving ASEST0 were found: the relevance of personality traits for team roles and the resolution of conflicts regarding requirements and stakeholders.

The novel aspects of ASEST+ focus on the course design, role allocation, a win-win Lego game, and team rules agreements. ASEST+ focuses on Scrum teams. Approaches to team-based learning, project-problem based learning, and role-playing gaming were combined in ASEST+ to train the teams in collaborative and technical practices. In addition, ASEST+ establishes policies for role allocation within Scrum teams by considering the team members' personality traits.

The rules agreements in ASEST+ have a dynamic nature and are established regarding communication and conflict management linked to agile practices. To our knowledge, there is a lack of proposals of teaching-learning frameworks currently out here to improve team cohesion leading to better agile teamwork. Therefore, the ASEST+ framework attempts to contribute to better prepare software engineers to effectively perform in teams in agile environments.

6.1.4 Experimental validation with ASEST framework

The experimental validation allowed testing the validity of our proposal and to contrast our results with other related studies through four quasi-experiments. ASEST0 was tested using two studies. The quasi-experiment 1 (described in section 3.2.2) reported on a study at the graduate level of a group of students applying ASEST0. It indicated that team cohesion, team performance, and team learning significantly increased compared with the students' perceptions in a group that did not receive this intervention.

The quasi-experiment 2 (described in section 3.2.3) reports on the results of a study of ASEST0 involving three teams of undergraduate students with the ultimate goal of observing ASEST0 with teams performing in a company setting. The study showed the levels of team cohesion, team learning, and team performance increased after the intervention. Despite their limitations, these studies showed ASEST0 to be effective. They also contributed to identifying difficulties with the approach. Moreover, a survey showed positive appreciation of the students on the framework proposal. Their perceptions concurred with the findings from related studies on teamwork teaching-learning frameworks.

During the second iteration, ASEST+ was tested through two quasi-experiments. The results of a study of two groups of students applying ASEST+ (quasi-experiment 3 described in section 5.4.1) indicated that their perceptions of team cohesion, team performance, and team learning significantly increased compared with the perceptions of students in the groups that did not receive this intervention.

The results of the quasi-experiment 4 (described in section 5.4.2) revealed that the students’ perceptions of team cohesion, team performance, and team learning significantly increased by the end of the intervention. Furthermore, the mediational role of team cohesion between the antecedents and the outcomes was evaluated during this last experiment. The tests showed that team cohesion mediated relationships between the antecedents personality, conflicts, and task interdependence and the outcomes team learning and team performance through the application of ASEST+.

The results of these tests showed that greater values for personality traits of agreeableness, conscientiousness, extraversion, and openness to experience and less neuroticism, as well as greater task interdependence, have a positive influence on team cohesion, which in turn lead to greater team learning and team performance. In addition, lower values for conflicts can be associated with greater team cohesion, which in turn lead to greater team learning and team performance.

The mediational study should be considered however as a first step towards understanding the mediational role of team cohesion between the antecedents task interdependence, conflicts, and personality and the outcomes team performance and team learning. The generalization of its findings requires the design of new time-sensitive studies, preferably with larger samples.

In addition, the perceptions of the teachers participating in quasi-experiments 3 and 4 were investigated. Although the perceptions of only two teachers are not sufficient to make valid generalizable conclusions they showed positive appreciation to the teaching-learning framework, and indicated further ideas to improve.

6.2 Theoretical and methodological considerations

This dissertation has aimed at filling the existing gap we found in the software engineering education field regarding the lack of studies addressing the relationship between team cohesion and team performance behaviors. In addition, considering the divergence of findings from previous studies, both perspectives, i.e. team performance outcomes (referred to as ‘team performance’) and team performance behaviors (in terms of ‘team learning’) have been addressed.

As mentioned while presenting the concepts in the introduction, in a meta-analysis of two decades of literature addressing team learning research, (Bell et al., 2012) found that some researchers had aligned team learning with performance outcomes and for others, this construct was conceptualized as behaviors driving learning processes. Bell and his colleagues argued this is a problem considering that researchers sometimes had found teams can learn, yet not experience changes in their performance. The results of this doctoral research show both team performance outcomes and team performance behaviors significantly improve through the application of the ASEST framework.

However, it should be taken into consideration that team cohesion is an emergent state, thus a dynamic entity that changes over time (Marks et al., 2001). While this research has intended to address a time-sensitive approach through the lens of the IMO model (Ilgen et al., 2005) and a longitudinal study design, still more efforts could be beneficial to better understand the temporal dynamics of team cohesion. For instance, a time-series experiment design would be helpful to closer examine dynamic team cohesion and its relationships with other variables over time through the application of ASEST framework.

Time-sensitive designs are important for mediational analyses as mediational processes necessarily develop over time (Maxwell et al., 2011). As said by Maxwell and his colleagues, despite it has been argued that there is a need for longitudinal designs to study mediation; still, most substantive studies continue to be based on cross-sectional designs. Our longitudinal mediational analysis considered this matter, showing evidence of the mediational role of team cohesion between the antecedents personality, conflicts, and task interdependence and the outcomes team performance and team learning.

However, the temporal dynamics of teamwork should be further analyzed not just regarding the mediational role of team cohesion but also including dynamics on team-level inputs (antecedents) and outcomes as well. For instance, task interdependence has been considered by researchers as both an antecedent and a mediating variable (Mathieu et al., 2008). Team learning has been also studied as an outcome while some researchers have found it to mediate relationships (A. C. Edmondson et al., 2007).

Thus, exploring other possible mediation models that include for example several measurements of possible mediators at the same moment could allow analyzing more complex combinations leading to richer conclusions. In addition, it should be taken into consideration that more meaningful results on the personality antecedent could arise from a more fine-grained analysis of traits. The practicability of such studies should be considered however as they are more time-consuming and expensive designs.

6.3 Final words and future research directions

Although the findings presented in this dissertation have limitations in terms of generalization to other populations, the ASEST framework might benefit students in other software engineering programs. Keeping in mind the framework applicability issues (discussed in section 5.4) and the potential limitations outlined, it would be interesting to study the effects of ASEST in other situations, including companies' settings. Considering not only perceptions of students but also measured team cohesion, team performance, and team learning, including the perspective of teachers and clients, could bring more insights into the effectiveness of the framework.

ASEST might also help to improve similar learning environments through knowledge transfer of educational content and case studies derived from its application. In addition, the information gathered could contribute to better understanding software engineering students' teams to further improve teamwork in software engineering education.

We also believe that ASEST0 might be valuable for engineering education, as it does not target specific engineering routines, therefore it could be easily adapted. Further, ASEST+ could be revised in order to be applied with other engineering teams. Two major topics should be addressed: Firstly, what other specific roles should be used to form the teams and how do they relate with personality traits. Second, what rules leading to better team behaviors related to the specific engineering task should be included instead of agile routines?

ASEST might also contribute to new ways of training professional agile teams. In the latest State of Agile Survey (VersionOne, 2016), a long-running report on agile software development it is stated that while the vast majority of respondents and their organizations have succeeded in adopting agile practices, they recognize that there are challenges to scaling agile approaches. As top challenges for adopting agile methods, the participants mentioned the lack of skills or experience with agile methods (47%) along with insufficient training (34%). Although we cannot guarantee that our results can be transferred to organizational settings, the results of quasi-experiment 2 carried out in a company showed preliminary evidence suggesting that ASEST0 might also contribute to some extent to improve the cohesion of teams in such environments.

Despite the final version of ASEST was not tested in organizational settings, its improvements are coherent with the most current strategies for agile software development in professional practice regarding antecedents of team cohesion. Further, its learning strategies are aligned with findings from educational research in software industry as well. Though, in this context, it would be interesting to explore other factors that could influence team cohesion in organizational contexts. For example, it should be considered that in real organizational settings it is possible to find engineers working in several teams at the same time and members can move on or off the teams for several reasons such as project demands and promotions. Then, factors like turnover and other emergent states such as team trust or team empowerment might influence the level of cohesion and the performance behaviors and outcomes.

Further, some changes in ASEST might be necessary. For instance, it might be needed to include other activities as part of the training on team working skills like coordination, decision making, etc. that might be more significant for professional teams. Similarly, the team rules agreement should address these aspects as well. Considering that normally practitioners with different levels of expertise are found in these contexts, the course design, e.g. the topics and intended learning outcomes, might vary as well.

To study the effects of ASEST in professional environments it should be investigated how ASEST would work in an industrial setting before designing any experimentation. Therefore, information on the industrial context should be collected beforehand, i.e. aspects like the nature of the software organization, skills and experience of software staff, type of software products, and technologies used. Before designing an experiment to assess the effects of the framework we should determine the process for allocating and administering the intervention, the population, methods to be used to reduce bias, and the data collection process.

A recommended way to study the effects of ASEST in company settings would be to firstly perform multiple case studies in different companies and investigate similarities before going into more strong experimental designs. In this way, new hypotheses from this type of context could be identified, leading to more focused experimental studies afterward. It would also contribute to identifying the existence of confounding variables and other unexpected sources of bias beforehand.

Since the COVID-19 pandemic has shifted the way we work forcing universities and companies into remote work, it should be investigated if the framework would be still effective when the teams would not work in a face-to-face environment. In addition, it is necessary to further explore the strategies included in this framework considering the student teams working from home. As interaction and team dynamics are different in remote teams, some adaptations might be needed. For instance, address specific training on team working skills concerning team processes issues for virtual teams (e.g. coordination in a dispersed environment).

As suggested by the teachers participating in the quasi-experiments 3 and 4, a software tool to support the application of this proposal has been prototyped. When this software is fully developed it will also facilitate further experimental work. In the ASEST software tool, other teachers' recommendations will be considered. For example, it would be interesting to identify team behavioral patterns related to cohesion antecedents, which in turn might contribute to recognizing necessary team rules that can ultimately lead to improving their performance. In addition, it would be beneficial in such a virtual environment to address mechanisms to prevent requirement conflicts through task interdependences tracking. In doing so, early actions to anticipate possible team cohesion drops could be taken. Also, a way to continue assessing the framework (e.g. its activities, course learning design) could provide further pedagogical guidance to the teachers.

Finally, this dissertation is an invitation to educational researchers for exploring what other aspects at the individual, team and organizational level should be taken into account for an effective application of the framework in different countries, programs, and professional environments. Further, it is a call not only in education but also in the software engineering research field to continue addressing new ways of cooperatively learning and working. In recommending so, we join the request of other researchers (Lenberg et al., 2015) on giving the human and social aspects of software engineering the special attention that it requires.

References

-. (2015). Learning programming through a business project: Engineering education in financial IT case. *Proc. - ICL 2015*, *September*, 527–532. https://doi.org/10.1109/ICL.2015.7318084

Abdelnour-nocera, J., & Sharp, H. (2008). Adopting Agile in a large organization: a Story of Sociotechnical Change. *Proc. - Agile Processes in Software Engineering and Extreme Programming*. https://doi.org/10.1007/978-3-540-68255-4

Abdelnour-Nocera, J., & Sharp, H. (2012). Understanding Conflicts in Agile Adoption through Technological Frames. *Int.Jour.of Sociotechnology and Knowledge Development*, *4*(2), 29–45. https://doi.org/10.4018/jskd.2012040104

Abdul, A., Bass, J. M., Ghavimi, H., & Adam, P. (2018). Product innovation with scrum: A longitudinal case study. *International Conference on Information Society, i-Society 2017*, *2018-Janua*(October), 22–27. https://doi.org/10.23919/i-Society.2017.8354664

ABET. (2012). Criteria for accrediting computing programs. In *Computing*. ABET. http://www.abet.org/uploadedFiles/Accreditation/Accreditation_Step_by_Step/Accreditation_Documents/Current/2013_-_2014/cac-criteria-2013-2014.pdf

Acuña, S. T., Gómez, M., & Juristo, N. (2009). How do personality, team processes and task characteristics relate to job satisfaction and software quality? *Information and Software Technology*, *51*(3), 627–639. https://doi.org/10.1016/j.infsof.2008.08.006

Acuña, S. T., Gómez, M. N., & de Lara, J. (2008). Empirical study of how personality, team processes and task characteristics relate to satisfaction and software quality. *Proc.- ESEM ’08*, 291. https://doi.org/10.1145/1414004.1414056

Ahmad, S., Kasmuri, E., Muda, N. A., & Muda, A. K. (2012). Learning Through Practice Via Role-Playing: Lessons Learnt. In *Proc. - Edulearn12: 4Th Int. Conf. on Education and New Learning Technologies* (Issue July, pp. 5069–5075).

Aktunc, O. (2012). Entropy metrics for agile development processes. *Proc. - 23rd IEEE International Symposium on Software Reliability Engineering Workshops, ISSREW 2012*, 7–8. https://doi.org/10.1109/ISSREW.2012.36

Aldekhail, M., Chikh, A., & Ziani, D. (2016). Software Requirements Conflict Identification: Review and Recommendations. *International Journal of Advanced Computer Science and Applications*, *7*(10).

Alhubaishy, A., & Benedicenti, L. (2017). Toward a model of emotional contagion influence on agile development for mission critical systems. *Proc. - HPCS 2017*, 541–544. https://doi.org/10.1109/HPCS.2017.86

Alkhatib, G., Daoud, D., Zaroor, M. I., Issa, G., & Turani, A. (2011). Incorporating Innovative Practices in Software Engineering Education. *Proc. - IEEE EDUCON Education Engineering 2011 – Learning Environments and Ecosystems in Engineering Education Session*, 1–6.

Alsaedi, O., Toups, Z., & Cook, J. (2016). Can a Team Coordination Game Help Student Software Project Teams? *Proc. - IEEE/ACM Cooperative and Human Aspects of Soft. Eng.*, 33–39.

Anand, R. Vijay, & Dinakaran, M. (2017). Handling stakeholder conflict by agile requirement prioritization using Apriori technique. *Computers and Electrical Engineering Journal*, *61*, 126–136. https://doi.org/10.1016/j.compeleceng.2017.06.022

Anand, R.V., & Dinakaran, M. (2017). Multi-voting and binary search tree-based requirements prioritisation for e-service software project development. *Electronic Government an Interational Journal*, *13*(2).

Argote, L., Gruenfeld, D., & Naquin, C. (2001). Group learning in organizations. In M. E. & Turner (Eds.), *Groups at work: Theory and research* (pp. 369–412).

Arvold, M., & Goode, N. (2015). Teaching Teamwork: A Training Video Designed for Engineering Students. *Proc. - ASEE Annual Conference and Exposition*.

Baldissera, P., & Delprete, C. (2014). Human powered vehicle design: A challenge for engineering education. *Proc. - 12th Biennial Conference on Engineering Systems Design and Analysis, ESDA 2014*, *1*(July). https://doi.org/10.1115/ESDA2014-20549

Balijepally, V., Mahapatra, R., Nerur, S. P., & Nerur, S. (2006). Assessing Personality Profiles of Software Developers in Agile Development Teams. *Communications of the Association for Information Systems*, *18*, 55–75. https://aisel.aisnet.org/cais/vol18/iss1/4

Barbosa, I., Oliveira, M., Reis, P., Gomes, T., & Da Silva, F. (2017). Towards understanding the relationships between interdependence and trust in software development: A qualitative research. *Proc. - CHASE 2017*, 66–69. https://doi.org/10.1109/CHASE.2017.12

Bareiss, R., & Griss, M. (2008). A story-centered, learn-by-doing approach to software engineering education. *ACM SIGCSE Bulletin*, *40*(1), 221. https://doi.org/10.1145/1352322.1352217

Bareiss, R., & Katz, E. (2011). An exploration of knowledge and skills transfer from a formal software engineering curriculum to a capstone practicum project. *Proc. - 24th IEEE-CS Conference on Software Engineering Education and Training*, 71–80. https://doi.org/10.1109/CSEET.2011.5876159

Bareiss, R., & Mercier, G. (2010). A graduate education in software management and the software business for mid-career professionals. *Proc. - Software Engineering Education Conference*. https://doi.org/10.1109/CSEET.2010.26

Barksdale, J. T., Ragan, E. D., Mccrickard, D. S., & Tech, V. (2009). Easing Team Politics in Agile Usability A Concept Mapping Approach. *Proc. - 2009 Agile Conference*.

Barney, H. T., Moe, N. B., Dybå, T., & Aurum, A. (2009). Balancing Individual and Collaborative Work in Agile Teams. *Proc. - Agile Processes in Software Engineering and Extreme Programming: 10th International Conference XP 2009*. https://doi.org/10.1007/978-3-642-01853-4

Basholli, A., Baxhaku, F., Dranidis, D., & Hatziapostolou, T. (2013). Fair assessment in software engineering capstone projects. *6th Balkan Conference in Informatics on - BCI '13*, 244–250. https://doi.org/10.1145/2490257.2490268

Bass, J.M. (2012). Influences on agile practice tailoring in enterprise software development. *Proc. - Agile India 2012*. https://doi.org/10.1109/AgileIndia.2012.15

Bass, Julian M., McDermott, R., & Lalchandani, J. T. (2015). Virtual Teams and Employability in Global Software Engineering Education. *2015 IEEE 10th International Conference on Global Software Engineering*, 115–124. https://doi.org/10.1109/ICGSE.2015.21

Bass, Julian M. (2015). How product owner teams scale agile methods to large distributed enterprises. *Emp. Soft. Eng.*, *20*(6), 1525–1557. https://doi.org/10.1007/s10664-014-9322-z

Bass, Julian M. (2013). Agile Method Tailoring in Distributed Enterprises: Product Owner Teams. *Proc. - IEEE 8th Int. Conf. on Global Soft. Eng.*, 154–163. https://doi.org/10.1109/ICGSE.2013.27

Baum, T., Kortum, F., Schneider, K., Brack, A., & Schauder, J. (2017). Comparing pre-commit reviews and post-commit reviews using process simulation. *Journal of Software: Evolution and Process*, *29*(11), 1–15. https://doi.org/10.1002/smr.1865

Baumgart, R., Hummel, M., & Holten, R. (2015). Personality Traits of Scrum Roles in Agile Software Development Teams - A Qualitative Analysis. *ECIS 2015 Proceedings*, 0–15.

Beal, D. J., Cohen, R. R., Burke, M. J., & Mclendon, C. L. (2003). Cohesion and Performance in Groups: A Meta-Analytic Clarification of Construct Relations. *Journal of Applied Psychology*, *88*(6), 989–1004. https://doi.org/10.1037/0021-9010.88.6.989

Beck, J. (2008). Fair division as a means of apportioning software engineering class projects. In *SIGCSE Bull.* https://doi.org/http://doi.acm.org/10.1145/1352322.1352161

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., & Thomas, D. (2001). *Manifesto for Agile Software Development*. http://www.agilemanifesto.org

Becker, S. A., Brown, M., Dahlstrom, E., Davis, A., DePaul, K., Diaz, V., & Pomerantz, J. (2018). *Horizon Report > 2018 Higher Education Edition* (C. Louisville (ed.)). EDUCAUSE, 2018. https://doi.org/ISBN 978-0-9906415-8-2

Beever, J., & Hess, J. L. (2016). Deepwater horizon oil spill: An ethics case study in environmental engineering. *Proc. - ASEE Annual Conference and Exposition*, *2016-June*. https://doi.org/10.18260/p.26647

Belbin, R. M. (2013). *How to Use Belbin Team Role To Form a Team*. www.belbin.dom

Bell, B. S., Kozlowsk, S. W. J., & Blawath, S. (2012). Team Learning: A Theoretical Integration and Review. In *2012* (pp. 859–909). S. W. J. Kozlowski (Ed.), The Oxford handbook of organizational psychology: Vol. 2.Oxford University Press. http://digitalcommons.ilr.cornell.edu/articles/926%0AThis

Bellenzier, M., Horácio, J., Audy, N., Prikladnicki, R., & Luciano, E. M. (2015). How the Scrum Adoption Relates to Productivity of Software Development Teams? *Proc. - 6th Brazilian Workshop on Agile Methods*. https://doi.org/10.1109/WBMA.2015.18

Bendix, L., & Pendleton, C. (2014). Collaboration in the absence of communication. *Proc. - Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE*, 294–299. https://doi.org/10.1109/WETICE.2014.81

Benton, M. C., & Radziwill, N. M. (2011). A path for exploring the agile organizing framework in technology education. *Proc. - Agile 2011*, 131–134. https://doi.org/10.1109/AGILE.2011.51

Beranoagirre, A. (2011). Utilization of Software Engineering Education in Degree Terminal Subjects in Mechanical Engineering. *Proc. - 4th International Conference of Education, Research and Innovation (ICERI)*, *July*, 2992–2994.

Berkling, K., Geisser, M., Hildenbrand, T., & Rothlauf, F. (2007). Offshore Software Development: Transferring Research Findings into the Classroom. *Proc. - 1st International Conference on Software Engineering Approaches for Offshore and Outsourced Development*, 1–18. https://doi.org/10.1007/978-3-540-75542-5_1

Bezuidenhout, C. N., & Baier, T. J. A. (2011). An evaluation of the literature on integrated sugarcane production systems: A scientometrical approach. *Outlook on Agriculture*, *40*(1), 79–88. https://doi.org/10.5367/oa.2011.0025

Bhannarai, R., & Doungsaard, C. (2017). Agile person identification through personality test and kNN classification technique. *ICSITech 2016 Proccedings*, 215–219. https://doi.org/10.1109/ICSITech.2016.7852636

Bihari, T., Malkiman, I., Chaabouni, M., Bolinger, J., Ramanathan, J., Ramnath, R., & Herold, M. (2011). Enabling scalability, richer experiences and ABET-accreditable learning outcomes in computer science Capstone courses through inversion of control. *Proc. - Frontiers in Edu. Conf., FIE*. https://doi.org/10.1109/FIE.2011.6142872

Bishop, Dave, & Deokar, A. (2014). Toward an understanding of preference for agile software development methods from a personality theory perspective. *Int. Conf. on System Sciences*, 4749–4758. https://doi.org/10.1109/HICSS.2014.583

Bishop, David, Deokar, A. V., & Sarnikar, S. (2016). On Understanding Preference for Agile Methods Among Software Developers. *Information Resources Management Journal*, *29*(3). https://doi.org/10.4018/IRMJ.2016070102

Bleicher, D., Foy, W., & Tabrizi, N. (2012). Rational Team Concert in Software Engineering and Management Education. *Proc. - 8th European Conference on Management Leadership and Governance*.

Boehm, B., & Koolmanojwong, S. (2014). Combining software engineering education and empirical research via instrumented real-client team project courses. In *2014 IEEE 27th Conference on Software Engineering Education and Training, CSEE and T 2014 - Proceedings* (pp. 209–211). https://doi.org/10.1109/CSEET.2014.6816808

Bollen, K. A., & Hoyle, R. H. (1990). Perceived Cohesion: A Conceptual and Empirical Examination. *Soc. Forces*, *69*(2), 479–504.

Borrego, M., Tech, V., Mcnair, L. D., & Tech, V. (2013). Team effectiveness theory from industrial and organizational psychology applied to engineering student project teams — A review. *Journal of Engineering Education*, *102*(4), 1–49.

Bosnic, I., Cavrak, I., Orliic, M., & Zagar, M. (2013). Picking the right project: Assigning student teams in a GSD course. *Software Engineering Education Conference, Proceedings*, 149–158. https://doi.org/10.1109/CSEET.2013.6595246

Bottcher, A., Worch, N., Kamper, A., & Trescher, R. (2013). Using improvisational theater in team-based teaching activities. *IProc. - EEE Global Engineering Education Conference, EDUCON*, 356–361. https://doi.org/10.1109/EduCon.2013.6530129

Bourn, R., & Baxter, S. C. (2014). Creating environments for fostering effective critical thinking in mathematics education (Math-EFFECTs). *Proc. - ASEE Annual Conference and Exposition*.

Bowen, D. M., Alvaro, M., Mejia, D., & Saffi, M. (2005). Industry Practices for Providing Engineers with Team Skills. *Proc. - ASEE Annual Conference and Exposition*, 7999–8006.

Brady, P. A., Chen, J., & Champney, D. (2016). Future K-12 teacher candidates take on engineering challenges in a project-based learning course. *Proc. - ASEE Annual Conference and Exposition*, *2016-June*. https://doi.org/10.18260/p.26993

Branco, D., Prikladnicki, R., & Conte, T. (2012). A preliminary study on personality types in teams Scrum. *CIbSE 2012 Proceedings*.

Bresman, H., & Zellmer-Bruhn, M. E. (2012). The Structural Context of Team Learning: Effects of Organizational and Team Structure on Internal and External Learning. *Organization Science*, *24*(4), 1120–1139.

Brinker, K. R., & Marcolina, R. C. (2016). A lesson in conflict mitigation: integrating divergent design philosophies. *Proc. - Int. Telemetering Conf.*, 506–515.

Brooks, A. K. (1994). Power and the production of knowledge. *Human Resource Development Quarterly*, *5*(3), 213–235.

Buchan, J., Bano, M., Zowghi, D., Macdonell, S., & Shinde, A. (2017). Alignment of Stakeholder Expectations about User Involvement in Agile Software Development. *Proc. - EASE'17*, *June*.

Budd, A. J., & Ellis, H. J. C. (2008). Spanning the gap between software engineering instructor and student. *Frontiers in Education Conference, FIE*, 10–15. https://doi.org/10.1109/FIE.2008.4720516

Buffardi, K. (2015). Localized open source collaboration in software engineering education. *Proceedings - Frontiers in Education Conference, FIE*, *2014*. https://doi.org/10.1109/FIE.2015.7344142

Busetta, P. (2017). Addressing team awareness by means of a requirement prioritization tool. *Proc. - REFSQ Workshop*.

Butgereit, L. (2017). When Agile meets rigid: A case for automated testing with tangible printed reports. *Proc. - NextComp 2017*, 1–4. https://doi.org/10.1109/NEXTCOMP.2017.8016167

Cabrera, I., Villalon, J., & Chavez, J. (2017). Blending Communities and Team-Based Learning in a Programming Course. *IEEE Transactions on Education*, *60*(4), 288–295. https://doi.org/10.1109/TE.2017.2698467

Cagle, R. (2012). Enterprise architecture facilitates adopting Agile development methodologies into a DoD acquisition. *SysCon 2012 - 2012 IEEE International Systems Conference, Proceedings*, 19–23.

https://doi.org/10.1109/SysCon.2012.6189431

Campbell, J. P. (1990). Modeling the performance prediction problem in industrial and organizational psychology. In M. D. D. & L. M. & Hough (Eds.), *Handbook of industrial and organizational psychology* (2nd ed., pp. 687–732). Palo Alto, CA: Consulting Psychologists Press.

Cao, L. (2012). Dynamic capability for trustworthy software development. *J. Softw. Evol. and Proc.*, *24*, 837–850. https://doi.org/10.1002/smr

Cao, L., Mohan, K., Ramesh, B., & Sarkar, S. (2013). Adapting funding processes for agile IT projects: An empirical investigation. *European Journal of Information Systems*, *22*(2), 191–205. https://doi.org/10.1057/ejis.2012.9

Carrillo De Gea, J. M., Nicolás, J., Fernández Alemán, J. L., Toval, A., Ouhbi, S., & Idri, A. (2016). Co-located and distributed natural-language requirements specification: Traditional versus reuse-based techniques. In *Journal of Software: Evolution and Process* (Vol. 28, Issue 3, pp. 205–227). https://doi.org/10.1002/smr.1772

Carron, A. (1982). Cohesiveness in sport groups: interpretations and considerations. *Sport Psychology*, *7*, 123–138.

Carron, A. V., Widmeyer, W. N., & Brawley, L. R. (1985). The Development of an Instrument to Assess Cohesion in Sport Teams: The Group Environment Questionnaire. *Journal of Sport Psychology*, *7*(3), 244–266. https://doi.org/10.1123/jsp.7.3.244

Cartwright, D. (1968). The nature of group cohesiveness. In *D. Cartwright, & A. Zander (Eds.) Group dynamics: Research and theory. 3rd. ed. New York: Harper & Row.*

Cartwright, D., & Zander, A. (1960). Group cohesiveness: Introduction. In *D. Cartwright & A. Zander (Eds.), Group dynam- ics: Research and theory (2nd ed., pp. 69–94). New York, NY: Harper Row.*

Casallas, R., & Lopez, N. (2008). An environment to help develop proffesional software engineering skills for undergraduate students. *International Journal of Engineerig Education*, *24*(4).

Case, S., Schneider, S. K., White, L. J., Kass, S. J., Manning, K., & Wilde, N. (2013). Integrating globally distributed team projects into software engineering courses. *Proc. - 3rd International Workshop on Collaborative Teaching of Globally Distributed Software Development, CTGDSD 2013*, 25–29. https://doi.org/10.1109/CTGDSD.2013.6635242

Castro-Hernandez, A. (2016). Content and temporal analysis of communications to predict task cohesion in software development global teams. *Proceedings - 11th IEEE International Conference on Global Software Engineering Companion Proceedings, ICGSEW 2016*, 65–68. https://doi.org/10.1109/ICGSEW.2016.24

Castro-hernández, A., Swigger, K., & Ponce-flores, M. P. (2015). Effects of Cohesion-Based Feedback on the Collaborations in Global Software Development Teams. *EAI EA Endorsed Transactions on Collaborative Computing*, *1*(6), 1–15.

Castro-Hernandez, A., Swigger, K., Ponce-Flores, M. P., & Teran-Villanueva, J. D. (2016). Measures for predicting task cohesion in a global collaborative learning environment. *Proc. - ICGSEW 2016*, 31–36. https://doi.org/10.1109/ICGSEW.2016.23

Chau, T., & Maurer, F. (2004). *Tool Support for Inter-team Learning in Agile Software Organizations* (pp. 98–109). https://doi.org/10.1007/978-3-540-25983-1_10

Chen, C. Y., & Chong, P. P. (2011). Software engineering education: A study on conducting collaborative senior project development. *Journal of Systems and Software*, *84*(3), 479–491. https://doi.org/10.1016/j.jss.2010.10.042

Chen, Chung-yang, Hong, Y., & Chen, P. (2014). Effects of the Meetings-Flow Approach on Quality Teamwork in the Training of Software Capstone Projects. *IEEE Transactions on Education*, *57*(3), 201–208.

Chen, Chung-yang, & Teng, K. (2011). The design and development of a computerized tool support for conducting senior projects in software engineering education. *Computers & Education*, *56*(3), 802–817. https://doi.org/10.1016/j.compedu.2010.10.022

Chen, Chungyang, Chao, K., & Hung, Y. (2011). *The social and team-related aspects in investigating the meetings-flow approach: A preliminary study*.

Chen, D., Dietrich, C. B., & Dietrich, C. (2011). Transition from Undergraduate Research Program Participants to Researchers and Open Source Community Contributors. *Proc. - ASEE Annual Conf & Expo*.

Chen, J. (2011). An approach for assessing students' performance in software engineering capstone projects. *Int. Conf. on Eng. Educ. and Int. Conf. on Educ. and Educational Tech.*, 161–166.

Chen, J. J. Y., & Wu, M. M. Z. (2015). Integrating extreme programming with software engineering education. In *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2015 - Proceedings* (pp. 577–582). https://doi.org/10.1109/MIPRO.2015.7160338

Chen, J., Qiu, G., Yuan, L., Zhang, L., & Lu, G. (2011). Assessing Teamwork Performance in Software Engineering Education: A Case in a Software Engineering Undergraduate Course. *2011 18th Asia-Pacific Software Engineering Conference*, 17–24. https://doi.org/10.1109/APSEC.2011.50

Chen, L. (2017). Continuous Delivery: Overcoming adoption challenges. *Journal of Systems and Software*, *128*, 72–86. https://doi.org/10.1016/j.jss.2017.02.013

Chen, W.-F. (2010). Work in Progress - An Investigation of Varied Game-Based Learning Systems in Engineering Education. *Proc. - 40th ASEE/IEEE Frontiers in Education Conference*, 26–27.

Chen, W.-F., Wen-Hsiungwu, Chuang, T.-Y., & Chou, P.-N. (2011). The Effect of Varied Game-Based Learning Systems in Engineering Education : An Experimental Study. *Int. Journal of Eng. Edu.*, *27*(3), 482–487.

Chen, W., Wu, W., Wang, T., & Su, C. (2008). Work in Progress - A Game-based Learning System for Software Engineering Education. *38th ASEE/IEEE Frontiers in Education Conference*, 12–13.

Chetankumar, P., & Ramachandran, M. (2009a). Story Card Based Agile Software Development. *International Journal of Hybrid Information Technology*, *2*(2), 125–140.

Chetankumar, P., & Ramachandran, M. (2009b). Story card Maturity Model (SMM): A process improvement framework for agile requirements engineering practices. *Journal of Software*, *4*(5), 422–435. https://doi.org/10.4304/jsw.4.5.422-435

Chetankumar, P., & Ramachandran, M. (2009c). Story Card Process Improvement Framework for Agile Requirements. *Handbook of Research on Soft. Eng. and Productivity Tech.*, 61–84. https://doi.org/10.4018/978-1-60566-731-7.ch006

Chetankumar, P., & Ramachandran, M. (2008). Story cards process improvement framework. *Int. Conf. on Soft. Eng. Research and Practice, SERP 2008*, 415–421.

Chidambaram, L., & Carte, T. (2005). Diversity: Is there more than meets the eye? A longitudinal study of the impact of technology support on teams with differing diversity. *Proc. - 38th Hawaii Int Conf on System Sciences*.

Chilton, M. A. (2012). Technology in the classroom: Using video links to enable long distance experiential learning. *Journal of Information Systems Education*, *23*(1), 51–62.

Chin, C., & Yue, K. (2011). Vertical stream curricula integration of problem-based learning using an autonomous vacuum robot in a mechatronics course. *European Journal of Engineering Education*, *36*(5), 485–504. https://doi.org/10.1080/03043797.2011.603039

Chiocchio, F., & Essiembre, H. (2009). Cohesion and performance: A meta-analytic review of disparities between project teams, production teams, and service teams. In *Small*

Group Research (Vol. 40, Issue 4). https://doi.org/10.1177/1046496409335103

Choi, E. (2013). Applying Inverted Classroom to Software Engineering Education. *International Journal of E-Education, e-Business, e-Management and e-Learning*, *3*(2). https://doi.org/10.7763/IJEEEE.2013.V3.205

Choi, K. S., Deek, F. P., & Im, I. (2008). Exploring the underlying aspects of pair programming: The impact of personality. *Inf. and Soft. Tech.*, *50*(11), 1114–1126. https://doi.org/10.1016/j.infsof.2007.11.002

Chouseinoglou, O. (2015). Introducing Critical Thinking to Software Engineering Education. *Studies in Computational Intelligence*, *496*(January 2014), 183–195. https://doi.org/10.1007/978-3-319-00948-3-12

Collazos, C. A., Ochoa, S. F., Zapata, S., Giraldo, F. D., Lund, I. M., Aballay, L., & Torres de Clunie, G. (2010). CODILA: A Collaborative and Distributed Learning Activity applied to software engineering courses in Latin American Universities. *Proc. - IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing CollaborateCom 2010*, *6*, 1–9.

Congalton, J. (2014). *A Mixed Methods Investigation of Ethnic Diversity and Productivity in Software Development Teams* [Massey University]. http://mro.massey.ac.nz/xmlui/bitstream/handle/10179/5716/02_thesis.pdf?sequence=2&isAllowed=y

Costa, P. T. J., & McCrae, R. R. (1992). NEO Personality Inventory. In *Psychological Assessment Resources*. Spanish version, TEA Ediciones, Madrid, 2002.

Costa, P. T., & Mccrae, R. R. (1995). Domains and Facets: Hierarchical Personality Assessment Using the Revised NEO Personality Inventory. *J. Pers. Assess.*, *64*(1), 21–50. https://doi.org/10.1207/s15327752jpa6401_2

Coultas, C. W., Driskell, T., Burke, C. S., & Salas, E. (2014). A Conceptual Review of Emergent State Measurement: Current Problems, Future Solutions. *Small Group Research*, *45*(6), 671–703. https://doi.org/10.1177/1046496414552285

Cram, W. ., Brohman, M. K., Chan, Y. E., & Gallupe, R. B. (2016). Information & Management Information systems control alignment: Complementary and conflicting systems development controls. *Info. & Manag.*, *53*(2), 183–196. https://doi.org/10.1016/j.im.2015.09.012

Crawford, B., Soto, R., Barra, C. L. de la, Crawford, K., & Olguin, E. (2014). Agile Software Teams Can Use Conflict to Create a Better Products. *Proc. - HCI International 2004*, 24–29. https://doi.org/10.1007/978-3-319-07857-1_43

Curşeu, P. L., & Pluut, H. (2013). Student groups as learning entities: The effect of group diversity and teamwork quality on groups' cognitive complexity. *Studies in Higher Education*, *38*(1), 87–103. https://doi.org/10.1080/03075079.2011.565122

Damian, D., & Borici, A. (2012). Teamwork, coordination and customer relationship management skills: As important as technical skills in preparing our SE graduates. *1st Int.Workshop on Soft. Eng. Educ. Based on Real-World Experiences, EduRex 2012*, 37–40. https://doi.org/10.1109/EduRex.2012.6225704

Damian, D., Lassenius, C., Paasivaara, M., Borici, A., & Schröter, A. (2012). Teaching a globally distributed project course using Scrum practices. *2nd International Workshop on Collaborative Teaching of Globally Distributed Software Development, CTGDSD 2012*, 30–34. https://doi.org/10.1109/CTGDSD.2012.6226947

Dayaram, K., & Fung, L. (2012). Team Performance: Where Learning Makes the Greatest Impact. *Research and Practice in Human Resource Management*, *20*(1), 28–39.

de Assis, D. M., Carvalho Larieira, C. L., & Costa, I. (2017). Difficulties in the adoption and use of scrum method in brazilian companies using plan- driven processes: multiple case study. *Revista de Gestao e Projetos*, *8*, 5585.

De Farias, I. H., De Azevedo, R. R., De Moura, H. P., & Da Silva, D. S. M. (2012). Elicitation of communication inherent risks in distributed software development. *Proc. -*

IEEE 7th International Conference on Global Software Engineering Workshops, ICGSEW 2012, 37–42. https://doi.org/10.1109/ICGSEW.2012.18

De Melo, C. O., S. Cruzes, D., Kon, F., & Conradi, R. (2013). Interpretative case studies on agile team productivity and management. *Information and Software Technology*, *55*(2), 412–427. https://doi.org/10.1016/j.infsof.2012.09.004

Dechant, K., Marsick, V. J., & Kasl, E. (1993). Towards a model of team learning. *Studies in Continuing Education*, *15*(1), 1–14.

Decker, R. (1995). Management team formation for large scale simulations. *Developments In Business Simulation & Experiential Exercises*, *22*, 128–129.

Decuyper, S., Dochy, F., & Van den Bossche, P. (2010). Grasping the dynamic complexity of team learning: An integrative model for effective team learning in organisations. *Educational Research Review*, *5*, 111–133.

Demirors, E., Sarmagik, G., & Demirors, O. (1997). The Role of Teamwork in Software Development : Microsoft Case Study. *23rd EUROMICRO Conf.: New Frontiers of Info. Tech.*

Denninger, O. (2008). Game Programming and XNA in Software Engineering Education. *Proc. - CGAT08*. https://pdfs.semanticscholar.org/3d5d/a2d296971608419b38d1cc17dc8a0e54ca0c.pdf

Díaz, J., Pérez, J., & Garbajosa, J. (2014). Agile product-line architecting in practice: A case study in smart grids. *Information and Software Technology*, *56*(7), 727–748. https://doi.org/10.1016/j.infsof.2014.01.014

Dingsoyr, T., Faegri, T. E., Dyba, T., Haugset, B., & Lindsjorn, Y. (2016). Team Performance in Software Development: Research Results versus Current Advice. *IEEE Software*, *33*(4), 106–110. https://doi.org/10.1109/MS.2016.100

Dingsøyr, T., & Lindsjørn, Y. (2013). Team performance in agile development teams: Findings from 18 focus groups. In *Lecture Notes in Business Information Processing* (Vol. 149). https://doi.org/10.1007/978-3-642-38314-4_4

Dingsøyr, Torgeir, Fægri, T. E., Dybå, T., Haugset, B., & Lindsjørn, Y. (2016). Team Performance in Software Development. *Voice of Evidence. IEEE Software*.

Dingsøyr, Torgeir, Nerur, S., Balijepally, V., & Moe, N. B. (2012). A decade of agile methodologies: Towards explaining agile software development. *Journal of Systems and Software*, *85*(6), 1213–1221. https://doi.org/10.1016/j.jss.2012.02.033

Distante, D. (2007). *Challenges and Lessons Learned in Teaching Software Engineering and Programming to Hearing-impaired Students*.

Doman, M., Besmer, A., & Anne, O. (2015). Teaching Tip Managing Software Engineering Student Teams Using Pellerin's 4-D System. *Journal of Information Systems Education*, *26*(4), 257–265.

Domino, M. A., Collins, R. W., Hevner, A. R., & Cohen, C. F. (2004). Conflict in collaborative software development. *Proc. - SIGMIS Conference '03*, 44–51. https://doi.org/10.1145/761849.761856

Drury, M., Conboy, K., & Power, K. (2012). Obstacles to decision making in Agile software development teams. *Journal of Systems and Software*, *85*(6), 1239–1254. https://doi.org/10.1016/j.jss.2012.01.058

Druskat, V. U., & Kayes, C. (2000). Learning versus Performance in Short-Term Project Teams. *Small Group Research*, *31*(3), 328–353. https://doi.org/10.1177/104649640003100304

Duim, L. van der, Andersson, J., & Sinnema, M. (2007). Good practices for Educational Software Engineering Projects. *29th Int. Conf. on Soft. Eng*.

Dullemond, K., Van Gameren, B., & Van Solingen, R. (2009). How technological support can enable advantages of agile software development in a GSE setting. *Proceedings - 2009 4th IEEE International Conference on Global Software Engineering, ICGSE*

2009, 143–152. https://doi.org/10.1109/ICGSE.2009.22

Dumslaff, U. (2008). Change management: From knowledge about innovative SE to capabilities for industrial SE projects. *Proceedings International Conference on Software Engineering*. https://doi.org/10.1145/1368088.1368204

Dunai, L. D., Prieto, A., Chillaron, M., & Antonino-Daviu, J. A. (2015). Education in electric and electronic engineering via students involvement in innovative projects. *IECON 2015 - 41st Annual Conference of the IEEE Industrial Electronics Society*, *November*, 1862–1866. https://doi.org/10.1109/IECON.2015.7392372

Edmondson, A. (1999). Psychological safety and learning behavior in work teams. *Administrative Science Quarterly*, *44*(2), 350. https://doi.org/10.2307/2666999

Edmondson, A. C., Dillon, J. R., & Roloff, K. S. (2007). Three Perspectives on Team Learning. *The Academy of Management Annals*, *1*(1), 269–314. https://doi.org/10.1080/078559811

Elizalde, H., Rivera-Solorio, I., Pérez, Y., Morales-Menéndez, R., Orta, P., Guerra, D., & Ramírez, R. A. (2008). An educational framework based on collaborative reverse engineering and active learning: A case study. *International Journal of Engineering Education*, *24*(6), 1062–1070.

Ellis, A. P. J., Hollenbeck, J. R., Ilgen, D. R., Porter, C. O. L. H., West, B. J., & Moon, H. (2003). Team learning Collectively connecting the dots. *Journal of Applied Psychology*, *88*(5), 821–835.

Ellis, H. (2014). *Software Engineering : Effective Teaching and Learning Approaches and Practices*. *January 2007*. https://doi.org/10.1145/1226816.1226822

Ellis, H., Hislop, G. W., & Purcell, M. (2013). Project Selection for Student Involvement in Humanitarian FOSS. *26th Int. Conf. on Soft. Eng. Eduation and Traininnt. Conf. On Soft. Eng. Eduation and Training*.

Ellis, H. J. C. (2007). An assessment of a self-directed learning approach in a graduate Web application design and development course. *Ieee Transactions on Education*, *50*(1), 55–60. https://doi.org/10.1109/te.2006.888907

Ellis, H. J. C., & Hislop, G. W. (2013). Project selection for student participation in humanitarian FOSS. *Proceedings of the 13th Annual ACM SIGITE Conference on Information Technology Education - SIGITE '13*, 155. https://doi.org/10.1145/2512276.2512326

Emam, A., & Mostafa, M. G. (2012). Using Game Level Design as an Applied Method for Software Engineering Education. *Proceedıngs of the 7th International Conference on Computer Games (CGAMES), 2012*, 248–252. https://doi.org/10.1109/CGames.2012.6314583

Epstein, R. G. (2008). A Software Engineering Course with an Emphasis on Software Processes and Security. *Proc. - 21st Conference on Software Engineering Education and Training*, 67–73. https://doi.org/10.1109/CSEET.2008.19

Esfahani, H. C., & Yu, E. (2010). *A Repository of Agile Method Fragments* (pp. 163–174). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-14347-2_15

Etzkorn, L. H., Gholston, S. E., Fortune, J. L., Stein, C. E., Utley, D., Farrington, P. A., & Cox, G. W. (2004). A comparison of cohesion metrics for object-oriented systems. *Information and Software Technology*, *46*(10), 677–687. https://doi.org/10.1016/j.infsof.2003.12.002

Etzkorn, L. H., Gholston, S., & Hughes, W. E. (2002). A semantic entropy metric. *Journal of Software Maintenance and Evolution*, *14*(4), 293–310. https://doi.org/10.1002/smr.255

Fægri, T. E. (2010). Adoption of Team Estimation in a Specialist Organizational Environment Adoption of team estimation in a specialist organizational environment. *Proc. - Agile Processes in Software Engineering and Extreme Programming*. https://doi.org/10.1007/978-3-642-13054-0

Fagerholm, F., Johnson, P., Guinea, A. S., Borenstein, J., & Munch, J. (2013). Onboarding in Open Source Software Projects: A Preliminary Analysis. *2013 IEEE 8th International Conference on Global Software Engineering Workshops*, 5–10. https://doi.org/10.1109/ICGSEW.2013.8

Fagerholm, F., Kuhrmann, M., & Münch, J. (2017). Guidelines for using empirical studies in software engineering education. *PeerJ Computer Science*, *3*, e131. https://doi.org/10.7717/peerj-cs.131

Fagerholm, F., & Pagels, M. (2014). Examining the structure of lean and agile values among software developers. *Lecture Notes in Business Info. Processing*, 218–233. https://doi.org/10.1007/978-3-319-06862-6

Fairley, R., & Willshire, M. J. (2011). Teaching Software Engineering to Undergraduate System Engineering Sudents. *Proc. - ASEE Annual Conference and Exposition*.

Faria, E. S. J., Yamanaka, K., & Tavares, J. A. (2012). EXtreme learning of programming - A methodology based in eXtreme programming to programming learning. *IEEE Latin America Transactions*, *10*(2), 1589–1594. https://doi.org/10.1109/TLA.2012.6187603

Ferdiana, R. (2016). Software engineering education learning process for professional developers. *Journal of E-Learning and Knowledge Society*, *12*(2), 71–83.

Fernandes, S., & Barbosa, L. S. (2016). Collaborative environments in software engineering teaching: A FLOSS approach. *Proceedings of the European Conference on E-Learning, ECEL*, *2016-Janua*, 201–206.

Festinger, L. (1950). Informal social communication. *Psychological Review*, *57*, 271–282.

Fisher, R., & Ury, W. (2011). *Getting to yes. Negotiating an agreement without giving in* (2nd ed.). Penguin Group.

Fisser, S., & Browaeys, M.-J. (2010). Team learning on the edge of chaos. *Learning Organization*, *17*(1), 58–68.

Fraenkel, J. R., & Wallen, N. E. (2009). *How to design and evaluate research in education* (7th Ed). McGraw-Hill Higher Education.

Frailey, D. J. (2006). Bringing Industrial Methods to the Classroom. *Proc. - 19th Conf. on Soft. Eng. Educ. and Training Workshops (CSEETW'06)*. https://doi.org/10.1109/CSEETW.2006.7

França, A. C. C., Fabio, Q. B., Felix, A. D. L. C., & Carneiro, D. E. S. (2014). Motivation in software engineering industrial practice: A cross-case analysis of two software organisations. *Information and Software Technology*, *56*(1), 79–101. https://doi.org/10.1016/j.infsof.2013.06.006

Frankl, G., Bitter, S., & Kaufmann, B. (2014). Win-for-all in software engineering education: Balancing social dilemmas to foster collaboration. *Proc. - 2014 IEEE 27th Conference on Software Engineering Education and Training*, 163–167. https://doi.org/10.1109/CSEET.2014.6816795

Freitas, D., Trindade, G., Fatima, T., & Tait, C. (2008). A tool for supporting the communication in distributed software development environment. *Journal of Computer, Science and Technology*, *8*(2).

Gabelica, C., Van den Bossche, P., Fiore, S. M., Segers, M., & Gijselaers, W. H. (2016). Establishing team knowledge coordination from a learning perspective. *Human Performance*, *29*(1), 33–53. https://doi.org/10.1080/08959285.2015.1120304

Gao, Y., Wang, F. Y., Sun, W., Dong, X., Liu, X., & Li, S. S. (2016). A CDIO-based social manufacturing laboratory: Prototype for CPSS-based production processes. *ASEE Annual Conference and Exposition, Conference Proceedings*, *2016-June*. https://doi.org/10.18260/p.26270

Garcia, I. A., & Pacheco, C. L. (2014). Using TSPi and PBL to support software engineering education in an upper-level undergraduate course. *Computer Applications in Engineering Education*, *22*(4), 736–749. https://doi.org/10.1002/cae.21566

Garry, L., & Change, S. (2013). Does group cohesion matter to decision quality in information systems development teams? *European, Mediterranean & Middle Eastern Conference on Information Systems 2013*.

Gary, K. A. (2008). The software enterprise: practicing best practices in software engineering education. *International Journal of Engineering Education*, *24*(4), 705–716. https://www.scopus.com/inward/record.uri?eid=2-s2.0-51549087646&partnerID=40&md5=d6f851f9c64269b0d6901a46e1906a13

Gates, A. Q., & Villa, E. Y. (2014). Developing Communities of Practice to Prepare Software Engineers With Effective Team Skills. In L. Yu (Ed.), *Overcoming challanges in software engineering education: delivering non-technical knowledge and skills* (pp. 52–70). IGI Global. https://doi.org/10.4018/978-1-5225-3923-0.ch073

Georgas, J. C. (2011). Teams Battling Teams: Introducing Software Engineering education in the first year with ROBOCODE. *ASEE Annual Conf. and Expo*.

Germain, É., & Robillard, P. N. (2008). Towards software process patterns: An empirical analysis of the behavior of student teams. *Information and Software Technology*, *50*(11), 1088–1097. https://doi.org/10.1016/j.infsof.2007.10.018

Ghani, I., Jawawi, D. N. A., Niknejad, N., Khan, M., & Jeong, S. R. (2016). A Survey of Agile Transition Models. In *Emerging Innovations in Agile Software Development* (Issue January, pp. 141–164). https://doi.org/10.4018/978-1-4666-9858-1.ch008

Ghobadi, S., Campbell, J., & Clegg, S. (2017). Pair programming teams and high-quality knowledge sharing: A comparative study of coopetitive reward structures. *Information Systems Frontiers*, *19*(2), 397–409. https://doi.org/10.1007/s10796-015-9603-0

Gómez, M., & Acuña, S. T. (2007). Study of the relationships between personality, satisfaction and product quality in software development teams. *19th Int. Conf. on Soft. Eng. & Knowledge Eng.*, 292–295.

Goncalves, G. S., Lima, G. L. B., Maria, R. E., Wisnieski, R. T., Dos Santos, M. V. M., Ferreira, M. A., Da Silva, A. C., Olimpio, A., Otero, A. G. L., De Vasconcelos, L. E. G., Sato, L. Y. C., Silva, H. N. A., Marques, J. C., Mattei, A. L. P., Da Cunha, A. M., Dias, L. A. V., & Saotome, O. (2015). An interdisciplinary academic project for spatial critical embedded system agile development. *AIAA/IEEE Digital Avionics Systems Conference - Proceedings*, 8C31-8C311. https://doi.org/10.1109/DASC.2015.7311475

Gonzalez, F. G., & Golf, F. (2015). FAST learning: Follow Accomplishments of Senior Teams Retention and Recruitment. *Porc. - ASEE Annual Conference and Exposition*.

Gotel, O., Kulkarni, V., Neak, L. C., Scharff, C., & Seng, S. (2007). Introducing Global Supply Chains into Software. *Software Engineering Approaches for Offshore and Outsourced Development*. https://doi.org/10.1007/978-3-540-75542-5

Gotel, O., Kulkarni, V., Say, M., Scharff, C., & Sunetnanta, T. (2009). A global and competition-based model for fostering technical and soft skills in software engineering education. *Proceedings - 22nd Conference on Software Engineering Education and Training, CSEET 2009*, 271–278. https://doi.org/10.1109/CSEET.2009.36

Gotel, O., Phal, D., & Say, M. (2009). *Evolving an Infrastructure for Student Global Software Development Projects: Lessons for Industry*. 117–126.

Gren, L. (2017). The Links Between Agile Practices, Interpersonal Conflict, and Perceived Productivity. *Proc. - ACM Int. Conf.*, 292–297. https://doi.org/10.1145/3084226.3084269

Gren, L., Torkar, R., & Feldt, R. (2017). Group development and group maturity when building agile teams: A qualitative and quantitative investigation at eight large companies. *Journal of Systems and Software*, *124*, 104–119. https://doi.org/10.1016/j.jss.2016.11.024

Gruenfeld, D. H., Martorana, P. V., & Fan, E. T. (2000). What do groups learn from their worldliest members? Direct and indirect influence in dynamic teams. *Organizational Behavior and Human Decision Processes*, *82*(1), 45–59.

Guetat, S., Ben, S., & Dakhli, D. (2011). Software Solutions Construction According to Information Systems Architecture Principles. In *CENTERIS 2011, Part II, CCIS 220* (pp. 408–417). Springer-Verlag Berlin Heidelberg.

Habib, M. (2016). *Supply chain management: Practices, applications and challenges*.

Hackman, J. R., & Katz, N. (2010). Group Behavior and Performance. In G. Fiske, S. T., Gilbert, D. T., & Lindzey (Ed.), *Handbook of social psychology* (5th ed.). Wiley. https://doi.org/10.1002/9780470561119.socpsy002032

Hadar, I., & Sherman, S. (2012). Agile vs. plan-driven perceptions of software architecture. *5th International Workshop on Co-Operative and Human Aspects of Software Engineering, CHASE 2012 - Proceedings*, 50–55. https://doi.org/10.1109/CHASE.2012.6223022

Hajou, A., Batenburg, R., & Jansen, S. (2014). How the pharmaceutical industry and agile software development methods conflict: A systematic literature review. *Proceedings - 14th International Conference on Computational Science and Its Applications, ICCSA 2014*, 40–48. https://doi.org/10.1109/ICCSA.2014.19

Hannay, J. E., & Benestad, H. C. (2010). Perceived productivity threats in large agile development projects. *Proc. - ESEM 2010*, 16–17. https://doi.org/10.1145/1852786.1852806

Harris, E. Y., Keller, D., Stachura, D., & Wolfe, G. (2017). Bioinformatics sister courses: An interdisciplinary collaborative learning framework to teach bioinformatics. *Proceedings - 2016 International Conference on Computational Science and Computational Intelligence, CSCI 2016*, 382–385. https://doi.org/10.1109/CSCI.2016.0079

Hayes, A. F. (2009). Beyond Baron and Kenny: Statistical Mediation Analysis in the New Millennium. *Communication Monographs*, *76*(4), 408-420. https://doi.org/10.1080/03637750903310360

Hayes, A. F. (2012). *PROCESS: A Versatile Computational Tool for Observed Variable Mediation, Moderation, and Conditional Process Modeling*. http://www.afhayes.com/%0Apublic/process2012.pdf

Hayes, A. F. (2013). *Introduction to Mediation, Moderation, and Conditional Process Analysis A Regression-Based Approach*. Guilford Publications.

Hazzan, O., & Dubinsky, Y. (2009). Reflection in software engineering education. *Proceeding of the 24th ACM SIGPLAN Conference Companion on Object Oriented Programming Systems Languages and Applications - OOPSLA '09*, 691. https://doi.org/10.1145/1639950.1639967

Hebig, R., & Wang, H. (2017). On tackling quality threats for the assessment of measurement programs: A case study on the distribution of metric usage and knowledge. *Science of Computer Programming*, *135*, 45–74. https://doi.org/10.1016/j.scico.2016.09.006

Heesen, H., Engelhardt, K., Reimer, F., Hammitt, J. K., Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., … Neill, J. (2002). Assessing Team Functioning in Engineering Education. *Proc. - ASEE Annual Conference*, *6*(2), 199–216. https://doi.org/10.1023/A:1018988827405

Heikkilä, V. T., & Lassenius, C. (2011). *Research plan – Planning product releases in global multi-team agile development projects*. *May 2014*. https://doi.org/10.1109/ICGSE-W.2011.29

Hemer, D. (2008). Peer Assessment of Group-based Software Engineering Projects. *19th Australian Software Engineering Conference, ASWEC (2008)*, 470–478. https://doi.org/10.1109/ASWEC.2008.8

Henry, J., Lande, D., & Tarbell, G. (2008). It can't get more real: Using multiple teams to develop multiple software products for multiple customers. *Proc. - 2008 21st IEEE-CS Conference on Software Engineering Education and Training Workshop,* *CSEETW'08*. https://doi.org/10.1109/CSEETW.2008.3

Henry, X. S. D., Mitra, M., Nagchaudhuri, A., & Zhang, L. (2016). Automated multiparameter water monitoring system as an experiential learning platform for undergraduate STEM majors. *Proc. - ASEE Annual Conference and Exposition, Conference Proceedings*, *2016-June*. https://doi.org/10.18260/p.26356

Hernandez, A. C. (2017). *Content and Temporal Analysis of Communications to Predict Task Cohesion in Software Development Global Teams*. University of North Texas.

Hernandez, A., Chen, P., Clemmons, C. C., & Dong, J. J. (2016). Addressing the learning needs of minority students in engineering through participatory design. *ASEE Annual Conference and Exposition, Conference Proceedings*, *2016-June*. https://doi.org/10.18260/p.26526

Hislop, G. (2006). Scaffolding Student Work in Capstone Design Courses. *Proc. - 36th Annual Conf. Frontiers in Educ.*, 1–4. https://doi.org/10.1109/FIE.2006.322630

Hoegl, M., & Gemuenden, H. G. (2001). Teamwork Quality and the Success of Innovative Projects: A Theoretical Concept and Empirical Evidence. *Organization Science*, *12*(4), 435–449. https://doi.org/10.1287/orsc.12.4.435.10635

Hoegl, M., Parboteeah, K. P., & Gemuenden, H. G. (2003). When teamwork really matters: Task innovativeness as a moderator of the teamwork-performance relationship in software development projects. *Journal of Engineering and Technology Management - JET-M*, *20*(4), 281–302. https://doi.org/10.1016/j.jengtecman.2003.08.001

Hoegl, M., & Proserpio, L. (2004). Team member proximity and teamwork in innovative projects. *Research Policy*, *33*(8), 1153–1165. https://doi.org/10.1016/j.respol.2004.06.005

Hohpe, G., Ozkaya, I., Zdun, U., & Zimmermann, O. (2016). The Software Architect 's Role in the Digital Age. *IEEE Software*, *33*(6), 30–39. https://doi.org/doi.ieeecomputersociety.org/10.1109/MS.2016.137

Holmes, R., Craig, M., & Stroulia, E. (2014). Lessons Learned Managing Distributed Software Engineering Courses. *Proc. - ICSE '14*.

Holt, G. (2005). Software Risk Management From a System Perspective. *C ROSS T ALK The Journal of Defense Software Engineering*, *February*.

Honig, W. L. (2008). Teaching successful "real-world" software engineering to the "Net" Generation: Process and quality win! *Software Engineering Education Conference, Proceedings*, 25–32. https://doi.org/10.1109/CSEET.2008.38

Hsu, M., Chen, I. Y., Chiu, C., & Ju, T. L. (2007). *Exploring the antecedents of team performance in collaborative learning of computer software* (Vol. 48, pp. 700–718). https://doi.org/10.1016/j.compedu.2005.04.018

Huang, P. H. P., Chen, M. C. M., & Chen, S. C. S. (2008). The Practice Training in the Software Engineering Education. *9th Int. Conf. for Young Computer Scientists*, 2636–2640. https://doi.org/10.1109/ICYCS.2008.443

Huang, S.-T., Cho, Y.-P., & Lin, Y.-J. (2006a). Exploring the Cognitive Apprenticeship Approach for Teaching Introductory Software Engineering. *Proc. - 19th Conference on Software Engineering Education and Training Workshops*.

Huang, S.-T., Cho, Y.-P., & Lin, Y.-J. (2006b). Implementation and Evaluation of Teaching an Introductory Software Engineering Course Framed in Cognitive Apprenticeship. *Proc. 3th Asia Pacific Soft. Eng. Conf. (APSEC'06)*, 477–484. https://doi.org/10.1109/APSEC.2006.39

Huen, W. H. (2007). Systems Engineering of Complex Software Systems. *Proc. - 37 Th ASEE/IEEE Frontiers in Education Conference*, 16–21.

Humphrey, W. S. (1987). *Characterizing the Software Process: A Maturity Framework*.

Hwang, D. J., & Blandford, D. K. (2000). A Multidisciplinary Team Project for Electrical Engineering , Computer Engineering , and Computer Science Majors. *ASEE Annual Conference Proceedings*, 339–345.

IEEE; ACM. (2014). *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*. https://doi.org/10.1145/2534860

IEEE. (1990). IEEE Standard Glossary of Software Engineering Terminology. In *IEEE Std 610.12- 1990* (Issue 1). https://doi.org/10.1109/IEEESTD.1990.101064

Iivari, J., & Iivari, N. (2011). The relationship between organizational culture and the deployment of agile methods. *Info. and Soft. Tech.*, *53*(5), 509–520. https://doi.org/10.1016/j.infsof.2010.10.008

Ilgen, D. R., Hollenbeck, J. R., Johnson, M., & Jundt, D. (2005). Teams in Organizations: From Input-Process-Output Models to IMOI Models. *Annual Review of Psychology*, *56*(1), 517–543. https://doi.org/10.1146/annurev.psych.56.091103.070250

Jaakkola, H., Henno, J., & Rudas, I. J. (2006). IT curriculum as a complex emerging process. *2006 IEEE International Conference on Computational Cybernetics, ICCC*. https://doi.org/10.1109/ICCCYB.2006.305731

Jamaludin, N. A. A., Sahibuddin, S., Hashim, M. N. H., & Shaharuddin, S. A. (2012). Development of Electronic Learning Industrial Environment (eLIN) System for Requirement Engineering Education. *Proc. - 3rd Int. Conf. on Computer Technology and Development*.

Jehn, K. A. (1995). A multimethod examination of the benefits and detriments of intragroup conflict. *Adm. Sci. Q*, *40*, 256–282.

Jeremic, Z., Jovanović, J., & Gasevic, D. (2009). Semantic web technologies for the integration of learning tools and context-aware educational services. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *5823 LNCS*, 860–875. https://doi.org/10.1007/978-3-642-04930-9_54

Jeremić, Z., Jovanović, J., & Gašević, D. (2011). An Environment for Project-based Collaborative Learning of Software Design Patterns. *International Journal on Engineering Education*, *27*(1), 41–51.

Johanyák, Z. C. (2016). Real-World Software Projects as Tools for the Improvement of Student Motivation and University- Industry Collaboration. *Proc. - Int. Conf. on Industrial Engineering, Management Science and Application*.

Jordan, A., Tomayko, J., & Gibbs, N. (2017). Hall of Fame Nomination : Studio-Based Master of Software Engineering Program at Carnegie Mellon University. *Proc. - 30th IEEE Conference on Software Engineering Education and Training*. https://doi.org/10.1109/CSEET.2017.10

Jovanovic, M., Mesquida, A. L., & Mas, A. (2015). Process improvement with retrospective gaming in agile software development. *Communications in Computer and Information Science*, *543*, 287–294. https://doi.org/10.1007/978-3-319-24647-5_23

Joy, M. (2005). Group projects and the computer science curriculum. *Innovations in Education and Teaching International*, *42*(1), 15–25. https://doi.org/10.1080/14703290500048788

Jun, H., & Lingli, H. (2010). Improving Undergraduates ' Software Engineering Ability by Adapting Blended Learning Model. *Int. Conf. on Artificial Intelligence and Educ.*, 521–524.

Kang, D., Jung, J., & Bae, D.-H. (2011). Constraint-based human resource allocation in software projects. *Software Practice and Experience*, *41*, 551–577. https://doi.org/10.1002/spe

Kantipudi, M., Collofello, J. S., Collier, K. W., & Medeiros, S. (2012). *Software engineering course projects: Failures and recommendations* (pp. 324–338). https://doi.org/10.1007/3-540-55963-9_60

Karn, J. S., Syed-Abdullah, S., Cowling, A. J., & Holcombe, M. (2007). A study into the effects of personality type and methodology on cohesion in software engineering teams. *Behaviour & Information Technology*, *26*(2), 99–111. https://doi.org/10.1080/01449290500102110

Kasl, E., Marsick, V., & Dechant, K. (1997). Teams as learners: A research-based model of team learning. *Journal of Applied Behavioral Science*, *33*(2), 227–246.

Keith, M., Demirkan, H., & Goul, M. (2017). The Role of Task Uncertainty in IT Project Team Advice Networks. *Decision Sciences*, *48*(2), 207–247. https://doi.org/10.1111/deci.12226

Kennedy, J. D., Australian, R., & Force, A. (2017). Innovation and certification in aviation software. *Proc. - ICNS 2017*, *April 2017*. https://doi.org/10.1109/ICNSURV.2017.8011916

Kessler, R., & Dykman, N. (2007). Integrating traditional and agile processes in the classroom. *ACM SIGCSE Bulletin*, *39*(1), 312. https://doi.org/10.1145/1227504.1227420

Khan, U. Z., Wahab, F., & Saeed, S. (2014). Integration of Scrum with Win-Win requirements negotiation model. *Middle - East Journal of Scientific Research*, *19*(1), 101–104. https://doi.org/10.5829/idosi.mejsr.2014.19.1.11770

Kidd, J. B., & Belbin, R. M. (2006). Management Teams - Why They Succeed or Fail. *The Journal of the Operational Research Society*, *33*(4), 392. https://doi.org/10.2307/2581652

Kilamo, T., Nieminen, A., Lautamäki, J., Aho, T., Koskinen, J., Palviainen, J., & Mikkonen, T. (2014). Code Knowledge Transfer in Collaborative Teams: Experiences from a Two-Week Code Camp. *Proc - ICSE Companion 2014*, *July 2016*. https://doi.org/10.1145/2591062.2591156

Kim, H.-K. (2006). A study on evaluation of component metric suites. *Computational Science and Its Applications*. https://doi.org/10.1007/3-540-68339-9_34

Kirkman, B. L., Rosen, B., Tesluk, P. E., & Gibson, C. B. (2004). The impact of team empowerment on virtual team performance: The moderating role of face-to-face interaction. *Academy of Management Journal*, *47*, 175–192.

Kisselburgh, L., Zoltowski, C., Beever, J., Hess, J., Krane, M., & Brightman, A. (2013). Using scaffolded, integrated, and reflexive analysis (SIRA) of cases in a cyber-enabled learning infrastructure to develop moral reasoning in engineering students. *Proceedings - Frontiers in Education Conference, FIE*, *March*, 1561–1563. https://doi.org/10.1109/FIE.2013.6685100

Kitchenham, B., Pearl Brereton, O., Budgen, D., Turner, M., Bailey, J., & Linkman, S. (2009). Systematic literature reviews in software engineering - A systematic literature review. *Information and Software Technology*, *51*(1), 7–15. https://doi.org/10.1016/j.infsof.2008.09.009

Kline, P. (2000). *The Handbook of Psychological Testing Second Edition* (Second). http://tocs.ub.uni-mainz.de/pdfs/084509694.pdf

KonterraGroup. (2015). *Empathic Listening – what is it?* konterragroup.net

Kozlowski, S. W. J., & Bell, B. S. (2012). Work Groups and Teams in Organizations. *Handbook of Psychology, Second Edition*. https://doi.org/10.1002/9781118133880.hop212017

Krivitsky, A. (2009). *Scrum Simulation with LEGO Bricks*. https://doi.org/https://www.scrumalliance.org/system/resource_files/0000/3689/Scrum-Simulation-with-LEGO-Bricks-v2.0.pdf

Krogstie, B. R. (2008). Power Through Brokering : Open Source Community Participation in Software Engineering Student Projects. *ICSE'08*, 791–800.

Kropp, M., Meier, A., & Perellano, G. (2016). Teaching agile collaboration skills in the classroom. *Proc. - IEEE 29th Conf on Soft Eng Edu and Training, CSEEandT 2016*, 118–127. https://doi.org/10.1109/CSEET.2016.30

Krutz, D. E., Malachowsky, S. A., Jones, S. D., & Kaplan, J. A. (2015). Enhancing the educational experience for deaf and hard of hearing students in software engineering. *Proceedings - Frontiers in Education Conference, FIE*, *2014*, 1–9. https://doi.org/10.1109/FIE.2015.7344327

Kumar, P., & Singh, S. K. (2016). A systematic assessment of aspect-oriented software development (AOSD) using JHotDraw application. *2016 International Conference on Computing, Communication and Automation (ICCCA)*, 779–784. https://doi.org/10.1109/CCAA.2016.7813840

Kuthyola, K. F., Liu, J. Y.-C., & Klein, G. (2017). Influence of Task Interdependence on Teamwork Quality and Project Performance. *Int. Conf. on Bussiness Information Systems*.

Lago, P., Muccini, H., & Babar, M. A. (2012). An empirical study of learning by osmosis in global software engineering. *Journal of Software: Evolution and Process*, *24*(6), 693–706. https://doi.org/10.1002/smr

Lakhanpal, B. (1993). Understanding the factors influencing the performance of software development groups: An exploratory group-level analysis. *Information and Software Technology*, *35*(8), 468–473. https://doi.org/10.1016/0950-5849(93)90044-4

Layman, L., & Williams, L. (2008). Addressing diverse needs through a balance of agile and plan-driven software development methodologies in the core software engineering course. *Int. Journal of Eng. Edu.*, *24*(4), 659–670. http://lucas.ezzoterik.com/papers/LWSB08.pdf

Lee, J., Liu, A., Cheng, Y. C., Ma, S. P., & Lee, S. J. (2012). Execution plan for software engineering education in Taiwan. In *Proceedings - Asia-Pacific Software Engineering Conference, APSEC* (Vol. 1, pp. 749–753). https://doi.org/10.1109/APSEC.2012.142

Lehmann-willenbrock, N. (2017). Team Learning: New Insights Through a Temporal Lens. *Small Group Research*, *48*(2), 123130. https://doi.org/10.1177/1046496416689308

Leitão, P., Colombo, A. W., & Restivo, F. (2003). An Approach to the Formal Specification of Holonic Control Systems. *Lecture Notes in Computer Science*, 59–70.

Lenberg, P., Feldt, R., & Wallgren, L. G. (2015). Behavioral software engineering: A definition and systematic literature review. *Journal of Systems and Software*, *107*, 15–37. https://doi.org/10.1016/j.jss.2015.04.084

Lenberg, P., Wallgren Tengberg, L. G., & Feldt, R. (2017). An initial analysis of software engineers' attitudes towards organizational change. *Empirical Software Engineering*, *22*(4), 2179–2205. https://doi.org/10.1007/s10664-016-9482-0

Lencioni, P. (2005). *Overcoming the Five Dysfunctions of a Team. A Field guide*. Jossey-Bass. Wiley Imprint.

Lewis, T. L., & Smith, W. J. (2008). Creating high performing software engineering teams: the impact of problem solving style dominance on group conflict and performance. *Proc. - 38th ASEE/IEEE Frontiers in Education Conference*. https://doi.org/10.1109/FIE.2008.4720498

Liang, J. S. (2012). Learning in troubleshooting of automotive braking system. *BJET*, *4*(3), 331–352.

Licorish, S. A., & Macdonell, S. G. (2015). Communication and personality profiles of global software developers. *Info. and Soft. Tech.*, *64*, 113–131. https://doi.org/10.1016/j.infsof.2015.02.004

Licorish, S. A., & Macdonell, S. G. (2013). What can developers' messages tell us? A psycholinguistic analysis of Jazz teams' attitudes and behavior patterns. *Proceedings of the Australian Software Engineering Conference, ASWEC*, 107–116. https://doi.org/10.1109/ASWEC.2013.22

Licorish, S. A., Macdonell, S. G., & Software, A. (2013). *How Do Globally Distributed Agile Teams Self-organise? - Initial Insights from a Case Study*. 157–164. https://doi.org/10.5220/0004437001570164

Licorish, S., Philpott, A., & MacDonell, S. G. (2009). Supporting agile team composition: A prototype tool for identifying personality (in)compatibilities. *Proc.- CHASE 2009*, 66–73. https://doi.org/10.1109/CHASE.2009.5071413

Liebenberg, L., & Mathews, E. H. (2012). Integrating innovation skills in an introductory engineering design-build course. *International Journal of Technology and Design Education*, *22*(1), 93–113. https://doi.org/10.1007/s10798-010-9137-1

Lieberman, M. A., Yalom, I. D., & Miles, M. B. (1973). *Encounter Groups: First Fact* (Basic book). Basic book, Inc.

Lindsjørn, Sjøberg, D. I. K., Dingsøyr, T., Bergersen, G. R., & Dybå, T. (2016). Teamwork quality and project success in software development: A survey of agile development teams. *The Journal of Systems and Software*, *122*, 274–286. https://doi.org/10.1016/j.jss.2016.09.028

Liu, W. H., & Cross, J. A. (2016). A comprehensive model of project team technical performance. *International Journal of Project Management*, *34*(7), 1150–1166. https://doi.org/10.1016/j.ijproman.2016.05.011

Long, J. (2010). Outsourcing in Next Generation Software Engineering Technology Education. *Proc.- ASEE Anual Conf. & Exposition*.

Lopez-Martin, C. (2008). Quality improvement applying design and code reviews for developing small programs. *IMETI 2008 - International Multi-Conference on Engineering and Technological Innovation, Proceedings*, *2*, 153–158. http://www.scopus.com/inward/record.url?eid=2-s2.0-84893155766&partnerID=tZOtx3y1

Lopez-Martin, C., Cornelio Yañez-Marquez, & Gutierrez-Tornes, A. (2008). Predictive accuracy comparison of fuzzy models for software development effort of small programs. *Journal of Systems and Software*, *81*(6), 949–960. https://doi.org/10.1016/j.jss.2007.08.027

Lopez-Nores, M., Pazos-Arias, J. J., Garcia-Duque, J., Blanco-Fernandez, Y., Diaz-Redondo, R. P., Fernandez-Vilas, A., Gil-Solla, A., & Ramos-Cabrer, M. (2009). Procedures and algorithms for continuous integration in an agile specification environment. *International Journal of Software Engineering and Knowledge Engineering*, *19*(1).

López-Querol, S., Sánchez-Cambronero, S., Rivas, A., & Garmendia, M. (2015). Improving civil engineering education: Transportation geotechnics taught through project-based learning methodologies. *Journal of Professional Issues in Engineering Education and Practice*, *141*(1), 1–7. https://doi.org/10.1061/(ASCE)EI.1943-5541.0000212

Loughry, M. L., Ohland, M. W., & Dewayne Moore, D. (2007). Development of a theory-based assessment of team member effectiveness. *Educational and Psychological Measurement*, *67*(3), 505–524. https://doi.org/10.1177/0013164406292085

Loughry, M. L., Ohland, M. W., & Woehr, D. J. (2014). Assessing Teamwork Skills for Assurance of Learning Using CATME Team Tools. *Journal of Marketing Education*, *36*(1), 5–19. https://doi.org/10.1177/0273475313499023

Ludi, S. (2006). Work in progress: Effectiveness of collaboration within a secure software engineering course for SE and computing students. *Frontiers in Education Conference, FIE*, 15–16. https://doi.org/10.1109/FIE.2006.322304

Lutz, M. J., Naveda, J. F., & Vallino, J. R. (2014). Undergraduate Software Engineering. *Commun. ACM*, *57*(8), 52–58. https://doi.org/10.1145/2632361

Lynch, T. D., Herold, M., Bolinger, J., Deshpande, S., Bihari, T., & Ramanathan, J. (2011a). An Agile Boot Camp : Using a LEGO ® -Based Active Game to Ground Agile Development. *Proc- Frontiers in Educ. Conf.*, 1–6.

Lynch, T. D., Herold, M., Bolinger, J., Deshpande, S., Bihari, T., & Ramanathan, J. (2011b). An Agile Boot Camp : Using a LEGO ® -Based Active Game to Ground Agile Development Principles . *Frontiers in Edu. Conf.*, *October*.

https://doi.org/10.1109/FIE.2011.6142849

Lyytinen, K., & Rose, G. M. (2005). How Agile is Agile Enough ? Towards A Theory of Agility in Software Development. *Business Agility and Information Technology Diffusion*, *May 2014*. https://doi.org/10.1007/0-387-25590-7

Mackinnon, D. P., Lockwood, C. M., Hoffman, J. M., & West, S. G. (2002). A Comparison of Methods to Test Mediation and Other Intervening Variable Effects. *Psychol Methods*, *7*(1).

Magni, M., Proserpio, L., Hoegl, M., & Provera, B. (2009). The role of team behavioral integration and cohesion in shaping individual improvisation. *Research Policy*, *38*(6), 1044–1053. https://doi.org/10.1016/j.respol.2009.03.004

Mahnic, V., & Casar, A. (2016). A Computerized Support Tool for Conducting a Scrum-Based Software Engineering Capstone Course. *The International Journal of Engineering*, *32*(1), 278–293.

Mahnic, Viljan. (2010). Teaching Scrum through Team-Project Work : Students' Perceptions and Teacher's Observations. *International Journal of Engineering Education*, *26*(February), 96–110.

Mahnic, Viljan. (2012). A capstone course on agile software development using scrum. In *IEEE Transactions on Education* (Vol. 55, Issue 1, pp. 99–106). https://doi.org/10.1109/TE.2011.2142311

Mahnic, Viljan. (2015a). From Scrum to Kanban: Introducing Lean Principles to a Software Engineering Capstone Course. *International Journal of Engineering Education*, *31*(4), 1106–1116.

Mahnic, Viljan. (2015b). Scrum in software engineering courses: An outline of the literature. *Global Journal of Engineering Education*, *17*(2), 77–83.

Malheiro, B., Silva, M. F., Ferreira, P. D., & Guedes, P. B. (2019). Learning Engineering with EPS@ISEP: Developing Projects for Smart Sustainable Cities. *International Journal of Engineering Pedagogy (IJEP)*, *9*(4), 33. https://doi.org/10.3991/ijep.v9i4.10259

Mamei, A., Todtenhoefer, R., & Petkovic, D. (2011). Work in progress—Elassys: Online tool for teamwork analysis and assessment in software engineering education. *Frontiers in Education …*, 11–13. https://doi.org/10.1109/FIE.2011.6142842

Manamendra, M. A. S. C., Manathunga, K. N., Perera, K. H. D., & Kodagoda, N. (2013). Improvements for agile manifesto and make agile applicable for undergraduate research projects. In *Proceedings of the 8th International Conference on Computer Science and Education, ICCSE 2013* (pp. 539–544). https://doi.org/10.1109/ICCSE.2013.6553969

Marks, M. A., Mathieu, J. E., & Zaccaro, S. (2001). A Temporally Based Framework and Taxonomy of Team Processes. *Academy of Management Review*, *26*(3), 356–376. https://doi.org/10.5465/AMR.2001.4845785

Marquardt, M. J. (2002). *Building the learning organization* (2nd ed.). Davies-Black Publishing.

Marques, M. (2015). Software engineering education — Does gender matter in project results? — A Chilean case study. *2015 IEEE Frontiers in Education Conference (FIE)*, *2015-Decem*, 1–8. https://doi.org/10.1109/FIE.2015.7344175

Martin, Angela, Davies, R., & Eckstein, J. (2006). Politics and Religion in Agile Development. *EXTREME PROGRAMMING AND AGILE PROCESSES IN SOFTWARE ENGINEERING, PROCEEDINGS*. https://doi.org/10.1007/11774129

Martin, Antonio, Pareto, L., & Bosch, J. (2013). Improving Businesses Success by Managing Interactions among Agile Teams in Large Organizations. *Proc. - ICSOB 2013*.

Massa, N. M., Masciadrelli, G. J., & Mullett, G. J. (2005). Re-engineering technician education for the new millennium. *ASEE Annual Conference and Exposition,* *Conference Proceedings*, 12005–12019.

Masson, P., & Udas, K. (2009). An agile approach to managing open educational resources. *On the Horizon*, *17*(3), 256–266. https://doi.org/10.1108/10748120910993286

Mathieu, J., Maynard, M. T., Rapp, T., & Gilson, L. (2008). Team Effectiveness 1997-2007: A Review of Recent Advancements and a Glimpse Into the Future. *Journal of Management*, *34*(3), 410–476. https://doi.org/10.1177/0149206308316061

Maxwell, S. E., Cole, D. A., Mitchell, M. A., Maxwell, S. E., Cole, D. A., Bias, M. A. M., Maxwell, S. E., Cole, D. A., & Mitchell, M. A. (2011). Bias in Cross-Sectional Analyses of Longitudinal Mediation: Partial and Complete Mediation Under an Autoregressive Model Bias in Cross-Sectional Analyses of Longitudinal Mediation: Partial and Complete Mediation Under an Autoregressive Model. *Multivariate Behavioral Research ISSN:*, *46*(5), 816–841. https://doi.org/10.1080/00273171.2011.606716

Maznevski, M. L., Chudoba, K. M., & Robey, D. (2000). Bridging Space Over Time: Global Virtual Team Dynamics and Effectiveness. *Organization Science*, *11*(5), 473–492.

Mazni, O., Norliza, K., Sharifah Lailee, S.-A., Nor Laily, H., & Rohaida, R. (2015). Assessing personality types preferences amongst software developers: A case of Malaysia. *ARPN Journal of Engineering and Applied Sciences*, *10*(3), 1499–1504.

Mazni, O., Sharifah-Lailee, S.-A., & Azman, Y. (2010). Agile Documents: Toward Successful Creation of Effective Documentation. *Proc. - XP 2010*, 196–201.

Mcavoy, J., & Butler, T. (2009). A Failure to Learn in a Software Development Team: The Unsuccessful Introduction of an Agile Method. *Information Systems Development: Challenges in Practice, Theory, and Education*, *1*. https://doi.org/10.1007/978-0-387-68772-8_1

McAvoy, J., & Butler, T. (2005). A paradox of virtual teams and change: An implementation of the theory of competing commitments. *International Journal of ECollaboration*, *2*(3), 1–24. https://doi.org/http://dx.doi.org/10.4018/jec.2006070101

Mcgrath, J. E. (1984). *Groups: Interaction and Performance*. Prentice-Hall.

Mclean, C. R., & Jain, S. (2007). A Virtual Manufacturing Environment for Interoperability Testing. *5th International Industrial Simulation Conference*, 331–337.

McMahon, P. E. (2004). Bridging agile and traditional development methods: A project management perspective. *CrossTalk*, *4*, 16–20. https://doi.org/10.1109/9780470049167.ch2

Mead, P. F., Natishan, M., Schmidt, L., Greenberg, J., Bigio, D., Gupte, A., & Park, C. (2000). Engineering Project Team Training System (EPTTS) For Effective Engineering Team Management. *ASEE Annual Conference Proceedings*, 2497–2506.

Membrillo-Hernández, J., J. Ramírez-Cadena, M., Martínez-Acosta, M., Cruz-Gómez, E., Muñoz-Díaz, E., & Elizalde, H. (2019). Challenge based learning: the importance of world-leading companies as training partners. *International Journal on Interactive Design and Manufacturing*, *13*(3), 1103–1113. https://doi.org/10.1007/s12008-019-00569-4

Mendonça, D., Brooks, J. D., & Grabowski, M. (2014). Linking team composition to team performance: An application to postdisaster debris removal operations. *IEEE Transactions on Human-Machine Systems*, *44*(3), 315–325. https://doi.org/10.1109/THMS.2014.2306198

Miglierina, M. (2015). Application deployment and management in the cloud. *16th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, SYNASC 2014*, *Section IV*, 422–428. https://doi.org/10.1109/SYNASC.2014.63

Minocha, S., Petre, M., & Roberts, D. (2008). Using wikis to simulate distributed requirements development in a software engineering course - Open Research Online. *International Journal of Engineering Education*, *24*(4), 689–704.

Moe, N. B. (2013). Key challenges of improving agile teamwork. In H. Baumeister & B.Weber (Eds.), *Agile Proc. in Soft. Eng. and Extreme Programming. XP 2013. Lecture Notes in Business Information Processing* (Vol. 149, pp. 76–90). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-38314-4_6

Moe, N. B., Dingsøyr, T., & Dybå, T. (2009). Overcoming Barriers to Self-Management in Software Teams. *IEEE Software*, *26*(6), 20–26. https://doi.org/10.1109/MS.2009.182

Moe, N. B., Faegri, T. E., Cruzes, D. S., & Faugstad, J. E. (2016). Enabling knowledge sharing in agile virtual teams. *11th IEEE International Conference on Global Software Engineering, ICGSE 2016*, 29–33. https://doi.org/10.1109/ICGSE.2016.30

Molleman, E., & Beukel, A. van den. (2007). Worker Flexibility and Its Perceived Contribution to Performance: The Moderating Role. *Human Factors and Ergonomics in Manufacturing*, *17*(2), 117–135. https://doi.org/10.1002/hfm.20069

Monaghan, C., Bizumic, B., Reynolds, K., Smithson, M., Johns-Boast, L., & van Rooy, D. (2015). Performance of student software development teams: the influence of personality and identifying as team members. *European Journal of Engineering Education*, *40*(1), 52–67. https://doi.org/10.1080/03043797.2014.914156

Monasor, M. J., Vizcaíno, A., & Piattini, M. (2010). An educational environment for training skills for global software development. *Proc. - 10th IEEE Int. Conf. on Advanced Learning Technologies, ICALT 2010*, 99–101. https://doi.org/10.1109/ICALT.2010.35

Moyne, M. M., Herman, M., Walsh, C., & Holland, D. P. (2018). IUSE: A web-based tool for engineering design pedagogy research. *ASEE Annual Conference and Exposition, Conference Proceedings*, *2018-June*.

Mujkanovic, A., & Bollin, A. (2016). Improving learning outcomes through systematic group reformation. *Proc. - 9th International Workshop on Cooperative and Human Aspects of Soft. Eng. - CHASE '16*, 97–103. https://doi.org/10.1145/2897586.2897615

Mukhopadhyay, D. (2010). A practitioner's approach to software system design. *International Symposium on Electronic System Design, ISED 2010*, 20–23. https://doi.org/10.1109/ISED.2010.13

Mullen, B., & Cooper, C. (1994). The relation between group cohesiveness and performance: an integration. *Psychological Bulletin*, *115*(2), 210–227.

Mullen, B., & Copper, C. (1994). The relation between group cohesiveness and performance. *Psych. Bull.*, *115*(2), 210–227.

Nagchaudhuri, A. (2001). Introduction of mechatronics in pre-college programs and freshman design course in an active and cooperative learning framework. *Proceedings - Frontiers in Education Conference*, *3*, 17–22. https://doi.org/10.1109/fie.2001.964030

Nandigam, J., Gudivada, V. N., & Hamou-Lhadj, A. (2008). Learning software engineering principles using open source software. *Proc. - 38th Annual Frontiers in Educ. Conf.* https://doi.org/10.1109/FIE.2008.4720643

Neill, C. J., Defranco, J. F., Sangwan, R. S., Neill, C. J., Defranco, J. F., Improving, R. S. S., Neill, C. J., Defranco, J. F., & Sangwan, R. S. (2017). *Improving collaborative learning in online software engineering education education*. *3797*. https://doi.org/10.1080/03043797.2016.1203293

Nguyen, D. M., Truong, T. V., & Le, N. B. (2013). Deployment of capstone projects in software engineering education at duy tan university as part of a university-wide project-based learning effort. *Proceedings - 2013 Learning and Teaching in Computing and Engineering, LaTiCE 2013*, 184–191. https://doi.org/10.1109/LaTiCE.2013.27

Nizami, K. (2007). Global Software Development and Delivery. *Dr. Dobb's Journal: The World of Software Development*, *32*(8), 22–28. http://ezproxy.lib.ucf.edu/login?URL=http://search.ebscohost.com/login.aspx?direct=true&db=buh&AN=25734581&site=ehost-live

O'Leary, R., & Bingham, L. B. (2007). *A Manager ' s Guide to Resolving Conflicts in Collaborative Networks*.

O'Reilly, C; Morrow, P; Bustard, D. (2003). Improving conflict detection in optimistic concurrency control models. In *Soft. Conf. Manag.* Springer-Verlag Berlin Heidelberg.

Ohland, M. W., Carolina, N., Bullard, L. G., Felder, R. M., & Layton, R. A. (2012). Assessment of Team Member Effectiveness : Development of a Behaviorally Anchored Rating Scale for Self- and Peer Evaluation University of Michigan. *Academy of Management Learning & Education*, *11*(4), 609–630.

Ohland, M. W., Loughry, M. L., Woehr, D. J., Bullard, L. G., Felder, R. M., Finelli, C. J., Layton, R. A., Pomeranz, H. R., & Schmucker, D. G. (2012). The Comprehensive Assessment of Team Member Effectiveness: Development of a Behaviorally Anchored Rating Scale for Self- and Peer Evaluation. *Academy of Management Learning and Education*, *11*(4), 609–631. https://doi.org/10.5465/amle.2010.0177

Okudan, G. E., Horner, D., Bogue, B., & Devon, R. (2002). An investigation of gender composition on integrated project team performance: Part III. *ASEE Annual Conference Proceedings*, 7337–7344.

Olsson, H. H., Bosch, J., & Alahyari, H. (2013). Customer-specific teams for agile evolution of large-scale embedded systems. *Proceedings - 39th Euromicro Conference Series on Software Engineering and Advanced Applications, SEAA 2013*, 82–89. https://doi.org/10.1109/SEAA.2013.43

Omar, M., & Khasasi, N. (2017). Designing an agile Scrum team formation model. *Conf. on Inf. Systems 2017*, 145–154.

Omar, Mazni, & Syed-Abdullah, S.-L. (2010). Identifying Effective Software Engineering (SE) Team Personality Types Composition using Rough Set Approach. *Int. Symposium on Inf. Tech.*, 1499–1503.

Oni, O., & Letier, E. (2016). Optimizing the Incremental Delivery of Software Features Under Uncertainty. *Proc. - Requirements Engineering: Foundation for Software Quality, REFSQ 2016*. https://doi.org/10.1109/MS.2011.81

Onions, P., & Patel, C. (2009). Enterprise SoBA: Large-scale implementation of acceptance test driven story cards. *Proc. - IEEE International Conference on Information Reuse and Integration, IRI 2009*, 105–109. https://doi.org/10.1109/IRI.2009.5211600

Onwuegbuzie, A. J. (2003). Expanding the Framework of Internal and External Validity in Quantitative Research. *Research in the Schools*, *10*(1), 71–78.

Ozawa, H., & Zhang, L. (2013). Adapting agile methodology to overcome social differences in project members. *Proceedings - AGILE 2013*, 82–87. https://doi.org/10.1109/AGILE.2013.13

Paasivaara, M., Heikkilä, V., Lassenius, C., & Toivola, T. (2014). Teaching students scrum using LEGO blocks. *ICSE Companion'14*, 382–391. https://doi.org/10.1145/2591062.2591169

Paasivaara, M., Vanhanen, J., Heikkila, V. T., Lassenius, C., Itkonen, J., & Laukkanen, E. (2017). Do high and low performing student teams use scrum differently in capstone projects? *Proc.- ICSE-SEET 2017*, 146–149. https://doi.org/10.1109/ICSE-SEET.2017.22

Papatheocharous, E., Belk, M., Nyfjord, J., Germanakos, P., & Samaras, G. (2014). Personalised continuous software engineering. *Proc. - RCoSE'14*, 57–62. https://doi.org/10.1145/2593812.2593815

Park, S., & Maurer, F. (2010). A network analysis of stakeholders in tool visioning process for story test driven development. *Proceedings of the IEEE International Conference on Engineering of Complex Computer Systems, ICECCS*, 205–214. https://doi.org/10.1109/ICECCS.2010.29

Parsons, D. ., Lal, R. ., Ryu, H. ., & Lange, M. . (2007). Software development methodologies, agile development and usability engineering. *Proc. - 18th Australasian Conference on Information Systems*, 172–178. https://doi.org/10.1016/S0003-9993(03)00289-2

Patel, C., & Ramachandran, M. (2009). Best practices guidelines for agile requirements engineering practices. In *Implications, Handbook of Research on Software Engineering and Productivity Technologies: of Globalization* (pp. 1–14).

Patil, P., Shettar, P., Akki, M., Sunag, B., & Meena, S. M. (2016). Web technologies integrated with advance database management system: A laboratory experience. *Proceedings of the 2015 IEEE 3rd International Conference on MOOCs, Innovation and Technology in Education, MITE 2015*, 301–305. https://doi.org/10.1109/MITE.2015.7375334

Pazos, P. (2012). Conflict management and effectiveness in virtual teams. *Team Performance Management: An International Journal*, *18*(7/8), 401–417. https://doi.org/10.1108/13527591211281138

Pechau, J. (2011). Conflicting value systems in agile software development projects. *18th Conf. on Pattern Languages of Programs*. https://doi.org/10.1145/2578903.2579160

Pechau, J. (2012). Rafting the agile waterfall. *Proc. - EuroPLoP '11*, 1–15. https://doi.org/10.1145/2396716.2396731

Péraire, C., & Sedano, T. (2014). State-based monitoring and goal-driven project steering: field study of the SEMAT essence framework. *ICSE'14 Proc.*, 325–334. https://doi.org/10.1145/2591062.2591155

Petkovic, D., Thompson, G., Todtenhoefer, R., Huang, S., Levine, B., Parab, S., Singh, G., Soni, R., & Shrestha, S. (2010). Work in progress - E-TAT: Online tool for teamwork and "Soft skills" assessment in software engineering education. *Proc. - 40th ASEE/IEEE Frontiers in Education Conference*, S1G-1-S1G-3. https://doi.org/10.1109/FIE.2010.5673130

Petkovic, Dragutin, Sosnick, M., & Todtenhoefer, R. (2012). Work i n Progress : A Machine Learning Approach for Assessment and Prediction of Teamwork Effectiveness in Software Engineering Education. *Frontiers in Education Conference*, *i*, 1–3.

Petkovic, Dragutin, Todtenhoefer, R., & Thompson, G. (2006). Teaching Practical Software Engineering and Global Software Engineering : Case Study and Recommendations. *36th ASEE/IEEE Frontiers in Educ. Conf.*, 19–24.

Pfleeger, S. L., & Atlee, J. M. (2005). Capturing the requirements. In *Software engineering theory and practice*. Prentice Hall.

Plechawska-Wojcik, M., & Borys, M. (2012). Methods and technologies for quality improving of student team software projects. In *IEEE Global Engineering Education Conference, EDUCON*. https://doi.org/10.1109/EDUCON.2012.6201100

Plesa, S., & Prostean, G. (2018). Business Process Management for Model Based Design Automotive Projects. *Proc. - 14th International Symposium in Management*. https://doi.org/10.1016/j.sbspro.2018.04.007

Pollard, J. (2003). Problem centred learningto-research. *Proc. - 33rd ASEEIIEEE Frontiers in Education Conference*, 21–24.

Powers, T. A., Sims-Knight, J., Topciu, R. A., & Haden, S. C. (2002). Assessing Team Functioning in Engineering Education. *Proc. Amer. Soc. Eng. Educ. Annu. Conf. Exposit.*, 199–216.

Poženel, M., & Mahnič, V. (2016). Studying agile software estimation techniques: the design of an empirical study with students. *Global Journal of Engineering Education*, *18*(2), 53–58.

Practical Application: Conflict Resolution Scenario. (2016). https://study.com/academy/lesson/conflict-resolution-scenario-application.html.

Preacher, K. J., Rucker, D. D., & Hayes, A. F. (2007). Addressing moderated mediation hypotheses: Theory, methods, and prescriptions. *Multivariate Behavioral Research*, *42*(1), 185–227. https://doi.org/10.1080/00273170701341316

Raes, E., Kyndt, E., Decuyper, S., Van den Bossche, P., & Dochy, F. (2015). An exploratory study of group development and team learning. *Human Resource Development Quarterly*, *26*(1), 5–30.

Ragan, E. D., Frezza, S., & Cannell, J. (2009). Product-based Learning in Software Engineering Education. *39th IEEE Int. Conf. on Frontiers in Education Conference*, 524–529. https://doi.org/10.1109/FIE.2009.5350648

Rajala, B. S. A. (2013). Beyond 2020 : Preparing Engineers for the Future. *Proc. IEEE*, *100*.

Rajendran, S., Gary, K., & Koehnemann, H. (2012). A Tool for Teaching Risk. *2012 Ninth International Conference on Information Technology - New Generations*, 349–354. https://doi.org/10.1109/ITNG.2012.172

Ralph, P., & Shportun, P. (2013). Scrum Abandonment in Distributed Teams: A Revelatory Case. *Pacific-Asia Conference on Information Systems (PACIS)*. http://aisel.aisnet.org/cgi/viewcontent.cgi?article=1041&context=pacis2013

Ramakrishnan, S., & Cambrell, A. (2004). Service based framework for knowledge portals. In *SIGCSE Bulletin (Association for Computing Machinery, Special Interest Group on Computer Science Education)* (Vol. 36, Issue 3, p. 260). https://doi.org/10.1145/1026487.1008093

Ramesh, B., Cao, L., Kim, J., Mohan, K., James, T. L., Ramesh, B., Cao, L., Kim, J., Mohan, K., & James, T. L. (2017). Conflicts and complements between eastern cultures and agile methods: an empirical investigation. *European Journal of Information Systems*, *26*(2), 206–235. https://doi.org/10.1057/s41303-016-0023-0

Ramesh, B., Mohan, K., & Cao, L. (2012). Ambidexterity in agile distributed development: An empirical investigation. *Information Systems Research*, *23*(2), 323–339. https://doi.org/10.1287/isre.1110.0351

Ramingwong, S., & Ramingwong, L. (2015). Plasticine Scrum: An Alternative Solution for Simulating Scrum Software Development. In K. J. Kim (Ed.), *Lecture Notes in Electrical Engineering (LNEE)*. Springer-Verlag Berlin Heidelberg 2015.

Rassias, N., & Kirytopoulos, K. (2014). Evaluating risk factors in the operation of virtual teams in ICT projects. *Proc. - IEEE International Conference on Industrial Engineering and Engineering Management*, 1192–1197. https://doi.org/10.1109/IEEM.2014.7058827

Razali, N. M., & Wah, Y. B. (2011). Power comparisons of Shapiro-Wilk, Kolmogorov-Smirnov, Lilliefors, and Anderson-Darling tests. *J. Stat. Model. Anal.*, *2*(1), 21–33.

Ribaud, V., & Saliou, P. (2009). Revealing software engineering theory-in-use through the observation of software engineering apprentices' course-of-action. *4th International Multi-Conference on Computing in the Global Information Technology, ICCGI 2009*, 202–210. https://doi.org/10.1109/ICCGI.2009.37

Ribaud, V., Saliou, P., Brest, U. De, & Cedex, B. (2008). A project-based immersion system. *Proc. - CSEET 2008 - Workshop*. https://doi.org/10.1109/CSEETW.2008.1

Richardson, I., Milewski, A. E., Keil, P., & Mullick, N. (2006). *Distributed Development – an Education Perspective on the Global Studio Project*. 679–684.

Rico, D. F., & Sayani, H. H. (2009). Use of agile methods in software engineering education. *Proceedings - 2009 Agile Conference, AGILE 2009*. https://doi.org/10.1109/AGILE.2009.13

Rodin, R., Leet, J., Azua, M., & Bygrave, D. (2011). A pattern language for release and deployment management. *Proc. - EuroPLoP '11*. https://doi.org/10.1145/2578903.2579147

Rodriguez, G., Soria, A., & Campo, M. (2015). Virtual Scrum: A teaching aid to introduce undergraduate software engineering students to Scrum. *Computer Applications in Engineering Education*, *23*(1), 147–156. https://doi.org/10.1002/cae.21588

Rodríguez, M. J., Parets, J., Paderewski, P., Anaya, A., & Hurtado, M. V. (1999). HEDES: A System Theory Based Tool to Support Evolutionary Software Systems. *Computer Aided Systems Theory*, 450–464. https://doi.org/10.1007/10720123_39

Rombach, D., Münch, J., Ocampo, A., Humphrey, W. S., & Burton, D. (2008). Teaching disciplined software development. *Journal of Systems and Software*, *81*(5), 747–763. https://doi.org/10.1016/j.jss.2007.06.004

Root, D., Rosso-Llopart, M., & Taran, G. (2008). Proposal based studio projects: How to avoid producing "cookie cutter" software engineers. *Proc. - 21st Conference on Software Engineering Education and Training*, 145–151. https://doi.org/10.1109/CSEET.2008.20

Rothenberger, M. A., Srite, M., & Jones-Graham, K. (2010). The impact of project team attributes on ERP system implementations: A positivist field investigation. In *Information Technology and People* (Vol. 23, Issue 1). https://doi.org/10.1108/09593841011022555

Rusnjak, A., Kharbili, M. El, Hristov, H., & Speck, A. (2010). Managing the dynamics of e/mCommerce with a hierarchical overlapping Business-Value-Framework. *Proc. - 24th IEEE International Conference on Advanced Information Networking and Applications Workshops, WAINA 2010*, 461–466. https://doi.org/10.1109/WAINA.2010.23

Rusu, Adrian, Rusu, A., Docimo, R., Santiago, C., & Paglione, M. (2009). Academia-academia-industry collaborations on software engineering projects using local-remote teams. *ACM SIGCSE Bulletin*, *41*(1), 301. https://doi.org/10.1145/1539024.1508975

Rusu, Amalia, & Gowda, S. (2011). A comparative study of academic partnerships from a student perspective. *Proceedings - Frontiers in Education Conference, FIE*. https://doi.org/10.1109/FIE.2011.6142843

Rusu, Amalia, & Swenson, M. (2008). An Industry-Academia Team-Teaching Case Study for Software Engineering Capstone Courses. *Proc. - Frontiers in Education Conference, FIE*, 18–23.

Rutz, L., & Tanner, M. (2016). Factors that influence performance in Global Virtual Teams in outsourced software development projects. *2016 IEEE International Conference on Emerging Technologies and Innovative Business Practices for the Transformation of Societies, EmergiTech 2016*, 329–335. https://doi.org/10.1109/EmergiTech.2016.7737361

S.D. Teasley ; L.A. Covi ; M.S. Krishnan ; J.S. Olson. (2002). Rapid software development through team collocation. *IEEE Transactions on Software Engineering*, *28*(7), 671–683.

Sachdeva, V., & Chung, L. (2017). Handling non-functional requirements for big data and IOT projects in Scrum. *Proc. - Confluence 2017*, 216–221. https://doi.org/10.1109/CONFLUENCE.2017.7943152

Sadun, C. (2010). Scrum and Global Delivery: Pitfalls and Lessons Learned. *Agility Across Time and Space*, 71–89. https://doi.org/10.1007/978-3-642-12442-6

Salado, A., Morelock, J. R., & Lakeh, A. B. (2017). Decision-making, information seeking, and compromise: A simulation game activity in global industrial management. *ASEE Annual Conference and Exposition, Conference Proceedings*, *2017-June*(June). https://doi.org/10.18260/1-2--28100

Salameh, H., & Alnaji, L. (2014). Challenges Leading to Projects Struggle in IT Project Management Office. *WSEAS Transactions on Business and Economics*, *11*(1), 262–271.

Salas, E. (2005). Is there a "Big Five" in Teamwork? *Small Group Research*, *36*(5), 555–599. https://doi.org/10.1177/1046496405277134

Salas, Eduardo, Grossman, R., Hughes, A. M., & Coultas, C. W. (2015). Measuring Team Cohesion. *Human Factors*, *57*(3), 365–374.

https://doi.org/10.1177/0018720815578267

Sancho, P., Torrente, J., Marchiori, E. J., & Fernández-manjón, B. (2011). Enhancing Moodle to Support Problem Based Learning The Nucleo experience. *Proc. - IEEE Global Engineering Education Conference, EDUCON*, 1177–1182. https://doi.org/10.1109/EDUCON.2011.5773296

Sangwan, R., Bass, M., Mullick, N., Paulish, D. J., & Kazmeier, J. (2007). *Global software development handbook*. Publisher Auerbach publications.

Saniie, J., Oruklu, E., Hanley, R., Anand, V., & Anjali, T. (2015). Transforming computer engineering laboratory courses for distance learning and collaboration. *International Journal of Engineering Education*, *31*(1), 106–120.

Sarker, S., Sarker, S., & Schneider, C. (2009). Seeing remote team members as leaders: A study of US-Scandinavian teams. *IEEE Transactions on Professional Communication*, *52*(1), 75–94. https://doi.org/10.1109/TPC.2008.2007871

Sawyer, S., & Guinan, P. J. (1998). Software development: processes and performance. *IBM Systems Journal*, *37*(4).

Sawyer, Steve. (2004). Software development teams. *Commun. ACM*, *47*(12), 95–99. https://doi.org/10.1145/1035134.1035140

Scannell, M. (2010). *The big book of conflict resolution games*. McGraw Hill.

SchäFer, A. I., & Richards, B. S. (2007). From concept to commercialisation: student learning in a sustainable engineering innovation project. *European Journal of Engineering Education*, *32*(2), 143–165. https://doi.org/10.1080/03043790601118689

Schaffer, S., Lei, K., Reyes, L., Oakes, W., & Zoltowski, C. (2007). Assessing activity systems of design teams in a collaborative service learning environment. *ASEE Annual Conference and Exposition, Conference Proceedings*.

Schwaber, K., & Sutherland, J. (2020). *The Scrum Guide*. scrumguides.org

Seashore, S. E. (1954). *Group cohesiveness in the industrial work group*.

Sedano, T., Ralph, P., & Péraire, C. (2016). Practice and perception of team code ownership. *20th International Conference on Evaluation and Assessment in Software Engineering - EASE '16*, 1–6. https://doi.org/10.1145/2915970.2916002

Sedelmaier, Y., & Landes, D. (2015). A multi-perspective framework for evaluating software engineering education by assessing students' competencies: SECAT - A software engineering competency assessment tool. *Proceedings - Frontiers in Education Conference, FIE*, *2015-Febru*(February). https://doi.org/10.1109/FIE.2014.7044331

Seger, T., Hazzan, O., & Bar-Nahor, R. (2007). How Does Readiness for Agile Development Relate to Team Climate and Individual Personality Attributes? In *Agile Processes in Software Engineering and Extreme Programming* (pp. 257–260). https://doi.org/10.1007/978-3-540-73101-6_49

Seidel, R., Haemmerle, L., & Chambers, C. (2007). A multidisciplinary design education approach for supporting engineering product innovation. *ASEE Annual Conference and Exposition, Conference Proceedings*.

Senior, B., & Swailes, S. (2019). The dimensions of management team performance: a repertory grid study. *International Journal of Productivity and Performance Management*, *53*(4), 317–333. https://doi.org/10.1108/17410400410533908

Seppälä, O., Auvinen, T., Karavirta, V., Vihavainen, A., & Ihantola, P. (2016). What communication tools students use in software projects and how do different tools suit different parts of project work? *38th International Conference on Software Engineering Companion*, 432–435. https://doi.org/10.1145/2889160.2889196

Sessa, V. I., London, M., Pingor, C., Gullu, B., & Patel, J. (2011). Adaptive,. *Team Performance Management*, *17*(3/4), 146–167.

Sfetsos, P., & Stamelos, I. (2011). Improving Quality by Exploiting Human Dynamics in Agile Methods. In *Agile Software Development Quality Assurance* (pp. 154–170). Idea

Group. https://doi.org/10.4018/978-1-59904-216-9.ch008

Shahzad, S., & Slany, W. (2009). Knowledge Management Issues in Teaching Extreme Programming. *Proc. - I-KNOW '09 and I-SEMANTICS '09*.

Shaikh, M. K., Raza, A., & Ahsan, K. (2016). Software project management as team building intervention. *Journal of Basic and Applied Sciences*. https://doi.org/10.6000/1927-5129.2016.12.56

Shankarmani, R., Mantha, S. S., & Babu, V. (2011). Performance assessment of ASD team using FPL football rules as reference. *Proceedings - 2011 Annual IEEE India Conference: Engineering Sustainable Solutions, INDICON-2011*, 1–4. https://doi.org/10.1109/INDCON.2011.6139390

Shaw, M. E. (1981). *Group dynamics: The psychology of small group behavior*. NY: McGraw-Hill.

Shrivastava, S. V, & Rathod, U. (2015). Categorization of risk factors for distributed agile projects. *Information and Software Technology*, *58*, 373–387. https://doi.org/10.1016/j.infsof.2014.07.007

Shull, F. (2013). Getting an intuition for big data. *IEEE Software*, *30*(4), 3–6. https://doi.org/10.1109/MS.2013.76

Shuto, M., Washizaki, H., Kakehi, K., Fukazawa, Y., Yamato, S., & Okubo, M. (2016). Learning effectiveness of team discussions in various software engineering education courses. *2016 IEEE 29th Conference on Software Engineering Education and Training, CSEEandT 2016*, 227–231. https://doi.org/10.1109/CSEET.2016.31

Silva, F. Q. B., França, A. C. C., Suassuna, M., Mariz, L. M. R. D. S., Rossiley, I., Miranda, R. C. G. De, Gouveia, T. B., Monteiro, C. V. F., Lucena, E., & Cardozo, E. S. F. (2013). Team building criteria in software projects: A mix-method replicated study. *Information and Software Technology*, *55*(7), 1316–1340. https://doi.org/10.1016/j.infsof.2012.11.006

Silvestre, L., Ochoa, S. F., & Marques, M. (2016). Understanding the design of software development teams for academic scenarios. In IEEE Computer Society (Ed.), *34th International Conference of the Chilean Computer Science Society (SCCC)*. https://doi.org/10.1109/SCCC.2015.7416570

Simons, A., Latko, J., Saltos, J., Gutscoven, M., Quinn, R., Duarte, A., Malheiro, B., Ribeiro, C., Ferreira, F., Silva, M., Ferreira, P., & Guedes, P. (2017). Self-oriented solar mirror - An EPS@ISEP 2017 project. *ACM International Conference Proceeding Series*, *Part F1322*. https://doi.org/10.1145/3144826.3145360

Sims-Knight, J. E., Upchurch, R. L., Powers, T. A., Haden, S., & Topciu, R. (2002). Teams in software engineering education. *32nd ASEEE/IEEE Frontiers in Edu Conf*.

Singh, Param Vir; Tan, Yong; Mookerjee, V. (2011). Network effects: the influence of structural capital on open source project success. *MIS Quarterly: Management Information Systems*, *35*(4), 813–829.

Singh, S., Chen, H. C., Hunter, O., Grundy, J., & Hosking, J. (2005). Improving agile software development using eXtreme AOCE and aspect-oriented CVS. *Proceedings - Asia-Pacific Software Engineering Conference, APSEC*, *2005*, 752–759. https://doi.org/10.1109/APSEC.2005.71

Smith, E. K., Bird, C., & Zimmermann, T. (2016). Beliefs, practices, and personalities of software engineers. *Proc. - CHASE'16*, 15–18. https://doi.org/10.1145/2897586.2897596

Smith, K. A. (2012). Five Major Shifts in 100 Years of Engineering Education. *Proc. IEEE*, *100*(Centennial Special Issue). https://doi.org/10.1109/JPROC.2012.2190167

Smith, T., Cooper, K. M. L., & Longstreet, C. S. (2011). Software engineering senior design course. *Proc. 1st Int. Workshop on Games and Soft. Eng.*, 9–12. https://doi.org/10.1145/1984674.1984679

Smith, T., Tull, A., Cooper, K., & Longstreet, S. (2011). Using simulation training games to create more active and student centered learning environments for software and systems engineering education. *Proc. of 1st Int. Conf. on Simulation and Modeling Methodologies, Technologies and Applications*, 386–392. https://doi.org/10.5220/0003621603860392

So, C., & Scholl, W. (2009). Perceptive Agile Measurement: New Instruments for Quantitative Studies in the Pursuit of the Social-Psychological Effect of Agile Practices. *Proc- Int. Conf. on Agile Processes and Extreme Programming in Soft. Eng.*, 83–93.

Sole, D., & Edmondson, A. C. (2002). Situated knowledge and learning in dispersed teams. *British Journal of Management*, *13*(2), 17–34.

Somech, A., Desivilya, H. S., & Lidogoster, H. (2009). Team conflict management and team effectiveness: The effects of task interdependence and team identification. *Journal of Organizational Behavior*, *30*(3), 359–378. https://doi.org/10.1002/job.537

Standish Group. (2015). *"The Standish Group Report Chaos."* https://doi.org/10.1145/1145287.1145301

Stankovic, N. (2009). Single development project. *Journal of Systems and Software*, *82*(4), 576–582. https://doi.org/10.1016/j.jss.2008.12.046

Stavru, S. (2014). A critical examination of recent industrial surveys on agile method usage. *Journal of Systems and Software*, *94*, 87–97. https://doi.org/10.1016/j.jss.2014.03.041

Stawnicza, O. (2015). Distributed team cohesion-not an oxymoron. The impact of information and communications technologies on teamness in globally distributed IT projects. *International Journal of Information Systems and Project Management*, *3*(2), 23–39. https://doi.org/10.12821/ijispm030202

Steghöfer, J., Burden, H., Alahyari, H., & Haneberg, D. (2017). No silver brick : Opportunities and limitations of teaching Scrum with Lego workshops. *The Journal of Systems & Software*, *131*, 230–247. https://doi.org/10.1016/j.jss.2017.06.019

Steghöfer, J., Knauss, E., & Ericsson, M. (2016). Teaching Agile – Addressing the Conflict Between Project Delivery and Application of Agile Methods Categories and Subject Descriptors. *IEEE/ACM 38th IEEE International Conference on Software Engineering Companion*, 303–3012.

Stettina, C. J., Zhao, Z., Back, T., & Katzy, B. (2013). Academic education of software engineering practices: Towards planning and improving capstone courses based upon intensive coaching and team routines. *Software Engineering Education Conference, Proceedings*, 169–178. https://doi.org/10.1109/CSEET.2013.6595248

Stock, T., & Kohl, H. (2018). Perspectives for International Engineering Education:: Sustainable-oriented and Transnational Teaching and Learning. *Procedia Manufacturing*, *21*(July 2019), 10–17. https://doi.org/10.1016/j.promfg.2018.02.089

Stoica, A. J., & Islam, S. (2012). Educational methods for software and systems development. In *Proc. - 2012 15th International Conference on Interactive Collaborative Learning, ICL 2012*. https://doi.org/10.1109/ICL.2012.6402127

Sudol, L. A., & Jaspan, C. (2010). Analyzing the strength of undergraduate misconceptions about software engineering. *Proceedings of the Sixth International Workshop on Computing Education Research - ICER '10*, 31. https://doi.org/10.1145/1839594.1839601

Sullivan, J. F., Knight, D. K., & Carlson, L. E. (2002). Team building in lower division projects courses. *Proc. - Frontiers in Education Conference*, 7–12.

T. DeMarco, T. L. (1999). *Peopleware: Productive Projects and Teams* (second).

Tafliovich, A., Petersen, A., & Campbell, J. (2015). On the Evaluation of Student Team Software Development Projects. *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, 494–499. https://doi.org/10.1145/2676723.2677223

Tai, G. (2005). A communication architecture from rapid prototyping. *ACM SIGSOFT*

Software Engineering Notes, *30*(4), 1. https://doi.org/10.1145/1082983.1083120

Tambo, T., Olsen, M., & Bækgaard, L. (2015). Motives for feral systems in Denmark. In *Web Design and Development: Concepts, Methodologies, Tools, and Applications* (pp. 193–222).

Tang, F. (2015). When communication quality is trustworthy? Transactive memory systems and the mediating role of trust in software development teams. *R&D Management*, *45*, 41–59. https://doi.org/10.1111/radm.12051

Taylor, K. J. (2016). Adopting Agile software development: the project manager experience. *Information Technology and People*, *29*(4), 670–687. https://doi.org/10.1108/ITP-02-2014-0031

Teiniker, E., Paar, S., & Lind, R. (2011). A Practical Software Engineering Course with Distributed Teams. *14th Int. Conf. on Interactive Collaborative Learning*, *September*, 195–201.

Tenhunen, L. J., Niittymaeki, S., & Aarnio, S. (2010). Rapid prototyping service model by the CDIO educational framework. *Annals of DAAAM and Proceedings of the International DAAAM Symposium*, *21*(1), 1571–1572.

Tesluk, P. E., Quigley, N. R., & Tesluk, P. E. (2009). A Longitudinal Study of Team Conflict, Conflict Management, Cohesion, and Team Effectiveness. *Group & Organization Management*, *34*(2), 170–205.

Thomas, J. C., & Baker, S. W. (2008). Establishing an agile portfolio to align IT investments with business needs. *Proceedings - Agile 2008 Conference*, 252–258. https://doi.org/10.1109/Agile.2008.29

Tikanov, V. (2014). *How to deal with conflicting developers in Scrum team*. Project Management Forum. pm.stackexchange.com

Tindale, R. S., & Vollrath, D. A. (1997). The Emerging Conceptualization of Groups as Information Processors. *Psychological Bulletin*, *121*(1), 43–64.

Triviño, A., De La Rubia, E., Moreno, F. A., Lopez-Martinez, F. J., & Sanchez-Martinez, J. J. (2015). Implementing a competitive learning framework in Chemical Engineering degree in Spain and its applicability on an inter-university scenario. *Proceedings - Frontiers in Education Conference, FIE*, *2015-Febru*(February), 0–3. https://doi.org/10.1109/FIE.2014.7044244

Valerdi, R., & Madachy, R. (2007). Impact and contributions of MBASE on software engineering graduate courses. *Journal of Systems and Software*, *80*(8), 1185–1190. https://doi.org/10.1016/j.jss.2006.09.051

Van den Bossche, P., Gijselaers, W. H., Segers, M. S. R., & Kirschner, P. A. (2006). Social and cognitive factors driving teamwork in collaborative learning environments: Team learning beliefs and behaviors. *Small Group Research*, *37*(5), 490–521.

Van Der Duim, L., Andersson, J., & Sinnema, M. (2007). Good practices for Educational Software Engineering Projects. *Proc. - 29th Int. Conf. on Soft. Eng*, 698–707. https://doi.org/10.1109/ICSE.2007.40

van der Vegt, G. S., & Bunderson, J. S. (2005). Learning and performance in multidisciplinary teams: The importance of collective team identification. *Academy of Management Journal*, *48*(3), 532–547.

van der Vegt, Gerben S, Emans, B. E., & Vuert, E. (2001). Patterns of interdependence in work teams: a two-level investigation of the relations with job and team satisfaction. *Personnel Psychology*, *54*, 51–69.

van Kelle, Evelyn; van der Wijst, Per; Plaat, Aske; Visser, J. (2015). An Empirical Study into Social Success Factors for Agile Software Development. *Proc. - IEEE/ACM 37th IEEE International Conference on Software Engineering CHASE 2015*.

Van Woerkom, M., & Croon, M. A. (2009). The relationships between team learning activities and team performance. *Personnel Review*, *38*(5), 560–577.

Vanhanen, J., Lehtinen, T. O. A., & Lassenius, C. (2012). Teaching real-world software engineering through a capstone project course with industrial customers. *St International Workshop on Software Engineering Education Based on Real-World Experiences, EduRex 2012*, 29–32. https://doi.org/10.1109/EduRex.2012.6225702

Vasilescu, B., Posnett, D., Ray, B., van den Brand, M. G. J., Serebrenik, A., Devanbu, P., & Filkov, V. (2015). Gender and Tenure Diversity in GitHub Teams. *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*, 3789–3798. https://doi.org/10.1145/2702123.2702549

Vat, K. H. (2006). Integrating Industrial Practices in Software Development through Scenario-Based Design of PBL Activities: A Pedagogical Re-Organization Perspective The Industrial Practices in Software Development. *Informing Science + IT Educ. Conf. InSITE 2006*.

Vat, K. H. (2017). Integrating Industrial Practices in Software Development through Scenario-Based Design of PBL Activities: A Pedagogical Re-Organization Perspective. *Issues in Informing Science and Information Technology*, *3*, 687–708. https://doi.org/10.28945/924

Venkatesan, V., & Sankar, A. (2014). Investigation of student's personality on pair programming to enhance the learning activity in the academia. *Journal of Computer Science*, *10*(10), 2020–2028. https://doi.org/10.3844/jcssp.2014.2020.2028

VersionOne. (2016). *Agile Methods: 11th annual State of Agile*.

Vilkki, K. (2010). *When agile is not enough*. Nokia Siemens Networks.

Villanueva, I., Manthe, R. L., & Knapstein, K. M. (2013). Development of a design- And project-based framework to include scientific reasoning in an undergraduate, introductory-level bioengineering laboratory course. *ASEE Annual Conference and Exposition, Conference Proceedings*.

Villavicencio, Monica. (2014). *Development of a framework for the education of software measurement in software engineering undergraduate programs. Thesis presented to the degree of Doctor of Philosophy*. École De Technologie Supérieure Université Du Québec.

Villavicencio, Monica, Narvaez, E., Izquierdo, E., & Pincay, J. (2017). Learning scrum by doing real-life projects. *Proc. - EDUCON 2017*, *April*, 1450–1456. https://doi.org/10.1109/EDUCON.2017.7943039

Villavicencio, Mónica, Narváez, E., Izquierdo, E., Pincay, J., Superior, E., Ingeniería, F. De, & Gustavo, C. (2017). *Learning Scrum by doing real-life projects*. *April*, 1450–1456. https://doi.org/10.1109/EDUCON.2017.7943039

Vivian, R., Falkner, K., Falkner, N., & Tarmazdi, H. (2016). A Method to Analyze Computer Science Students' Teamwork in Online Collaborative Learning Environments. *ACM Trans. Comput. Educ.*, *16*(2), 1–28. https://doi.org/10.1145/2793507

Voulgari, I., & Komis, V. (2015). Exploring group cohesion in Massively Multiplayer Online Games. *Proceedings of the European Conference on Games-Based Learning*, *2015-Janua*, 564–570.

Wakefield, R. L., Leidner, D. E., & Garrison, G. (2008). A model of conflict, leadership, and performance in virtual teams. *Information Systems Research*, *19*(4), 434–455. https://doi.org/10.1287/isre.1070.0149

Wang, Z. (2009). Team, Leadership, Ethic, and Profession in Software Engineering Education. *WRI World Congress on Software Engineering*, 81–83. https://doi.org/10.1109/WCSE.2009.360

Watkins, K. Z. (2009). Peer evaluation as a needed web 2.0 activity in project management for teaching practical software engineering. *Proc. - 10th ACM Conference on SIG-Information Technology Education - SIGITE '09*, 173. https://doi.org/10.1145/1631728.1631774

Watkins, K. Z., & Barnes, T. (2010). Competitive and agile software engineering education. *Proc. - IEEE SoutheastCon 2010 (SoutheastCon)*, 111–114.

https://doi.org/10.1109/SECON.2010.5453908

Way, T. P. (2015). A Virtual Laboratory Model for Encouraging Undergraduate Research. *Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education - SIGCSE '06*, *December*. https://doi.org/10.1145/1121341.1121406

Weaver, C. (2015). Human factors engineering framework for applying NUREG. *9th International Topical Meeting on Nuclear Plant Instrumentation, Control, and Human-Machine Interface Technologies, NPIC and HMIT 2015*.

Weimar, E., Nugroho, A., Visser, J., & Plaat, A. (2013). Towards high performance software teamwork. *Proc. - 17th International Conference on Evaluation and Assessment in Software Engineering - EASE'13*, 212–215. https://doi.org/10.1145/2460999.2461030

Wellen, J., & Neale, M. (2006). Deviance self-typicality and group cohesion: The corrosive effects of the bad apples on the barrel. *Small Group Research*, *37*, 165–186.

Wellington, C. A., Briggs, T., & Girard, C. D. (2005a). Comparison of student experiences with plan-driven and agile methodologies. *Proc. - 35th Frontiers in Educ. Conf.*, 18–23. https://doi.org/10.1109/FIE.2005.1611951

Wellington, C. A., Briggs, T., & Girard, C. D. (2005b). Examining team cohesion as an effect of software engineering methodology. *Proc. - Workshop on Human and Social Factors of Software Engineering - HSSE '05*, 1–5. https://doi.org/10.1145/1083106.1083122

Wendorff, P. (2002). Organisational Culture in Agile Software Development. *Proc. - POROFES 2002*, 145–157.

Whitworth, E. (2008). Experience report: The social nature of agile teams. *Proceedings - Agile 2008 Conference*, 429–435. https://doi.org/10.1109/Agile.2008.53

Whitworth, E., & Biddle, R. (2007). Motivation and Cohesion in Agile Teams. *Proc. - Agile Processes in Software Engineering and Extreme Programming*, 62–69. https://doi.org/10.1007/978-3-540-73101-6_9

Wielenga-Meijer, E. G., Taris, T. W., Wigboldus, D. H., & Kompier, M. A. (2012). Don't bother me Learning as a function of task autonomy and cognitive demands. *Human Resource Development International*, *15*(1), 5–23.

Wiese, C. W., & Burke, C. S. (2019). Understanding Team Learning Dynamics Over Time. *Frontiers in Psychology*, *10*(1417). https://doi.org/10.3389/fpsyg.2019.01417

Wilson, J. M., Goodman, P. S., & Cronin, M. A. (2007). Group learning. *Academy of Management Review*, *32*(4), 1041–1059.

Wilson, Jeanne M., O'Leary, M. B., Metiu, A., & Jett, Q. R. (2008). Perceived proximity in virtual work: Explaining the paradox of far-but-close. *Organization Studies*, *29*(7), 979–1002. https://doi.org/10.1177/0170840607083105

Wolfe, J., & Box, T. M. (1987). Team Cohesion Effects on Business Game Performance. *Simulation & Games*, *19*, 82–98. https://doi.org/10.1177/003755008801900105

Wongthongtham, P., & Kasisopha, N. (2010). An ontology-based method for measurement of transferability and complexity of knowledge in multi-site software development environment. *Lecture Notes in Computer Science*, *6746 LNAI*, 238–252. https://doi.org/10.1007/978-3-642-24788-0_22

Wood, C. (1998). Meeting customer needs using participatory techniques. *Proceedings 1998 Australasian Computer Human Interaction Conference. OzCHI'98 (Cat. No.98EX234)*, *10*, 336. https://doi.org/10.1109/OZCHI.1998.732236

Wood, S., Michaelides, G., & Thomson, C. (2013). Successful extreme programming: Fidelity to the methodology or good teamworking. *Information and Software Technology*, *55*(4), 660–672. https://doi.org/10.1016/j.infsof.2012.10.002

Woodley, H. J. R., Mclarnon, M. J. W., & Neill, T. A. O. (2019). The Emergence of Group Potency and Its Implications for Team Effectiveness. *Frontiers in Psychology*, *10*. https://doi.org/10.3389/fpsyg.2019.00992

Wu, W.-H., Chen, W.-F., Wang, T.-L., & Su, C.-H. (2008). Developing and evaluating a game-based software engineering educational system. *International Journal of Engineering Education*, *24*(4), 681.688.

Xavier, S., Murphy, C., & Systems, D. (2016). Work in Progress : A Student Activity Dashboard for Ensuring Project-based Learning Compliance. *Proc. - ASEE Annual Conference and Exposition*.

Xiao, D., & Miller, R. C. (2014). A Multiplayer Online Game for Teaching Software Engineering Practices. In *Poster - L@S'14*. https://doi.org/10.1145/2556325.2567858

Xiaohua, W., Zhi, W., & Ming, Z. (2008). The relationship between developers and customers in agile methodology. *Proc. - Int. Conf. on Computer Science and Information Technology, ICCSIT 2008*, 566–572. https://doi.org/10.1109/ICCSIT.2008.9

Xiong, Y., & Wang, A. (2010). A new combined method for UCD and software development and case study. *2nd International Conference on Information Science and Engineering, ICISE2010 - Proceedings*, 1–4. https://doi.org/10.1109/ICISE.2010.5690032

Xu, W., & Frezza, S. (2011). A case study: Integrating a game application-driven approach and social collaborations into software engineering education. In *13th Int. Conf. on Enterprise Information Systems: Vol. 4 SAIC* (Issue HCI/-, pp. 23–32). https://doi.org/10.5220/0003445300230032

Xuan, Q., Fang, H., Fu, C., & Filkov, V. (2015). Temporal motifs reveal collaboration patterns in online task-oriented networks. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, *91*(5), 052813. https://doi.org/10.1103/PhysRevE.91.052813

Xudong, L., Tao, X., & Chunxia, Z. (2008). Discussion about an undergraduate course on software architecture. *Proc. - CSSE 2008*, *5*, 616–619. https://doi.org/10.1109/CSSE.2008.436

Xue, Y., Yu, T., & Hock Hai, T. (2015). Fostering Fast-response Spontaneous Virtual Team: Effects of Member Skill Awareness and Shared Governance on Team Cohesion and Outcomes. *Journal of the Association for Information Systems*, *16*(11), 919–946. https://doi.org/10.17705/1jais.00414

Yan, X. (2012). On collective efficacy in the competitive ball games. *Advances in Intelligent and Soft Computing*, *2*, 603–608. https://doi.org/10.1007/978-3-642-25437-6_82

Yang, H. L., & Tang, J. H. (2004). Team structure and team performance in IS development: A social network perspective. In *Information and Management* (Vol. 41, Issue 3, pp. 335–349). https://doi.org/10.1016/S0378-7206(03)00078-8

Yang, X., Tong, Y., & Teo, H. H. (2015). Effects of Member Skill Awareness and Shared Governance on Team Cohesion and Outcomes Fostering Fast-Response Spontaneous Virtual Team : Effects of Member Skill. *Journal of the Association for Information Systems*, *16*(11), 919–946.

Yasin, A., Gaber, S., Omar, M., Mohd, H., & Baharom, F. (2009). *Designing story card in extreme programming using machine learning technique*. 1–5.

Ye, E., Liu, C., & Polack-Wahl, J. A. (2007). Enhancing software engineering education using teaching aids in 3-D online virtual worlds. *Frontiers in Education Conference, FIE*, 8–13. https://doi.org/10.1109/FIE.2007.4417884

Yilmaz, L., & Phillips, J. (2006). Organization-Theoretic Perspective for Simulation Modeling of Agile Software Processes. In Q. W. D. Pfahl & D. M. R. P. Wernick (Eds.), *Lecture Notes in Computer Science. Software Process Change*. https://doi.org/10.1007/3-540-68339-9_34

Yilmaz, M., O'Connor, R. V., Colomo-Palacios, R., & Clarke, P. (2017). An examination of personality traits and how they impact on software development teams. *Information and Software Technology*, *86*, 101–122. https://doi.org/10.1016/j.infsof.2017.01.005

Yoshii, A., & Higa, K. (2011). Analysis of the peculiarity of the japanese software development style in offshore software development. *IEEJ Transactions on Electrical and Electronic Engineering*, *6*(1), 46–50. https://doi.org/10.1002/tee.20605

Young, S. M., Edwards, H. M., McDonald, S., & Thompson, J. B. (2005). Personality Characteristics in an XP Team: A Repertory Grid Study. *Proc. - 2005 Workshop on Human and Social Factors of Software Engineering*. https://doi.org/10.1145/1083106.1083123

Youssef, A., & Capiluppi, A. (2015). The impact of developer team sizes on the structural attributes of software. *Proceedings of the 14th International Workshop on Principles of Software Evolution - IWPSE 2015*, 38–45. https://doi.org/10.1145/2804360.2804365

Yu, L., Surma, D. R., & Hakimzadeh, H. (2014). *Incorporating Free_Open-Source Data and Tools in Software Engineering Education_ Computer Science & IT Book Chapter _ IGI Global*.

Zaccaro, S. J., & Lowe, C. A. (1988). Cohesiveness and performance on an additive task: evidence for multidimensionality. *The Journal of Social Psychology*, *128*(4), 546–558. https://doi.org/10.1080/00224545.1988.9713774

Zarraga-Rodriguez, M., Jaca, C., & Viles, E. (2015). Enablers of team effectiveness in higher education: Lecturers' and students' perceptions at an engineering school. *Team Performance Management*, *21*(5/6), 274–292.

Zeid, A. (2012). Integrating international students' contests with computer sciecnce capstone: Lessons learned and best practices. *Proceedings - Frontiers in Education Conference, FIE*. https://doi.org/10.1109/FIE.2012.6462385

Zeid, A. (2015). Using Simulation Games to Teach Global Software Engineering Courses. *Proceedings - Frontiers in Education Conference, FIE (2015)*.

Zeid, A., & El-Bahey, R. (2015). Extended abstract: Comparing cultural models in the context of teaching global software engineering. *IEEE International Professional Communication Conference*, 6–7. https://doi.org/10.1109/IPCC.2014.7020397

Zemke, S. (2007). Design Team Skills Curriculum For Intermediate Level Project Class. *ASEE Annual Conference and Exposition, Conference Proceedings*.

Zhang, F., Khomh, F., Zou, Y., & Hassan, A. E. (2014). An empirical study of the effect of file editing patterns on software quality. *Journal of Software: Evolution and Process*, *26*(11), 996–1029. https://doi.org/10.1002/smr.1659

Zhang, P., White, J., & Schmidt, D. C. (2016). HoliCoW: Automatically Breaking Team-based Software Projects to Motivate Student Testing. *Proc. - IEEE/ACM 38th Int. Conf. on Soft. Eng. Companion (ICSE-C)*, 436–439. https://doi.org/10.1145/2889160.2889197

Zhang, Y., & Liu, Y. (2012). Management enhanced double PBL based reform in advanced programming design course. In *Proceedings of the 14th IEEE International Conference on High Performance Computing and Communications, HPCC-2012 - 9th IEEE International Conference on Embedded Software and Systems, ICESS-2012* (pp. 1658–1663). https://doi.org/10.1109/HPCC.2012.244

Zimmermann, O. (2016). Designed and Delivered Today, Eroded Tomorrow? Towards an Open and Lean Architecting Framework Balancing Agility and Sustainability. *Proc. - 10th European Conference on Software Architecture Workshops (ECSA-W)*, *33*(6), 2016. https://doi.org/10.1007/s00607-016-0520-y

Appendixes

Appendix 1. Overview of teaching-learning frameworks in engineering education

1.1 Approaches combined with team-based learning in engineering education

Number of Studies	Approaches	Field
1	challenge-based-learning	Mechanical engineering; Sustainable engineering;
17	problem/ project-based learning	Sustainable engineering; Engineering design; Innovation, Bioinformatics; Civil engineering; Mechanical engineering; Mechatronics; Aerospace engineering; Information and computer technology
4	Kolb's learning styles and cycle of experiential learning	Sustainable Development and Innovations; Environmental engineering; Management Information Systems
1	diamond model	Sustainable Development and Innovations
6	ICT enhanced education	Programming; Engineering design, build, and test (DBT); Information Sciences, Computer engineering. Management Information Systems, Software engineering
1	simulation games	Industrial engineering
6	real-world projects/ collaboration with industry	Software engineering, Financial computing, Civil engineering, Innovation; Engineering technician education
1	situated learning	Computer engineering
1	collaborative project-based learning	Computer engineering
1	participatory design based	Computer engineering
1	learning by doing	Manufacturing
2	case-based analysis	Environmental engineering, Engineering education
2	competition	Electronics and electrical engineering, Mechanical engineering
5	multidisciplinary teams	Engineering education, Environmental engineering
1	design-based learning	Bioengineering
2	active learning	Aircraft design; Engineering technician education
1	reverse engineering	Aircraft design
1	metacognitive development	Engineering technician education

1.2 Scope of the studies related to teaching-learning frameworks

Reference	Discipline/Field	Description
(Membrillo-Hernández et al., 2019)	Mechanical engineering and Sustainable engineering development	Challenge-based-learning (CBL) to expose students to real experiences with high levels of uncertainty to achieve specific learning objectives
(Malheiro et al., 2019)	European Project Semester/ Smart Sustainable Cities	Project-based learning and capstone projects to the development of key engineering skills, including multidisciplinary teamwork
(Moyne et al., 2018)	Engineering design	A web-based system that collects and reports data to support teaching, learning, and research in teams. This tool has the aim of collecting data for the development of an educational framework in project-based engineering design education
(Stock & Kohl, 2018)	Sustainable Development and	The framework includes problem solving procedure based on Kolb's learning styles and a

	Innovations	Diamond-Model for providing a structure of the start-up development. The students performed in teams. A survey showed the teamwork skills improved with the application of this framework
(Cabrera et al., 2017)	Programming	The approach uses design patterns from online web communities and Team-Based Learning (TBL) to improve student grades
(Simons et al., 2017)	European Project Semester (EPS)/ self-oriented solar mirror (SOSM)	A project-based educational framework that contributes to teamwork in an international and multidisciplinary engineering environment
(Salado et al., 2017)	Industrial engineering education	A simulation game including activities designed to offer industrial engineering seniors experience in solving realistic decision-making problems. The students have to work in teams playing different roles as different types of companies in a global smartphone market
(Harris et al., 2017)	Bioinformatics	Problem-solving and team-learning approach to support the combination of students of computer science and biology to Bioinformatics separately at first then joined together at a later point in the semester
(Malheiro et al., 2019)	European Project Semester (EPS)/ self-oriented	Project-based learning framework to foster engineering skills like multidisciplinary teamwork
(Brady et al., 2016)	Engineering design	Project-based and team-based learning are combined
(A. Hernandez et al., 2016)	Senior computer engineering course	Situated learning, Collaborative Project-based Learning (CPBL), and participatory design-based approaches are combined
(Gao et al., 2016)	Social manufacturing	Learning by doing and project-based learning are combined in integrating 3D printing techniques for additive manufacturing and e-commerce for marketing
(Beever & Hess, 2016)	Environmental engineering	An engineering ethics case study within pedagogical support is used to make students reflect in teams to make decisions developing ethical reasoning
(X. S. D. Henry et al., 2016)	Monitoring water quality	Undergraduate engineering and computer science students set out to design and fabricate a water monitoring and data acquisition system. The students are exposed to a multi-disciplinary team of researchers and faculty members. Kolb's cycle of experiential learning guides the students' activities
(Patil et al., 2016)	Information sciences	This approach combines two courses on web technologies and databases in an integrated framework focused on ABET outcomes
(-, 2015)	Financial computer science	A team-based educational framework where students learn through real problem solving
(Triviño et al., 2015)	Electronics and Electrical Engineering	Team-based competitive framework. Students are gathered into teams and compete along with several tasks
(Dunai et al., 2015)	Electric and electronic engineering	Team-based framework aiming to solve innovative projects
Wadhwa, S.,	Engineering education	Pedagogical framework focusing on the

Barlow, A., Jadeja, S.		development of affective domain among the first-year engineering students
(López-Querol et al., 2015)	Civil engineering	Project-based learning framework where students solve real projects working in small teams
(Saniie et al., 2015)	Computer engineering	Real-time and team-based framework for design laboratories in distance education
(Bourn & Baxter, 2014)	Mathematics Education	A pedagogical framework to foster critical thinking in mathematics education
(Baldissera & Delprete, 2014)	Mechanical engineering	A proposal for involving students in vehicle design competitions events while they work in teams
(Kisselburgh et al., 2013)	Engineering education	Pedagogical framework Scaffolded, Integrated, and Reflexive Analysis (SIRA) of ethics cases to enhance the development of moral reasoning that extends beyond case-based analyses
(Villanueva et al., 2013)	Bioengineering	An approach that includes project-based and design-based learning to introduce to bioengineering students with scientific strategies. They are exposed to three competency domains: cognitive, intrapersonal, interpersonal
(Chilton, 2012)	Management of Information Systems	Combines the use of videos and experiential knowledge approach in a virtual classroom while students perform in teams
(Liebenberg & Mathews, 2012)	Design, build and innovation in engineering	An approach that uses problem-solving strategy in designing and building solutions to set technological problems working in teams.
(Liang, 2012)	Troubleshooting in automotive braking system (TiABS)	Web-based learning framework that supports a teamwork-based project design and implementation
(Chin & Yue, 2011)	Mechatronics	A vertical curricular model based on the PBL approach where the students have to perform in teams
(Tenhunen et al., 2010)	Rapid Prototyping Service RPS,	A model for rapid prototyping teams for undergraduate students
Curran, R., Van Tooren, M., Van Dijk, L. Systems (2009)	Aerospace design	A team-based approach is used for the development of projects in systems engineering for aerospace design
(Elizalde et al., 2008)	Craft design	Reverse Engineering and Active Learning concepts are combined for generating new knowledge in a collaborative way
(SchäFer & Richards, 2007)	Environmental engineering	Volunteer students engage in multidisciplinary teams to develop a project aiming to provide water for remote communities and developing countries
(Schaffer et al., 2007)	Engineering education	Leaning patterns to contribute to the design of collaborative environments for project teams
(Seidel et al., 2007)	Engineering product innovation	A training program that includes teamwork and project-based design courses in collaboration with industry
(Massa et al., 2005)	Engineering technician education	Active learning, real-world problem solving, and metacognitive development are combined to develop learner proficiency. It includes interdisciplinary teamwork in group reflection activities
(Ramakrishnan & Cambrell, 2004)	Software engineering	Online learning community to facilitate the interactions of the student in their group projects
(Pollard, 2003)	Information and	An approach to learning how to research by using

	computer technology	templates and problem-solving working in teams
(Nagchaudhuri, 2001)	Mechatronics	It focuses on pre-colleges programs. The proposal includes teamwork to develop projects.

Appendix 2: Excluded papers from literature review 1

1.1 Excluded papers not available

Authors	Title
Dykman, Nathan; Ragaisis, Saulius	Teaching HCI in SE curriculum
Ramakrishnan, S.	A service-oriented portal for software engineering education
Cooper, K.; Simmons, D.; Wong, W.E.	Revitalizing software engineering education in the 21 st Century

1.2 Excluded full books of proceedings

Conference	Year
2nd International Workshop on Software Engineering Course Projects	2006
19th International Conference on Software Engineering Education and Training	2006
26th International Conference on Software Engineering Education and Training	2013
3rd International Workshop on Collaborative Teaching of Globally Distributed Software Development	2013

1.3 Excluded papers not relevant for education

Reference	Scope of the study
(Lopez-Martin et al., 2008)	This paper aimed to compare personal Fuzzy Logic Models (FLM) with a Linear Regression Model
(Germain & Robillard, 2008)	The purpose of this study was to quantify activity patterns on three empirical axes (engineering, coding and V&V)
(Beranoagirre, 2011)	This paper argues on the utilization of software engineering knowledge to educate mechanical engineers
(Lopez-Martin, 2008)	The study compares the number of defects when design and code reviews are introduced in the individual development process
(Hohpe et al., 2016)	The paper argues on the software architect role and how teams approach architectural decision-making
(Poženel & Mahnič, 2016)	The study compares two software effort estimation techniques (planning poker and the team estimation game)

Appendix 3: Keyword analysis from papers of literature review 1

1.1 Papers without keywords

Reference	Title
(Valerdi & Madachy, 2007)	Impact and Contributions of MBASE on Software Engineering Graduate Courses
(Jaakkola et al., 2006)	IT Curriculum as a Complex Emerging Process
(Yu et al., 2014)	Incorporating Free/Open-Source Data and Tools in Software Engineering Education
(Rico & Sayani, 2009)	Use of agile methods in software engineering education
(Kantipudi et al., 2012)	Software Engineering Course Projects: Failures and Recommendations
(Frailey, 2006)	Bringing Industrial Methods to the Classroom
(Frankl et al., 2014)	Win-for-All in Software Engineering Education: Balancing Social Dilemmas to Foster Collaboration

(Krutz et al., 2015)	Enhancing the Educational Experience for Deaf and Hard of Hearing Students in Software Engineering
(Seppälä et al., 2016)	What communication tools students use in software projects and how do different tools suit different parts of project work?
(Jordan et al., 2017)	Hall of Fame Nomination: Studio-Based Master of Software Engineering Program at Carnegie Mellon University
(H. J. C. Ellis & Hislop, 2013)	Project Selection for Student Involvement in Humanitarian FOSS
(Boehm & Koolmanojwong, 2014)	Combining Software Engineering Education and Empirical Research via Instrumented Real-Client Team Project Courses
(Bareiss & Mercier, 2010)	A Graduate Education in Software Management and the Software Business for Mid-Career Professionals
(Bosnic et al., 2013)	Picking the Right Project: Assigning Student Teams in a GSD Course
(Julian M. Bass et al., 2015)	Virtual Teams and Employability in Global Software Engineering Education
(Stettina et al., 2013)	Academic Education of Software Engineering Practices: Towards Planning and Improving Capstone Courses Based upon Intensive Coaching and Team Routines
(Gotel, Kulkarni, et al., 2009)	A Global and Competition-based Model for Fostering Technical and Soft Skills in Software Engineering Education

1.2 Keywords clusters[4]

Cluster	Keywords	
1. Learning-teaching process and curriculum	1. self-directed learning	54. open learning
	2. meetings-flow	55. informal learning
	3. learning strategies	56. formal learning
	4. programming languages teaching	57. participatory learning
	5. competition-based model	58. learning
	6. meetings-flow	59. students' perspective
	7. experimental learning	60. motivating students
	8. integrated active learning	61. critical thinking
	9. simulation	62. practicum
	10. learning by osmosis	63. MOOCs
	11. learning by doing	64. blended learning
	12. e-learning	65. pedagogical strategies
	13. education strategies **(also included in C9, suggested by supervisor)**	66. educational environment
	14. learning analytic **(also included in C9, suggested by supervisor)**	67. teaching model
	15. active learning	68. simulators
	16. student centered learning environments	69. cognitive processes
	17. experiential learning	70. learning experience
	18. active learning	71. peer-based assessment
	19. ADDIE Instruction Design Model	72. peer assessment
	20. cognitive apprenticeship	73. self-directed learning
	21. intensive coaching	74. knowledge and skills transfer
	22. educational activities	75. learning process
	23. student activities	76. innovative teaching model
	24. technical soft skills	77. educational experience
	25. learning assessment	78. inverted classroom
	26. communities of practice	79. assessment

[4] The suggestions made by the supervisor of this doctoral research are highlighted in bold. Those made by the experts appear in italic, between brackets.

	27. curriculum 28. learning outcomes 29. professional software engineering skills 30. course practice 31. creative thinking 32. groupthink 33. competitive 34. curriculum design 35. story-centered curricula 36. classroom 37. evaluation 38. Bloom learning objectives 39. design scenarios 40. student perspective 41. ambidextrous 42. workshops 43. student perspective 44. project selection 45. education 46. student motivation 47. student activity dashboard 48. learning outcomes 49. individual performance **50.** engineering pedagogy **(also included in C9, suggested by supervisor)** **51.** quality of teaching and learning **(also included in C9, suggested by supervisor)** 52. IT Curriculum **(also included in C9, suggested by supervisor)** 53. diversity *(moved from unclassified, suggested by expert 1 and 2)*	80. teaching resources 81. teaching and learning strategies 82. assessment 83. blended e-Learning 84. programming ability 85. soft skills 86. theory-in-use 87. espoused theory 88. reflective practitioner 89. laboratory 90. teaching model 91. student-centered learning 92. situated cognition theory 93. practice training 94. engineering pedagogy **95.** quality of teaching and learning **(also included in C9, suggested by supervisor)** 96. didactical approaches **(also included in C9, suggested by supervisor)** **97.** curriculum design and implementation **(also included in C9, suggested by supervisor)** 98. education data mining 99. social dilemmas *(moved from unclassified, suggested by experts 1 and 2)*
2. Teamwork management	100. coordination 101. communication 102. teamwork quality 103. teamwork 104. assigning student teams 105. visual studio team system 106. teamwork skills 107. teams 108. group formation 109. communication tools 110. team coordination 111. team design 112. teamwork quality 113. virtual teams 114. remote teams	115. team estimation 116. student teamwork 117. team programming **118.** software engineering teamwork **(also included in C4, suggested by supervisor)** 119. team development 120. local teams 121. student project team 122. student teams

3. Project-problem based learning	123. project course 124. senior project 125. capstone course 126. capstone project 127. educational software engineering projects 128. project-based learning 129. software engineering capstone 130. course projects 131. team project courses 132. double PBL 133. projects 134. team projects 135. capstone design 136. project-based 137. student projects 138. studio project	139. project-based courses 140. capstone software project 141. project-based immersion system 142. based studio projects 143. course projects 144. capstone practicum project 145. student software project 146. capstone program 147. service-learning projects
4. Collaborative learning	148. collaborative development 149. collaborative learning 150. cooperative 151. wiki 152. collaboration skills 153. multiple teams 154. computer-supported collaborative learning activity 155. HBDI *(moved from unclassified, suggested by experts 1 and 3)* 156. cultural dimensions *(moved from unclassified, suggested by experts 1 and 2)*	157. collaborative software development 158. collaboration 159. social collaboration 160. distributed collaboration 161. online tool 162. virtual worlds 163. social skills 164. software engineering teamwork **(also included in C2, suggested by supervisor)**
5. Real projects resolution and industry links	165. real-client 166. industry collaboration 167. real-world learning 168. industry-academia partnership 169. software enterprise 170. IT labor force 171. employability 172. industrial methods 173. student involvement in humanitarian FOSS *(moved from unclassified, suggested by experts 1 and 2)* 174. professional responsibilities *(moved from unclassified, suggested by expert 1, 2 and 3)*	175. real-world problems 176. industrial collaboration 177. academic-industry partnerships 178. real-world team projects 179. "Real-World" software engineering 180. industry partnerships 181. multi-university collaboration
6. Virtual and global teams	182. global software development 183. distributed and global software development 184. internationalization 185. global software engineering 186. distributed software development 187. distributed development 188. multi- university collaboration **(moved from C5, suggested**	190. virtual world 191. global software engineering skills 192. global requirements elicitation 193. distributed software engineering 194. global studio project

	supervisor) 189. offshore development **(moved from C10, suggested supervisor)**	
7. Gamification	195. digital game-based learning 196. improvisational theater 197. role-playing 198. games for learning 199. game development 200. game level design 201. games 202. game-based learning 203. simulation training games 204. second life	205. game-based learning 206. gamification 207. game based education 208. game programming 209. educational games 210. simulation game
8. Agile methods	211. agile methods 212. Scrum 213. agile methodologies 214. Kanban 215. Scrumban 216. agile 217. short iteration duration and small teams 218. agile software development 219. agile development 220. agile learning environment	221. agile organizing framework 222. agile concepts 223. agile software development 224. planning poker 225. extreme programming 226. distributed scrum
9. Engineering education research field	227. software engineering education 228. computer engineering education 229. computer science education 230. engineering education 231. software engineering training and education **232.** academic education **(also included in C12, suggested by supervisor)** **233.** education strategies **(also included in C1, suggested by supervisor)** **234.** didactical approaches **(also included in C1, suggested by supervisor)** **235.** IT Curriculum **(also included in C1, suggested by supervisor)** 236. learning analytic **(also included in C9, suggested by supervisor)**	237. software and systems engineering ed ucation 238. requirement engineering education 239. global software engineering education **240.** engineering pedagogy **(also included in C1, suggested by supervisor)** **241.** curriculum design and implementation **(also included in C1, suggested by supervisor)**
10. Software engineering core discipline and related contents	242. project management 243. effort estimation 244. personal software process 245. fuzzy logic 246. software effort estimation 247. team software process 248. software development 249. size estimation 250. productivity 251. defect density 252. supply chain 253. requirements engineering 254. system development and evaluation	300. traceability 301. development tools 302. offshore software development 303. requirements specification 304. requirements reuse 305. humanitarian free and open source software (HFOSS) 306. free and open source software

	255. software patterns	307. software business
	256. ontologies	308. Win-for-All
	257. semantic web	309. software development methods
	258. engineering technology	310. process automation
	259. software process	311. software development process management
	260. design-implementation	312. design process
	261. team routines	313. CMMI
	262. software engineering practices	314. exploitation
	263. process measurement	315. software development tooling
	264. iterative and incremental development	316. infrastructure
	265. software engineering	317. UML
	266. open source	318. security
	267. FLOSS	319. outsourcing software development
	268. software projects	320. mathematics (STEM) program
	269. software testing	321. engineering ideas
	270. software error injection	322. software project
	271. software development	323. personal software process
	272. community driven development	324. team software process
	273. free/libre open source software	325. experimental software
	274. Free/Open-Source Data and Tools	326. software business
	275. operations support systems (OSS)	327. software engineering process
	276. project monitoring and steering	328. process patterns
	277. software process improvement	329. process activities
	278. software development methods	330. process monitoring
	279. mining software repositories	331. effort
	280. process mining	332. project control and modeling
	281. outsourcing software development	333. software engineering
	282. virtual agents	334. machine learning
	283. human aspects	335. software quality
	284. knowledge management	336. quality model
	285. process and quality	337. project and implementation
	286. security	
	287. programming languages	
	288. software effort estimation	
	289. programming	
	290. multiple software products	
	291. multiple customers	
	292. defect density	
	293. size estimation	
	294. effort estimation	
	295. good practices	
	296. software development	
	297. software management	
	298. personal software process	
	299. fuzzy logic	
11. Research and experimentation	338. experimental software engineering	347. action research
	339. experiment	348. empirical studies
	340. case study	349. field study
	341. empirical research	350. assessment and evaluation approaches
	342. experimental study	351. measurement of training effect
	343. quantitative evaluation	352. software case
	344. experimental study	

	345. quantitative evaluation 346. awareness *(moved from unclassified, suggested by experts 1, 2 and 3)*	studies
12. Levels of education	353. post-secondary education 354. undergraduate science 355. mid-career professionals 356. undergraduate students 357. undergraduate science 358. undergraduate students 359. higher education 360. undergraduate education 361. undergraduate and graduate 362. undergraduate software development project 363. graduate education	364. first year 365. undergraduate students 366. software engineering graduate courses **367.** academic education **(also included in C9, suggested by supervisor)** 368. undergraduate research projects
Unclassified	369. ROBOCODE 370. misconceptions 371. social processes 372. reflection 373. retrospective 374. XNA 375. Net Generation 376. hearing-impaired students 377. MBASE 378. deaf and hard of hearing students 379. independent college 380. course-of-action	381. model trains 382. fair division 383. globalization 384. complex emerging process 385. SimSE 386. Studio Project 387. gender influence 388. Electronic Learning Industrial Environment (eLIN) System 389. mechanical engineering 390. workshops

Appendix 4. Overview of studies included in the network of trends

Reference	Reason for its inclusion in the correspondent clusters	Cluster
(Viljan Mahnic, 2012)	The paper presents an undergraduate capstone course where students work in Scrum teams	A
(C. Y. Chen & Chong, 2011)	The paper reports on a case study that introduces the meetings-flow (MF) approach, a project-based educational collaboration environment, and showed its positive influence on software quality and progress	C
(Viljan Mahnic, 2010)	The paper presents the design of a course to teach Scrum through capstone projects	A
(Van Der Duim et al., 2007)	The paper describes experiences from two universities running an international course where student teams perform an industrial software project. A set of good practices for project-based software engineering education are given	V, R
(Chung-yang Chen & Teng, 2011)	The paper presents a computerized environment that supports the Meetings-Flow approach	C
(Minocha et al., 2008)	The paper reports on the effectiveness of using wikis for distributed requirement engineering in a software engineering course	C, V
(Wu et al., 2008)	The paper presents a game-based learning system to support the teaching of software development processes in a team-based environment	C, G

(Rico & Sayani, 2009)	The paper reports experiences about the introduction of agile methods in a capstone course for a master degree. A set of lessons learned are provided	A
(Garcia & Pacheco, 2014)	The paper presents an approach that integrates TSPi methodology and PBL. A course that uses the approach with local software industry collaboration is discussed	R
(Jeremic et al., 2009)	The paper presents a learning environment named DEPTHS that serves as a common ontological foundation for integrating different learning tools and systems. The environment aims to facilitate active examination of learning resources and work on real-world projects in small teams	R
(Rodriguez et al., 2015)	The paper presents an educational virtual world that simulates a Scrum-based team room for developing capstone projects in undergraduate courses	A
(Gotel, Kulkarni, et al., 2009)	The paper presents a model that brings undergraduate, graduate and industry students together in global software development projects	V, R
(Chung-yang Chen et al., 2014)	The paper reports on the Meetings-Flow approach in regard to quality teamwork in software capstone projects	C
(Jeremić et al., 2011)	The project-based collaborative learning environment named DEPTHS is further described and investigated in the context of software design pattern education	C, R
(Gary, 2008)	The paper presents a pedagogical model that emphasizes teaching software engineering through real projects in collaboration with industry	R
(Bareiss & Katz, 2011)	The paper reports on the results of a student survey regarding the capstone project they developed in a master program at Carnegie Mellon Silicon Valley. These team-based projects had industrial sponsors	R
(Valerdi & Madachy, 2007)	The paper presents the MBASE framework, an approach to the development of software systems that integrates several models to develop real products in the SE courses	R
(Viljan Mahnic, 2015a)	The paper presents a capstone course design for teaching principles of Kanban and Scrum while developing real projects	A, R
(Nguyen et al., 2013)	The paper reports on a study where students develop real-world projects in small teams. The authors discuss gaps in the IT labor market this study showed	R
(Stettina et al., 2013)	The paper reports on a quasi-experiment where the authors explore the effectiveness of coaching and team routines on a SE course	A
(Bareiss & Mercier, 2010)	The paper presents the curricula for software management and business undergraduate program at Carnegie Mellon's Silicon Valley. It emphasizes team-based and project-based pedagogy	C, R
(Collazos et al., 2010)	The paper presents the computer-supported collaborative learning environment CODILA for undergraduate SE students that foster collaborative skills in distributed development teams	C, V
(Casallas & Lopez, 2008)	The paper describes a strategy that targets real-project development using active teaching-learning methodologies to create scenarios with regular self-assessment for teams of undergraduate students	R
(Berkling et al., 2007)	The paper reports on a case study on the education of master students for offshore software development projects. Their	C, V

	approach includes issues on communication, knowledge management, and project and process management needed for remote collaboration platforms	
(Carrillo De Gea et al., 2016)	Two requirements specification techniques (traditional and reuse-based) were compared regarding their effect on a set of performance-based and perception-based variables in collocated and distributed settings for student teams	V
(Ferdiana, 2016)	The paper presents a teaching model for learning in organizations. The model helps teams to identify, learn and validate what is needed for them to successfully develop projects	C
(V. Mahnic & Casar, 2016)	The paper presents a tool that supports a Scrum-based software engineering capstone course	A
(Shuto et al., 2016)	The paper reports on a study that investigates the influence of team discussions on learning effectiveness in two software engineering courses	C
(Julian M. Bass et al., 2015)	The paper presents a model for introducing global software engineering into the computing curriculum	V
(Buffardi, 2015)	The paper reports on the involvement of students in local free and open-source software (LFOSS) organization	R
(J. J. Y. Chen & Wu, 2015)	The paper presents a method that integrates extreme programming (XP) with existing courses	A
(Marques, 2015)	The paper reports on a case study that shows mixed-gender software engineering student teams were more effective and coordinated	C
(Zeid, 2015)	The paper presents a model for distributed global software development simulation games	G, V
(Boehm & Koolmanojwong, 2014)	The paper provides a brief description of opportunities for empirical research in software engineering through team project courses that involve real clients	R
(Frankl et al., 2014)	The authors of this paper argue on the role of the win-win model for successful collaboration in software engineering student teams	C
(Zeid & El-Bahey, 2015)	The paper presents the scope of research that aims to identify contextual factors that influence productivity within globally distributed teams in teaching software engineering	V
(E. Choi, 2013)	The paper reports on a case study that investigates the application of the inverted class model in an introductory course of software engineering where the students had to collaborate in teams to develop projects	C
(Bottcher et al., 2013)	The paper reports experience using improvisational theater techniques to foster team building and creativity in software engineering courses	G
(Bosnic et al., 2013)	The paper discusses assigning students to project teams in a global software development course	V
(H. Ellis et al., 2013)	The paper presents an approach to identify an appropriate project for student involvement in Free Open Source Software that exposes students to real-world development teams	V, R
(Manamendra et al., 2013)	The appropriateness of Scrum for the undergraduate projects is evaluated through a course that was redesigned to introduce this methodology	A
(Lago et al., 2012)	The paper presents a pedagogical model that foster learning of complementary topics by working together with experts in global teams	V
(Ahmad et al., 2012)	The paper presents a pedagogical model based on role-playing for a software engineering master program	G

(Bleicher et al., 2012)	The paper reports on the use of collaborative development platforms of IBM to support team-based activities in software project management at the graduate level	C
(Emam & Mostafa, 2012)	The paper presents a game design approach for a collaborative involvement of software engineering student teams in developing projects that are not part of formal courses	C, G
(Jamaludin et al., 2012)	The paper presents the eLIN system, a collaborative environment for requirement engineering that supports PBL in undergraduate education	C
(Dragutin Petkovic et al., 2012)	The paper presents a method to assess and predict teamwork effectiveness by using machine learning techniques	C
(Plechawska-Wojcik & Borys, 2012)	The paper presents a method that emphasizes the role of real team projects and the improvement of the development process and software quality in SEE	C
(Y. Zhang & Liu, 2012)	The paper presents a course design that combines project-based learning and problem-based learning for programming undergraduate education	C
(Zeid, 2012)	The paper reports experiences on integrating competitions with capstone computer sciences courses to improve performance, quality, and communication of students teams	C
(Chung-yang Chen & Teng, 2011)	The paper presents a computer-supported collaborative learning environment that supports the Meetings-Flow approach	C
(Alkhatib et al., 2011)	The paper presents a framework that integrates creative thinking and the four brain approaches with agile methods and CMMI in SEE	A
(Benton & Radziwill, 2011)	The Agile Organizing Framework (AOF), an approach that includes three organizing principles in agile environments, is adapted to agile educational settings	A
(Bihari et al., 2011)	Description of a computer science and engineering program that emphasizes capstone courses allowing students to learn from real-world problems	R
(W.-F. Chen et al., 2011)	The paper presents an approach that targets game-based learning	G
(J. Chen et al., 2011)	The paper presents a method to assess teamwork performance	C
(J. Chen, 2011)	The paper presents a method to assess students' performance in capstone projects	C
(Lynch et al., 2011b)	The paper presents a Lego-based game approach to teach agile software development concepts	A, G
(Mamei et al., 2011)	The paper presents a tool to assess individual student performance and teamwork analysis in projects	C
(Amalia Rusu & Gowda, 2011)	The paper reports on qualitative research findings of a study on academic partnership for teaching hands-on classes	R
(T. Smith, Tull, et al., 2011)	The paper presents a simulation game approach for SEE	G
(Teiniker et al., 2011)	The paper describes a software engineering course that uses real-world software engineering projects developed by distributed teams. The pedagogical model includes constructivism, experiential and collaborative learning approaches	C, A, V, R
(Xu & Frezza, 2011)	The paper presents an approach that integrates collaborative practices and games	C, G
(Wongthongtham & Kasisopha, 2010)	The paper presents an approach to measure knowledge sharing in distributed teams	V

(W.-F. Chen, 2010)	The paper reports on the evaluation of two methods to team software engineering: a role-playing gaming strategy and a traditional drill-and-practice gaming strategy	G
(D. Petkovic et al., 2010)	The paper presents a tool for soft-skills teamwork assessment	C
(Watkins & Barnes, 2010)	The paper presents a model for a capstone course to solve real-world problems using competition and agile skills	A, R
(Gotel, Phal, et al., 2009)	The authors propose an infrastructure for students working in globally distributed teams	R, V
(Ragan et al., 2009)	The paper presents a method that uses collaborative problem-based learning to develop software projects	C
(Rico & Sayani, 2009)	The paper discusses the introduction of agile methods in a capstone course	A
(Wang, 2009)	Discussion learning on teams, leadership, and other related topics to prepare software engineers	C
(Bareiss & Griss, 2008)	The paper describes a software engineering program of the Carnegie Mellon team and real project–based	C, R
(J. Beck, 2008)	The paper presents a method for fairly assigning portions of team projects to students	C
(Budd & Ellis, 2008)	The paper describes project-based courses and presents a discussion of assistant teachers and instructors on their experiences along with the courses	R
(Hemer, 2008)	The paper presents a peer assessment method to assess individual contributions of team members along with a web tool to support this method	C
(P. H. P. Huang et al., 2008)	The paper presents a teaching model that combines student-centered learning, situated cognition theory, and practice training	C
(Krogstie, 2008)	The paper reports on a case study of a student’s team working in a real environment of an Open Software Development (OSD). The authors discuss how students collaborate as part of this OSD community	C, V, R
(Nandigam et al., 2008)	The paper describes a course proposal that uses Open Source Software to teach software engineering principles while students work in teams	V
(Amalia Rusu & Swenson, 2008)	The paper reports on a case study of a course in which capstone projects were coordinated by one full-time faculty instructor and one full-time industry practitioner	R
(W. Chen et al., 2008)	The paper presents a team game-based learning model to teach software engineering	G
(Ye et al., 2007)	The paper presents a teaching model that uses Second Life to support teamwork in computer sciences courses	G
(Kessler & Dykman, 2007)	The paper describes a course proposal that merges traditional and agile methods to teach software engineering	A
(Huen, 2007)	The paper presents a curriculum proposal that includes software product line architecture and iterative-incremental development to prepare students to overcome problems faced by globally distributed teams	V
(Duim et al., 2007)	The paper presents experiences of an educational project course taught at the University of Groningen and at Växjö University that aims students learn from industry working in teams	R
(Way, 2015)	The paper presents a virtual laboratory model for encouraging collaborative undergraduate research with faculty and other students	C
(Frailey, 2006)	The paper presents a discussion on how the industry can contribute to SEE	R

(Hislop, 2006)	The paper presents a capstone project design that aims at prototyping software products in collaborative teams	C
(S.-T. Huang et al., 2006b)	The paper presents an introductory SE course proposal that uses the ADDIE model and the Cognitive Apprenticeship framework to enhance team-based process-oriented software development project	C
(S.-T. Huang et al., 2006a)	A detailed description of the proposal in (S.-T. Huang et al., 2006b)	C
(Vat, 2006)	The paper presents a pedagogical model that uses PBL to teach industrial practices of software engineering	C, R
(Ludi, 2006)	The paper presents a graduate course in secure software engineering and the effect of the various background of the students had in teamwork, performance, and the project	C
(Dragutin Petkovic et al., 2006)	The paper presents a course on software engineering methods taught by two universities. The students work in distributed teams	V
(Richardson et al., 2006)	The paper presents a model for teaching globally distributed software development	V
(Valerdi & Madachy, 2007)	The paper presents a framework and complementary tools, to teach real-life software development	R
(Johanyák, 2016)	The paper presents an approach of team-based real-world projects to teach programming	R
(Xavier et al., 2016)	The paper describes a Web-based tool to support project-based learning compliance	C
(Mujkanovic & Bollin, 2016)	The paper presents a group reformation approach that considers the relations between individual characteristics and learning outcomes	C
(Seppälä et al., 2016)	The paper presents a discussion of a set of communication and collaboration tools used in a software engineering course to develop projects in teams	C
(Alsaedi et al., 2016)	The paper presents a team-based game aiming at coordination of tasks project	C, G
(P. Zhang et al., 2016)	The paper presents a method and tool for teaching coding and testing skills through real-world projects	R
(Silvestre et al., 2016)	The paper presents a method to design cohesive teams	C
(Fernandes & Barbosa, 2016)	The paper discusses the participation of student teams in Free/Libre Open Source Software (FLOSS) as an open and informal learning environment	C, V
(Gonzalez & Golf, 2015)	The paper presents a method to reduce attrition rates by bringing student teams from senior project courses to introductory courses to enforce learning and motivate them	C
(Viljan Mahnic, 2015b)	The paper reports a literature review on the use of Scrum in SEE	A
(Chouseinoglou, 2015)	The paper describes a course structure that uses critical thinking while combining lecture with project practicum aiming at studying organizational learning in software development organizations and teams	C
(Xiao & Miller, 2014)	The paper presents a web-based game for teaching programming	G
(Paasivaara et al., 2014)	The paper presents a Lego game for teaching Scrum	G, A
(Holmes et al., 2014)	The paper discusses lessons learned from capstone Open Source Projects for distributed student teams	V
(Basholli et al., 2013)	The paper presents a method for capstone projects students assessment	C
(Péraire & Sedano, 2014)	The paper presents a framework to monitor progress and manage projects	C

(Damian & Borici, 2012)	The paper describes a SE course design where Computer-Supported Cooperative Work (CSCW) is used and students apply agile methods in global teams	C, A, R
(Vanhanen et al., 2012)	The paper presents a capstone project course that targets real-world projects	R
(Damian et al., 2012)	The paper describes a SE course taught by two universities where students are exposed to collaborative tools, use agile methods, and work in globally distributed teams	C, A, V
(Chungyang Chen et al., 2011)	The fundamental constructs of an approach named Meeting-flow are presented. This approach aimed to enhance collaboration in student projects	C
(T. Smith, Cooper, et al., 2011)	The paper presents an agile development method for game projects	G, A
(Paasivaara et al., 2017)	The paper reports on how the performance of student teams using Scrum methodology to develop capstone projects differ	A
(Georgas, 2011)	The paper presents an approach to teach programming skills by developing robots in teams	C
(Jun & Lingli, 2010)	The paper presents a blended learning approach to foster team working	C
(Monasor et al., 2010)	The paper presents a global software development simulator where the students interact in virtual scenarios with agents that play different roles in the project requirements elicitation	V
(Long, 2010)	The paper presents the experiences of a capstone project to prepare students in globally distributed environments. The software project includes outsourcing as a required component	C
(Shahzad & Slany, 2009)	The paper presents experiences in teaching an XP course that emphasizes in knowledge management process while applying agile practices	A
(Adrian Rusu et al., 2009)	The paper reports on a case study where students work in a real-world project and local-remote paired teams	V, R
(Ribaud et al., 2008)	The paper presents an approach that emphasizes in real-world performed by students within a virtual company while tutored by experimented software engineers	V, R
(J. Henry et al., 2008)	The paper presents the results of a project where undergraduate and graduate students work in teams to solve real-world problems	R
(Root et al., 2008)	The paper presents a model where students are provided with templates they have to use to propose solutions to real problems	C, R
(Denninger, 2008)	The paper presents experiences of two courses that focus on game programming	G
(Stankovic, 2009)	The paper describes a project course that emphasizes problem solving and teamwork	C
(H. J. C. Ellis, 2007)	The paper describes an approach to support self-learning. The students independently define and develop projects in teams	C
(Epstein, 2008)	The paper describes a course where students develop projects for pretend companies. The course emphasizes the development of software intending to develop secure software and tackle several issues regarding the human dimension of software engineering	C, R
(Gotel et al., 2007)	The paper presents experiences of running projects where students are involved in globally distributed teams from three institutions	V
(Carrillo De Gea et al.,	The paper reports on a study with co-located and distributed	V

2016)	student teams from two universities to produce requirements documents in both traditional and reuse-based techniques	
(Rajendran et al., 2012)	The paper presents a new component developed for IBM Rational Jazz platform where student teams can manage the risk associated with their capstone projects	C
(Kropp et al., 2016)	The paper describes a curriculum design to teach agile and collaborative practices	C, A
(Kilamo et al., 2014)	The paper reports on a study along a course to observe how collaborative teamwork influences knowledge transfer	C
(Jun & Lingli, 2010)	The paper presents a model of blended learning for learning programming that facilitates collaborative teamwork	C
(Honig, 2008)	The paper reports on the use of TSP methodology in a course where students work on real projects	C, R
(Stoica & Islam, 2012)	The paper presents a model that combines classroom teaching of theoretical concepts and practice by solving real-world projects	R

C: Collaborative learning; G: Games and gamification; A: Agile methods; V: Global and virtual teams; R: Real projects and industry links

Appendix 5. Scope of the studies excluded from the network of trends

Reference	Scope of the study
(Rombach et al., 2008)	A quantitative study that assesses the benefits of disciplined software development on the individual level and provides recommendations with PSP and TSP as teaching tools
(Krutz et al., 2015)	Experiences in teaching software engineering to Deaf/HoH students are described
(Sedelmaier & Landes, 2015)	Presents a proposal to assess software engineering students competencies
(Lee et al., 2012)	Present the status of a two-phase-eight-year nation-wide effort in improving the software engineering education in Taiwan
(Ribaud & Saliou, 2009)	Revealing Software Engineering Theory-in-Use through the Observation of Software Engineering Apprentices' Course-of-action
(Dumslaff, 2008)	Description of knowledge transfer from innovative solutions into capabilities for a company
(Xudong et al., 2008)	Discussion about an undergraduate course on software architecture
(Distante, 2007)	Presents the experience gained from teaching courses that involved hearing-impaired students of an undergraduate software engineering and a programming language course in two different universities
(Jaakkola et al., 2006)	Discusses approaches in SE curriculum development and general good practices to improve it
(Marques, 2015)	Evaluates if mixed-gender software teams have better project results than one-gender teams
(Tafliovich et al., 2015)	Presents student perspective on the evaluation of software development team projects
(Fairley & Willshire, 2011)	Describes some concepts that should be taught in software engineering and ways to introduce them in the curricula
(Sudol & Jaspan, 2010)	Presents a methodology that approaches interactions among the misconceptions of software engineering based on a forced-choice paradigm and the strength of the misconceptions
(Hazzan & Dubinsky, 2009)	Discusses ways by which a reflective mode of thinking may assist software engineers in improving professional skills
(Xudong et al., 2008)	Presents the content and pedagogy of a software architecture course for undergraduate students

Appendix 6. Data from the students' survey on the preliminary framework ASEST0

Responses of participants in quasi-experiment 1
We learned how to better communicate in a team given opportunities for all to express the ideas to reach the objectives and avoid conflicts. I feel better prepared to interact with my teammates and get the project goal. Team member contribution assessment helped to put riders in evidence, which I believe was truly helpful.
I believe everything was very useful and we are better prepared now as a team. I liked the most the role gaming. I think it is an interesting way of learning and we are better prepared now to solve conflicts in our team. I think it would be good to have some way to know what rules are helping more.
I believe that as a result of these activities our team is more united. Team members care more about listening and supporting others. To continue improving I recommend checking the fulfillment of the agreement frequently.
I liked all the activities we did. We learned how to work in teams. We must perform in teams along with the program and we never before had focused training on how to do it well. I wish software engineering projects continue with real clients.
The course was pretty dynamic and its activities helped out the team a lot. By using rules and assessing how others were contributing help us to prevent future conflicts and to work united.
The members of my team found useful all the activities. We learned how to work in teams, communicate better and solve conflicts. We are better prepared to face new challenges united. I would have liked more sessions regarding role-playing gaming. It was a fun way to learn.
We learned how to prevent conflicts by using clear team rules. But sometimes it was difficult to know if some of them were working as they were supposed to. Maybe is better to focus on a few rules. Then add new ones when we can see their effects.
Checking how others are contributing is a good way of preventing conflicts. However, when we discussed it in my team we saw some people did unfair assessments. I think this assessment should do it several times to overcome opinion differences. I appreciate having the opportunity of working on a real project; even our client was not always available to clarify doubts.
It was very helpful to learn how to effectively perform in teams interacting with real clients. But I didn't like we had to fill too many questionnaires.
I would like to know how Belbin's roles match software team roles. I think that would be better for role gaming. I liked to work on real projects.
The activities, especially those related to the use of rules were very much helpful to my team. It was a bit difficult to reach what rules the agreements should include. I think more guidance is needed on that matter.
I enjoyed the role-playing game the most. It made us see things from a real perspective. It was also good to take us closer to the companies' experience by developing projects with real clients.
Too many questionnaires to fill, even I think they were helpful. I think it is important for agile teams to do this kind of training.
I believe all the activities that we did are important to learn how to work in software teams. I feel better prepared to interact with my teammates and to achieve the project goals.
I found the framework helped us to learn how to succeed in developing projects as a true team. It was great to have the opportunity to train our team to improve its performance.
It was a good experience. Using rules agreement helped to work in teams showing more respect and understanding. I would like to do it again.
It was a good experience for my team. We learned a lot about teamwork. I think rules are helpful but they should focus more on project tasks. In some meetings, we almost spent more

	time discussing rules than the project itself.
	Generally, my team enjoyed the activities. My teammates showed more collaboration after getting trained on team working skills. It was very helpful.
	I believe it is important for us, as future software engineers, to know how to perform in teams in effective ways solving real problems. I think the training on team working skills should include more activities because of its importance to make the rule agreements.
	Rules are helpful if everybody sticks to the agreement. For my team, some mates didn't engage in that from the beginning. However, after the member contribution assessment teammates compromised and participated more.
	Responses of participants in quasi-experiment 2
	I believe the framework is helpful to software teams. The training on team working skills was too short. I consider the role gaming was great and it can help even more including other similar activities.
	It was very helpful to learn how to perform in teams. Software engineering methodologies by themselves are not enough to guide us on this. Thank you very much for allowing us to learn by applying this framework. I think the company would benefit if all teams take this training.
	These activities were very good for improving our teamwork. We were not truly aware of how much our team needed this before making the activities. We now know better our team and the directions we still need to continue improving. Rules make our team live easier. We will continue applying this as a daily practice in our team.
	I liked the role game. It was good to assess how we were contributing. Having an agreement on how the team should collaborate was very helpful.
	I very much like the experience. I think it should be several cycles to continue improving. We will continue applying the agreements in our team.
	I especially liked to know my Belbin role and play the role game. In general, I think everything was dynamic and useful. Excellent that everybody saw the evaluation on his-her contribution, in those way riders got uncovered.
	I value how these activities have helped us to improve as effective software engineers team players. I think the agreements help to get project goals as a truly united team. I recommend the framework to the other teams of this company.
	I believe our team improved in several ways by applying this framework. The role gaming was too short but a fun way of learning how to perform in teams effectively. I think the questionnaires helped us to better understand our difficulties individually and as a team.
	I think team rules are truly helpful. These activities helped us to get to know each other and to identify our team problems and strengths. It is good to know what others think about our contribution to the team.
	This framework helps us to know how we could contribute in a proper way to the team. The team members know now what other team members expect from them regarding their behavior in working as part of the team. I believe this foments respect and collaboration.

Appendix 7. Comparison of ASEST0 to existing teamwork teaching frameworks

Approach	Addressed Software engineering education Trends	Addressed teamwork factors	Learning scenario	Preparation on team working skills	Team rules identification establishment	Team Member contribution assessment
ASEST0 (this implementati	Collaborative Learning	T. Rules Cohesion Performanc	Use of rules to regulate team	Individual diagnosis and training	Cooperative team rules agreement	Assessed on five areas,

on)	Real project resolution and links with industry Games and gamification Agile methods	e Behaviors Performance outcomes	behaviors. Role–play gaming for training team working skills. Development of capstone projects to solve real world problems using agile methods	by role-play gaming for solving simulated conflicting situations	that norm communication and conflicts resolution	according to high, medium and low levels of team performance behaviors
Meeting-flow (Chung-yang Chen & Teng, 2011) (C. Y. Chen & Chong, 2011)	Collaborative Learning Real project resolution and links with industry	Meetings flow Teamwork quality Product quality effectiveness and Project process efficiency	Modeling of teamwork and stakeholder involvement in functional meeting classes. Organization of the meeting classes flow	N/A	N/A	N/A
TeC (Alsaedi et al., 2016)	Collaborative Learning Games and gamification	Coordination Performance outcomes	Gaming to recreate the fundamental team coordination requirements in a simulated environment	Multi-way communication, cooperative goals, real-time stress situations, promote better coordination and communication skills	N/A	N/A
Learning by osmosis (Lago et al., 2012)	Global and virtual teams	Performance outcomes	Learning complementary topics by working together with those who are assumed to be experts	N/A	N/A	N/A
(Garcia and Pacheco 2014)	Collaborative Learning Real project	Effort Productivity Performance outcomes	Combination of TSPi teamwork methodology and PBL	N/A	N/A	Teacher and students produce peer and

	resolution and links with industry		supported by a virtual platform			team evaluations
(Silvestre et al., 2016)	Collaborative Learning	Team building Communication Coordination	Heuristic to design student teams	N/A	N/A	N/A

Appendix 8. Excluded papers from literature review 2

1.1 Excluded papers not available

Authors	Title
(Nizami, 2007)	Global Software Development and Delivery

1.2 Excluded full books of proceedings

Conference	Year
11th IEEE International Conference on Global Software Engineering Companion	2016
10th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing	2014
International Conference on Information Systems	2013
Agile Processes in Software Engineering and Extreme Programming: 8th International Conference	2007

1.3 Excluded papers not relevant for this study

Reference	Scope of the study
(Etzkorn et al., 2002)	The paper presents a new semantically-based metric for object-oriented systems
(Kumar & Singh, 2016)	The paper addresses Aspect-Oriented Programming (AOP)
(Youssef & Capiluppi, 2015)	The article addresses internal attributes of a code base
(Mukhopadhyay, 2010)	This paper tackles software system design topic
(Kim, 2006)	The paper addresses object-oriented metrics
(Etzkorn et al., 2004)	The paper reports on a comparison of various cohesion metrics
(Yan, 2012)	The paper reports on a study that focuses on competitive ball games teams

1.4 Excluded papers not fully addressing team cohesion construct

Study	Scope of the study
(Vasilescu et al., 2015)	This article addresses how gender and tenure diversity relate to team productivity and turnover
(Silva et al., 2013)	The authors investigate what criteria are used by project managers to select individuals in building software teams and how the criteria relate to project success
(Mendonça et al., 2014)	This article reports on a study that explores how changes in team composition relate to team performance
(F. Zhang et al., 2014)	The paper report on editing patterns on software quality
(Goncalves et al., 2015)	The paper presents a project to simulate the launching of a remote sensing microsatellite
(Lutz et al., 2014)	The paper presents an undergraduate software engineering program
(Sangwan et al., 2007)	It refers to a full book that addresses the success of global development teams. It includes planning, organization structure, and monitor and control

(Holt, 2005)	The paper addresses software risk management
(Rodríguez et al., 1999)	The paper presents a tool that implements the theory of systems and biology applied to software development
(Wakefield et al., 2008)	Investigation of conflicts and leadership in distributed teams. The findings show the role that leaders have to play to manage conflicts and how communication technologies are effective in mitigating task conflicts
(Watkins, 2009)	The paper reports on the use of web 2.0 in software engineering capstone courses

Appendix 9. Scope of the studies analyzed to identify team cohesion antecedents resulting from literature review 2*

Study	Contribution
(Hoegl & Gemuenden, 2001)	The paper presents a proposal of the teamwork quality concept. It includes the following facets: balance of member contributions, mutual support, effort, and cohesion. Explores the relationship between this concept and project success. They found cohesion relates to project success
(Hoegl & Proserpio, 2004)	The article reports on an evaluation of the relationship between team member proximity and the proposed concept of teamwork quality. They found team member proximity relates to team cohesion
(Jeanne M. Wilson et al., 2008)	The paper presents a model of perceived proximity, the feeling of being close to geographically distant colleagues. The authors found perceived proximity is related to team cohesion
(H. L. Yang & Tang, 2004)	By using a social network approach the authors study the relationship between team structure and ISD team performance. They found team cohesion to be related to performance
(Acuña et al., 2009)	The relationship between personality, team processes and task characteristics and job satisfaction, and software quality is explored. They found personality, task interdependence, and task autonomy related to team cohesion
(Singh, Param Vir; Tan, Yong; Mookerjee, 2011)	The paper reports on a study of the influence of structural capital on the rate of knowledge creation in an open-source project. The authors found that projects with greater cohesion among project members were more successful. Cohesion among the external contacts of a project was found to have an inverse relationship with the project success
(Magni et al., 2009)	The effects of team-level processes on individual improvisation in complex project domains are investigated. The authors found team cohesion affects individual improvisation and moderates the influence of team behavioral integration on individual improvisation
(Karn et al., 2007)	The effects of personality and software methodology on team cohesion are studied. They found them to be correlated
(Kang et al., 2011)	Proposal of an approach for human resource allocation. They identify cohesion as one of the team level characteristics related to developers that constrain human resource allocation in their approach
(Sarker et al., 2009)	Factors that explain the reason for an individual to be considered a leader by team members in different locations in virtual teams are investigated. They found that in cohesive teams, ISD ability, contribution, and knowledge transfer were found as significant predictors of remote leadership emergence
(Lakhanpal, 1993)	The question of how group characteristics influence team performance is answered. Team cohesion was found to strongly and significantly influence team performance
.(França et al., 2014)	Through two qualitative case studies, the authors investigate software engineers' motivation in industrial practice. They found team cohesion important in a motivating working environment establishment

(S. Wood et al., 2013)	The authors evaluate the impact of agile practices and general team factors in software teams' performance. They did not find team cohesion (measured as morale and belongingness) significantly related to performance
(Rothenberger et al., 2010)	The paper reports on how implementation teams' attributes affect the adoption of enterprise resource planning (ERP) systems. They found that cohesion is not a necessary precursor to project success as its impact depends on the adoption context
(Whitworth & Biddle, 2007)	The authors explore socio-psychological characteristics of agile teams. They found agile practices to influence team cohesion due to their ability to support collective team culture
(Xuan et al., 2015)	A proposal of a methodology for identifying temporal motifs in task-oriented social networks is presented. The authors found that models based on temporal motifs can be used to more precisely relate team cohesion to programmer productivity
(Chung-yang Chen et al., 2014)	The effects on the teamwork of a team process are known as the Meeting-flow approach. The authors found the approach to significantly improve team communication and coordination, as well as balances members' contributions by giving mutual support and effort. The approach had less influence, however, on team cohesion
(Fagerholm et al., 2013)	The process of onboarding into virtual Open Source Software teams is studied through multiple study cases. Team cohesion is mentioned concerning mentoring, which was found an important factor in onboarding. The authors consider mentoring may promote cohesion within distributed teams. This asseveration was not supported however through empirical evidence
(Acuña et al., 2008)	In this study the relationship between personality, team processes, tasks characteristics, product quality, and satisfaction in software teams. They found task characteristics, conflict, and personality traits significantly relate to the cohesion of student teams
(Freitas et al., 2008)	A tool to support communication in distributed software development is presented. The authors claim that the tool contributes to maintaining team cohesion. They do not provide however empirical evidence of it
(Whitworth, 2008)	A qualitative study that explores what aspects of agile software development practices influence team cohesion. They found that while often agile methods successfully support team cohesion within a limited period and context application, when these aspects change, some problems arise
(Stawnicza, 2015)	'Teamness' or a sense of team unity in globally distributed projects is addressed through a systematic literature review. Its relationship with the use of ICT is explored through interviews
(Castro-hernández et al., 2015)	The authors evaluate the effect of feedback on team members' behaviors in a global software development project. A cohesion-based feedback module for a collaborative software system was developed. They found that the teams that used the feedback module were significantly more cohesive than those not using it
(Liu & Cross, 2016)	The paper presents the development of a model of project team technical performance through a structural equation model. They found cohesion as one of the related factors
(Mcavoy & Butler, 2009)	The paper reports on field observations that investigate failures associated with learning a new agile software development methodology in a software project team. The authors found that the desire for cohesion within the team had positive benefits for the team but also influenced the failure of the team to learn. The authors hypnotize this could be related to the desire to conform that prevented double-loop learning in the team
(McAvoy & Butler, 2005)	The paper reports on a case study that investigates changes to the development environment produced by the introduction of a new software development methodology. The authors claim that the team cohesion, and the groupthink that evolved from it, was a factor in a failure to change to the

	new methodology
(Castro-Hernandez et al., 2016)	The article reports on a study that compared interaction-based measures and their ability to predict cohesion for virtual global software teams. The measures address different types of communication similarities and quantitative communication characteristics. They found several measures in both categories were good predictors of task cohesion
(Castro-Hernandez, 2016)	The article refers to the research questions and methodology of a study to compare interaction-based measures and their ability to predict cohesion for virtual global software teams
(Rutz & Tanner, 2016)	The authors investigate factors that influence team performance in global virtual teams. The authors claim that team cohesion leads to performance in global virtual teams. As a result of this study, they also found trust climate, leadership, diversity, collaboration, shared understanding, and goal setting to influence team cohesion
(Gabelica et al., 2016)	The article addresses how teams coordinate the creation of new knowledge. The study showed how team learning behaviors and team reflexivity, driven by task cohesion, and group potency support coordination development, which in turn predict team performance
(Xue et al., 2015)	Drawing from the team-shared mental model, the authors address the influence of awareness of members' skills and perception of shared governance in task cohesion for fast-response spontaneous virtual teams. They found both factors to significantly influence perceived task cohesion. Besides, task cohesion was positively related to performance and member satisfaction
(C. Wood, 1998)	The paper reports on a study related to team code ownership, an agile software development practice. Through observations and interviews, the authors concluded that team code ownership is a feeling to be engendered not police to be decreed and high team cohesion perceptions of developers positive influence this feeling
(Dingsoyr et al., 2016)	The article reports on factors influencing the performance of software development teams. Team cohesion was found to be one of them
(Sedano et al., 2016)	The paper reports on a study related to team code ownership, an agile software development practice. Through observations and interviews, the authors concluded that team code ownership is a feeling to be engendered not police to be decreed, and high team cohesion perceptions of developers positive influence this feeling
(Jovanovic et al., 2015)	The paper presents a set of games to improve agile software development processes. It includes "team cohesion games" that are game activities focused on team building and motivation. The authors state that the proposed team cohesion games may improve ice braking, morale, relationship, team building, innovation, and expectations
(Doman et al., 2015)	An approach to team management in classroom settings is presented. The authors claim that the approach led to better team cohesion. However, no empirical data is provided to support this claim
(Weimar et al., 2013)	The article aims to study factors influencing the performance of software development teams. The authors propose an extension of the teamwork factors model (Hoegl & Gemuenden, 2001). Besides they present a study design to validate this extended model. Team cohesion is seen as one of the factors influencing performance
(Garry & Change, 2013)	The paper discusses the design of a study that aims to answer the following research question: How does group cohesiveness impact the relationship between time pressure and group decision-making quality?
(Ralph & Shportun, 2013)	The paper reports on a case study that examines abandoned Scrum implementation. The authors found that during the transition to Scrum, team cohesion collapsed. This hindered both the development performance and performance of the Scrum transition itself. From the analysis of this case

	study, the authors propose team cohesion to be addressed together with task/team familiarity and transactive memory in Scrum adoption and team performance
(Wellington et al., 2005a)	The paper reports on a study that compares software engineering student teams working with a plan-driven methodology (TSP) others using the agile methodology XP. They use two measures to assess team cohesion. One of these measures did not find significant differences between the two kinds of teams. However, the second measure revealed higher levels of overall cohesion in XP teams and higher sub-team cohesion in TSP teams
(Chidambaram & Carte, 2005)	The article presents a team interaction model that uses collaborative technologies to leverage the positive aspects of diversity and prevent the negative ones. The model is tested by comparing collocated and virtual teams. The authors found that the most diverse teams had the largest cohesion increase. However, the most cohesive groups overall were the moderately diverse collocated teams
(Wellington et al., 2005b)	This paper reports on a similar study to the one reported in (Wellington et al., 2005a). Their findings support similar conclusions about how the methodology was affecting cohesion
(Rassias & Kirytopoulos, 2014)	The article reports on the development of a tool to assist project managers in assessing the functioning of virtual teams and its evaluation through a case study. Through a literature review team, cohesion was identified as one of the main factors that influence the functioning of a virtual team
(Case et al., 2013)	A proposal of an introduction program of globally distributed teams into undergraduate software engineering courses is presented. A pilot study is briefly discussed and guidelines of the design of further research are given. The authors claim that during the pilot study the virtual teams generally rated higher on negative team processes such as emotional conflict and lower on such positive team processes as cohesion. However, the authors do not present empirical evidence to support this claim
(Shaikh et al., 2016)	The article presents a method to form software engineering student teams. The method is tested through a study that compares the levels of team cohesion in a group that received the intervention compared with another that was not intervened. The study showed the method to be effective
(D. Chen et al., 2011)	The article presents the experiences of three participants in an undergraduate research program. They discuss possible causes of the team cohesion that participated in this program. The authors consider that the fact that participants appeared to see each other to be competent programmers enhanced cohesion. However, they do not present any empirical evidence to support this claim
(De Farias et al., 2012)	The paper reports on an interview study to identify risks associated with communication in distributed software development. The authors provide a set of recommendations to develop communication. Among these recommendations, they state that promoting frequent communication and socialization at the beginning of the projects may enhance cohesion

* The studies that were identified to address team cohesion antecedents are shown in grey

Appendix 10. Excluded papers from literature review 3

1.1 Excluded papers not available

Authors	Title
Koc, Guler; Aydos, Murat	Trustworthy Scrum: Development of Secure Software with Scrum
Richenhagen, Johannes; Pinnekamp, Jochen; Konenki, Vijay Kumar; Schumacher, Max	Developing Adaptive Lighting in an Agile Software Factory

Patel, C., Ramachandran, M.	INSERT: An Improved Story card Based Requirement Engineering Practice for Extreme Programming
Patel, C., Ramachandran, M.	Acceptance test driven story card development: An improved requirement elicitation process in XP
Plesa, Sorina; Prostean, Gabriela	Knowledge Management for Model Based Design Software Products

1.2 Excluded full books of proceedings

Conference	Year
CEUR Workshop	2017
9th International Workshop on Cooperative and Human Aspects of Software Engineering	2016
16th International Conference on Software Process Improvement and Capability Determination	2016
8th International Workshop on Cooperative and Human Aspects of Software Engineering	2015
22nd European Conference on Systems, Software and Services Process Improvement	2015
IEEE 9th International Conference on Global Software Engineering	2014
European Conference on Systems, Software and Services Process Improvement and Innovation	2014
14th International Conference on Computational Science and Its Applications	2014
1st Asia Pacific Requirements Engineering Symposium	2014
20th Americas Conference on Information Systems	2014
ACM / IEEE International Symposium on Empirical Software Engineering and Measurement	2013
19th Americas Conference on Information Systems	2013
13th International Conference on Software Process Improvement and Capability determination	2013
24th Australasian Conference on Information Systems	2013
International Conference on Business Information Systems Workshops	2013
8th International Conference on the Quality of Information and Communications Technology	2012
18th Americas Conference on Information Systems	2012
Human-Centered Software Engineering - 4th International Conference	2012
4th International Conference on Human-Centered Software Engineering	2016
ACM International Conference Proceeding Series	2012
5th International Conference Evaluation of Novel Approaches to Software Engineering	2016
7th European Conference on Modelling Foundations and Applications	2011
ACM International Conference Companion on Object Oriented Programming Systems Languages and Applications Companion	2010
ICSE Workshop on Cooperative and Human Aspects on Software Engineering	2009
14th Americas Conference on Information Systems	2008
International Conference on Software Engineering Theory and Practice	2007
13th Americas Conference on Information Systems	2007
AGILE Conference 2016	2006

1.3 Excluded papers not relevant for this study

Reference	Scope of the study
(Bezuidenhout & Baier, 2011)	The paper focuses on the sugarcane production field
(Steghöfer et al., 2016)	The paper focuses on the teaching of agile methods
(Faria et al., 2012)	The paper presents a methodology for programming teaching
(Masson & Udas, 2009)	The paper focuses on open educational resources
(Joy, 2005)	The paper argues on changes needed for computer science curricula

(Baum et al., 2017)	The paper reports on a comparison of two code review techniques
(Ghobadi et al., 2017)	The paper report on the role of competitive rewards on knowledge sharing in pair programming teams
(Abdul et al., 2018)	The paper report on the experiences in partnership between a company in the energy sector and a university
(Hebig & Wang, 2017)	The paper focuses on quality assessment of measurement programs
(Plesa & Prostean, 2018)	The paper focuses on knowledge management
(Zimmermann, 2016)	The paper presents a set of architecture practices and techniques from industrial experiences and existing literature
(Aktunc, 2012)	The paper focuses on software complexity metrics
(Guetat et al., 2011)	The paper focuses on the development of Information Systems urbanization
(Shankarmani et al., 2011)	The paper proposes a method that tackles agile teams performance based on football rules
(Lopez-Nores et al., 2009)	The paper focuses on continuous integration
(Angela Martin et al., 2006)	The paper reports experiences shared in a panel on politics and religion in agile software development
(Singh et al., 2005)	The paper presents a Concurrent Versioning System (CVS) for agile development
(Lyytinen & Rose, 2005)	The paper focuses on IT innovation in agile development
(Ghani et al., 2016)	This chapter book addresses models and frameworks for agile methods adoption
(Habib, 2016)	Book on supply chain management
(Tambo et al., 2015)	The book chapter focuses on the use of feral information systems in Denmark
(Hajou et al., 2014)	The paper reports the results of a literature review on the development of software projects in the pharmaceutical industry
(Olsson et al., 2013)	The paper focuses on agile working during software evolution
(Shull, 2013)	The paper focuses on big data
(Heikkilä & Lassenius, 2011)	The paper reports literature review on release planning in globally distributed agile software development

1.4 Excluded papers with too narrow scope

Reference	Scope of the study
(Stavru, 2014)	The paper addresses the trustworthiness of the surveys that investigate the usage of agile methods
(Dullemond et al., 2009)	The paper reports on advantages, challenges, and technological support to combine agile software development and global software engineering
(Díaz et al., 2014)	The paper presents a process for agile construction and evolution of product-line architectures
(Julian M Bass, 2013)	The paper reports on practitioners descriptions on the role of product owners The findings show that product owners deal with nine responsibilities
(L. Chen, 2017)	The paper addresses continuous delivery adoption
(Moe et al., 2016)	The paper presents a framework for establishing shared knowledge in global virtual agile teams
(van Kelle, Evelyn; van der Wijst, Per; Plaat, Aske; Visser, 2015)	The paper presents a conceptual model of social factors influencing software development projects success
(Vilkki, 2010)	A presentation that reports on the use of agile methods in Nokia Siemens Networks
(Esfahani & Yu, 2010)	The paper presents a repository of knowledge on the agile

	method
(Thomas & Baker, 2008)	The paper reports on experiences on the application of agile methods and the challenges associated with business needs and investments
(Chau & Maurer, 2004)	The authors propose a set of knowledge sharing tools to facilitate inter-team learning
(Miglierina, 2015)	The paper focuses on cloud computing and present a set of tools to facilitate release processes
(Iivari & Iivari, 2011)	The paper hypothesizes about the relations of organizational culture and agile methods deployment
(Barksdale et al., 2009)	The paper presents an approach the use concept maps to team interaction on an agile usability project. The authors state that the approach contributes to mitigating team conflicts although this claim is not supported
(S. A. Licorish & Macdonell, 2013)	The paper reports the results of analysis over teams' attitudes and behaviors of practitioners working in a Jazz repository
(Fægri, 2010)	The paper reports on barriers that can influence group learning and the adoption of agile methods in software organizations
(Abdelnour-nocera & Sharp, 2008)	The paper reports on a case study organization in adopting an agile development
(Xiaohua et al., 2008)	The paper discusses the developers and customer interactions in XP projects
(L. Yilmaz & Phillips, 2006)	The paper presents the basis of an agent-based tool for simulation modeling of agile software processes
(Oni & Letier, 2016)	The paper presents the basis of a tool for deciding what features develop
(Yoshii & Higa, 2011)	The paper reports on the Japanese software development style in Offshore software development (OSD) projects
(Park & Maurer, 2010)	The paper reports on network analysis in a software development community
(Sadun, 2010)	The paper presents experiences in working with subcontractors in distributed Scrum teams
(Barney et al., 2009)	The paper reports on an experience in Atlassian software company, where the developers get a day to work in whatever they like
(Seger et al., 2007)	The paper reports on relationships between specific indices of organizational climate The paper tackles the relationship between team climate and individual self-efficacy
(de Assis et al., 2017)	The paper addresses difficulties in the adoption of Scrum methodology for companies that used to apply plan-driven methodologies
(Bendix & Pendleton, 2014)	The paper reports problems regarding coordination processes and communication
(S. A. Licorish et al., 2013)	The paper reports the results of analysis over a Jazz repository of various practitioners roles, attitudes, and shared competences
(Cagle, 2012)	The paper discusses proposals to architecture design in adopting agile methods
(Yasin et al., 2009)	The paper presents a method for designing story cards in Extreme Programming using artificial intelligence techniques
(Onions & Patel, 2009)	The paper present the design of an extension of a tool for supporting story cards acceptance
(Weaver, 2015)	The paper presents a framework for the design and installation of systems in the Human Factors Engineering Program (NUREG-0711)

(Bellenzier et al., 2015)	The paper presents a conceptual model that relates the Scrum adoption to software team productivity
(Hadar & Sherman, 2012)	The paper reports on a qualitative study on software architecture in agile development
(J.M. Bass, 2012)	The paper reports the results of a case study that showed reasons to adopt agile practices in software enterprises
(Xiong & Wang, 2010)	The paper presents a method that uses User Center Design (UCD)
(Rusnjak et al., 2010)	The paper presents a framework for eCommerce software development
(Patel & Ramachandran, 2009)	This book section presents guidelines for agile requirement engineering
(Keith et al., 2017)	The paper reports on the role of task uncertainty in advice networks for IT projects in general
(Parsons et al., 2007)	The paper analyses practices of usability engineering in agile development
(Crawford et al., 2014)	The paper presents a general discussion on the role of conflicts management in agile development
(Mclean & Jain, 2007)	The paper focuses on interoperability tests
(Tai, 2005)	The paper focuses on effective communication in agile environments
(Leitão et al., 2003)	The paper focuses on the specification of holonic control systems
(Wendorff, 2002)	The paper discusses underlying assumptions from the perspective of organizational culture that can influence agile methods application
(Torgeir Dingsøyr et al., 2012)	The paper reports on agile software development research
(Buchan et al., 2017)	The paper reports on an exploratory case study that addresses expectations on user involvement in ASD
(Shrivastava & Rathod, 2015)	The paper report on the identification and classification of risk factors in distributed software development and management techniques
(Iivari & Iivari, 2011)	The paper focuses on organizational culture and the adoption of agile methods
(Kennedy et al., 2017)	The paper reports on a literature review that focuses on the aviation industry and the DO-178C standard for safe aviation software
(Moe et al., 2009)	The paper reports on a study that observed teams transition to an agile culture

Appendix 11. Scope of the studies on personality, conflicts and task interdependence in Agile Software Development resulting from literature review 3

1.1 Studies on personality in ASD

Study	Contribution
(Acuña et al., 2009)	The relationship between personality, team processes and task characteristics and job satisfaction and software quality is explored. The authors found personality, task interdependence and task autonomy related to team cohesion
(K. S. Choi et al., 2008)	In a study on the influence of MBTI personality types on pair programming agile practice, the authors compared groups with diverse types, alike types and opposite to each other in MBTI type. They found the sub-group of subjects who were diverse in personality type showed

	higher levels of productivity than both alike and opposite groups. Besides, comparing alike and opposite groups, the productivity of the opposite group was greater than that of the alike group
(Papatheocharous et al., 2014)	This paper describes the initial general ideas of an approach to form teams based on their collective personality traits along with tools and methods to improve productivity and satisfaction of software engineers
(Layman & Williams, 2008)	The authors studied the personality types and learning styles of undergraduate software engineering students. They found no significant difference in performance (student grades) of different personality types and learning styles
(S. Licorish et al., 2009)	The paper presents a prototype software tool component to assist in team formation by providing lightweight support for personality assessment
(Venkatesan & Sankar, 2014)	The paper reports on the effects of personality profiles of paired programming students on their academic performance
(S. A. Licorish & Macdonell, 2015)	The authors explore the relationship between team communication and personality traits in global software teams. They found all personality traits were represented and no one specifically predicted involvement of members in knowledge diffusion
(Gren et al., 2017)	This qualitative research confirmed the role of personality traits for group development and maturity
(M. Yilmaz et al., 2017)	The authors found that effective team structures support teams with higher emotional stability, agreeableness, extroversion, and conscientiousness personality traits
(Balijepally et al., 2006)	The authors analyzed research literature in personality psychology and group behavior to compare the Five-Factor Model (FFM) of personality and the Myers-Briggs Type Indicator (MBTI) typology. They found the model FFM provides better measures for all factors that are measured by MBTI. In addition, FFM also allows assessing Neuroticism trait
(Acuña et al., 2008)	The authors found job satisfaction positively related to agreeableness and conscientiousness personality traits. Besides, extraversion and software product quality were found correlated
(Mazni et al., 2010)	The paper reports on an action research study in applying Extreme Programming (XP) activities and generating agile documents. The authors found that a combination of good personality types in a team influences team performance
(E. K. Smith et al., 2016)	The paper reports on a study of personality differences among software engineers regarding work practices, beliefs, and personalities. The authors did not find personality differences between developers and testers. Managers were found to be conscientious and more extroverted. They found several differences for “engineers who are listening to music and for engineers who have built a tool”. In addition, surveyed developers that choose “Agile development is awesome” were found to be more extroverted and less neurotic
(Bhannarai & Doungsaard, 2017)	The paper proposes a method to predict people suitability for the agile methodologies by studying personality traits through the application of the Neighbour (k-NN) classification technique
(Dave Bishop & Deokar, 2014)	The paper reports that some personality characteristics play a part in agile methods preference. The author a found positive relationship between extraversion and agile preference as did openness and agile preference. A negative relationship between neuroticism and agile preference was found
(M. Omar & Khasasi, 2017)	The paper presents a model to form Scrum teams. A personality behaviors criterion is used in the process to form the teams. However, the paper does not clarify how these behaviors match Scrum roles
(Baumgart et al., 2015)	The paper reports on a study of Scrum teams to identify personality traits important for agile software development success. The authors found

	agreeableness is the most important factor for developers. Conscientiousness was found the most prominent factor for the Scrum Master. Agreeableness was found to be the fundamental attribute for Product Owner
(Mazni et al., 2015)	This paper presents personality types among agile and non-agile software developers. The authors found software developers were mostly introverted personality types. They also found intuitive, thinking, and judging personality types dimensions were dominant among software developers regardless of the software methodology used
(Branco et al., 2012)	The paper discusses the influence of personality types (based on MBTI) and social relations on the outcomes of Scrum teams. The authors found that psychological profiles influence the quality, productivity, and goal achievements
(Fagerholm & Pagels, 2014)	The paper reports on the value system of experienced developers working with Lean and Agile methods. The authors compared them to human values and individual personality and found that “Lean and Agile values are connected, but not equal, to universal values and personality”
(Sfetsos & Stamelos, 2011)	This book section focuses on human resources management in agile development. The authors propose a model that uses personalities and temperaments of developers to allocate and rotate developers in applying pair programming practice
(David Bishop et al., 2016)	The authors found personality contingencies (change adversity and work style) to influence preferences for agile methods
(Young et al., 2005)	The paper reports on a study over an XP team to identify how personality traits are related to the roles performed by team members

1.2 Studies on conflicts in ASD

Study	Contribution
(Drury et al., 2012)	The authors identified conflicting priorities as one of the six key obstacles to making decisions in agile software development
(Cao et al., 2013)	The paper describes six conflicts that can arise between traditional funding processes and agile development
(Ramesh et al., 2017)	The paper presents a framework that describes conflicts and complements between cultural responses and agile practices in Eastern countries
(Monica Villavicencio et al., 2017)	The paper presents some challenges in developing real-world projects in an academic environment. The most critical was the unavailability of the client, the conflicts among the members of a team, the use of Scrum to develop a software product as this is commonly something new for students, the use of a project management tool, and the allocation of teaching staff to guide and monitor students as a way of giving them support for realizing their projects according to expectations
(R. Vijay Anand & Dinakaran, 2017)	The paper presents a model to address the problem of stakeholder conflicts. The framework uses multi-voting and binary search trees to prioritize requirements
(Moe, 2013)	The paper reports on multiple case studies that tackle process improvement in agile software development teams. The authors found the long-term quality to conflict with short-term progress
(Rodin et al., 2011)	The paper presents patterns that include the necessary expertise for an effective release process in ASD. One of the proposed patterns contributes to eliminating conflicts regarding results and schedule
(Busetta, 2017)	The paper presents a requirement prioritization tool. The authors state that the application of prioritization activity led to a consensus as the tool served to conflict negotiation
(Taylor, 2016)	The paper discusses changes in the project manager role while adopting agile methods. Through an ethnographic approach, the authors found that project managers deal with conflicts dictated by the new development

	method. While they are expected to contribute towards success, they also should delegate decision-making to the agile teams. This situation provokes issues beyond their direct control
(Ozawa & Zhang, 2013)	The paper reports on sociocultural differences between Japanese and Chinese members while adopting agile methods. The authors discuss how different values from these cultures triggered conflicting situations and their solutions
(Alhubaishy & Benedicenti, 2017)	The authors study whether positive and emotional contagion influences behavioral teams in agile development. They found positive emotional contagion minimize teams conflict
(Khan et al., 2014)	The paper presents a framework to resolve conflicts between stakeholders. The framework integrates Scrum methodology and a Win-Win requirements negotiation model to bring agility
(Sachdeva & Chung, 2017)	The paper presents an approach for handling non-functional requirements security and performance. An industrial case study showed the approach helped in conflicting requirements situations. The state that conflicts in non-functional requirements could decrease performance
(Domino et al., 2004)	The paper reports on how task conflicts influence the agile development process and its outcomes. The authors found that low to moderate levels of task conflict improve performance and high levels mitigate otherwise anticipated positive outcomes
(Salameh & Alnaji, 2014)	The paper reports on factors that negatively influence project management offices in software organizations. Conflicts between project and departmental tasks and conflicting and unclear projects prioritization were found to affect
(Pechau, 2012)	The paper describes patterns of value-based conflicts that can affect the development of agile software projects. These patterns are related to planning, monitor and control, and communication
(Pechau, 2011)	The paper describes patterns of value-based conflicts that can affect the development of agile software projects. These patterns are related to information sharing
(Hannay & Benestad, 2010)	The paper reports on interviews conducted to identify threats to productivity in agile software projects. Conflicts between organizational control and flexibility were identified as one of them
(Ramesh et al., 2012)	The paper presents a conceptual framework of ambidexterity in agile distributed development. Through case study research they found that conflicting demands between alignment and adaptability can be addressed by practices that shape performance management and social context, which are antecedents of contextual ambidexterity
(Julian M Bass, 2015)	The paper reports on a study on the Scrum Master role in organizations adopting agile methods. The findings show conflicts of interest that arises when the Scrum Master and Project Manager roles are combined in practice
(Cram et al., 2016)	The paper presents the concept of Information Systems (IS) control alignment. It reports on how the IS control dimensions (i.e. the degree to which the control environment, control mechanisms, socio-emotional behaviors, and control execution) could complement and/or conflict with one another. Through case studies research the authors identified patterns of conflicting as well as complementary control dimension
(O'Reilly, C; Morrow, P; Bustard, 2003)	The paper presents a tool that supports automatic notification for direct conflicts while several developers work on the same revision of an artifact simultaneously in a Concurrent Versions System (CVS)
(Antonio Martin et al., 2013)	The paper presents an inter-team interactions framework. Inter-personal conflicts were identified as one of the factors (ten in total) included in this framework that can negatively influence inter-team interaction speed
(Butgereit, 2017)	The paper reports on the use of quality tests to solve the social conflict

	between agile software development teams and non-agile teams
(Cao, 2012)	The paper addresses dynamic capability in delivering trustworthy software in a changing environment. Conflicts raised by trustworthy development in dynamic environments are discussed
(Brinker & Marcolina, 2016)	The paper discusses conflicts faced by two teams implementing agile and plan-driven philosophy at the same time. The authors make suggestions on how to mitigate these conflicts
(Abdelnour-Nocera & Sharp, 2012)	The paper reports the results of a case study that identified conflicts between stakeholders arising in adopting agile methods
(Chetankumar & Ramachandran, 2008)	The presents a model for story card-based requirement engineering that includes solving requirement conflicts
(Chetankumar & Ramachandran, 2009c)	This book section describes the model for story card-based requirement engineering presented in (Chetankumar & Ramachandran, 2008)
(McMahon, 2004)	The paper reports on conflicts that arise when software companies that use agile methods collaborate with others that use traditional methods. The author provides recommendations to solve them
(Gren, 2017)	The paper reports on how agile practices relate to interpersonal conflicts. The author found Iterative Development and Customer Access to be negatively related to interpersonal conflicts
(R.V. Anand & Dinakaran, 2017)	The paper presents a model for prioritizing requirements. Stakeholder uses relative weighting to prioritize in this model. The approach tackles the problem of stakeholders conflicts

1.3 Studies on task interdependence in ASD

Study	Contribution
(Kuthyola et al., 2017)	The authors study whether task interdependence and teamwork quality relate to agile development. They confirmed this relationship through a survey study. In addition, they found teamwork quality to mediate the relationship between task interdependence and project performance
(Barbosa et al., 2017)	The paper reports on a preliminary qualitative study regarding the effects of trust on task interdependence in agile development. The authors found trust to impact task interdependence

Appendix 12. Analysis of the selected studies on the learning strategies in order to identify specific approaches to be included in ASEST+

1.1 Agile methodologies addressed in the articles related to the learning strategies

	XP	Scrum	Kanban	Crystal Clear	Other*
(Viljan Mahnic, 2012)		X			
(Viljan Mahnic, 2010)		X			
(Rico & Sayani, 2009)	X				X
(Rodriguez et al., 2015)		X			
(Viljan Mahnic, 2015a)		X	X		
(Viljan Mahnic, 2015b)		X			
(Kropp et al., 2016)	X	X			
(J. J. Y. Chen & Wu, 2015)			X		
(Manamendra et al., 2013)		X			
(Teiniker et al., 2011)	X	X			

(Kessler & Dykman, 2007)				X	
(Paasivaara et al., 2014)		X			
(Damian et al., 2012)		X			
(Paasivaara et al., 2017)		X			
(T. Smith, Tull, et al., 2011)		X			
(Shahzad & Slany, 2009)	X				
(Steghöfer et al., 2017)		X			
(Mónica Villavicencio et al., 2017)		X			
(V. Mahnic & Casar, 2016)		X			
(Lynch et al., 2011b)		X			

*It refers to a new proposal not widely used

1.2 Studies addressing Scrum and the learning strategies of ASEST

Study	Project-based learning strategy	Role-playing gaming strategy	Team-based strategy	General approach
(Lynch et al., 2011b)		Lego blocks		
(Viljan Mahnic, 2012)	Capstone course			
(Viljan Mahnic, 2010)	Capstone course			
(Rodriguez et al., 2015)	Capstone course			
(Kropp et al., 2016)	Capstone course		Agile collaboration	
(Manamendra et al., 2013)	Capstone course			
(Teiniker et al., 2011)	Capstone course			
(Viljan Mahnic, 2015a)				Literature review on Scrum in SEE
(Paasivaara et al., 2014)		Lego blocks		
(Paasivaara et al., 2017)	Capstone course	Lego blocks		
(Damian et al., 2012)	Capstone course			
(V. Mahnic & Casar, 2016)				Presents a software tool not available for free use
(T. Smith, Tull, et al., 2011)	Capstone course			
(Steghöfer et al., 2017)		Lego blocks		
(Mónica Villavicencio et al., 2017)	Capstone course	Lego blocks		

Appendix 13. Analysis of the selected studies on cohesion antecedents in order to identify specific approaches to be included in ASEST+

1.3 Methodologies addressed

Study	XP	Scrum	General approach

(Abdelnour-Nocera & Sharp, 2012)	X	X	
(R. Vijay Anand & Dinakaran, 2017)			Algorithm for requirement prioritization
(R.V. Anand & Dinakaran, 2017)			Algorithm for requirement prioritization
(Drury et al., 2012)		X	
(Khan et al., 2014)		X	
(Busetta, 2017)			Tool for requirement prioritization not available for free use
(Chetankumar & Ramachandran, 2009a)	X		
(Chetankumar & Ramachandran, 2009b)	X		
(Chetankumar & Ramachandran, 2009c)	X		
(Sachdeva & Chung, 2017)		X	
(Young et al., 2005)	X		
(Mazni et al., 2015)			Personality type preferences (based on MBTI) for agile developers
(Baumgart et al., 2015)		X	
(S. Licorish et al., 2009)			Tool to identify personality (in) compatibilities not available for free use
(M. Omar & Khasasi, 2017)		X	
(Papatheocharous et al., 2014)		X	

1.4 Excluded studies

Study	Conflicts	Personality	Reason to exclusion
(Abdelnour-Nocera & Sharp, 2012)	X		Through a case study, some types of conflicts are identified. It is not clarified however how to handle them
(Drury et al., 2012)	X		Through case studies, conflicting priorities between stakeholders are identified to influence decision-making. It is not clarified however how to handle them
(Sachdeva & Chung, 2017)	X		Focuses on security and performance non-functional requirements for big data and cloud projects
(M. Omar & Khasasi, 2017)		X	Does not clarify how to match personality traits-roles
(Papatheocharous et al., 2014)		X	Reports the first general ideas of their approach

Appendix 14. Data from the teachers' survey on the final framework ASEST+

1.1 Teachers' assessment on practicality and acceptability

Item	Score Averages
Practicality: the likelihood that the approach could be effectively used in agile software engineering education.	

1. Would this approach be easy to implement?	4,5
2. Would this approach take an unreasonable level of effort?	1
3. Would this approach be difficult to learn?	1
4. How confident are you that this approach would fit in with the program?	5
Acceptability: whether the teachers thought the approach would be accepted by their colleagues.	
1. Would you encourage the use of this approach as part of the program?	5
2. How likely is it that the other teachers of the program will adopt the use of this approach?	4

1.2 Teachers impressions and concerns

Impression: the teacher's overall perspective of the approach
1. What are your thoughts about the use of this approach? *Teacher 1:* I think the steps and phases of the framework are well structured. The materials provided for its application were clear. To have participated in a previous application of ASEST+ helped me to fully understand this proposal and how to apply it in my courses. I felt confident to do the activities with ease. The students showed high motivation. I believe ASEST+ has tremendous value for our program and to the education of software engineers in general as teamwork is extremely important for software engineers. Besides, the framework covers critical issues for software teams, especially for agile development. I believe that with more recommendations to adapt the framework to different contexts, this proposal would be more easily adopted by other teachers *Teacher 2:* I think the ASEST+ framework is a very good proposal to educate software engineers. As the main professor of the discipline “software engineering” of our program, I will propose to our colleagues to include ASEST+ in other courses as well. In the beginning, I was a bit hesitant about the use of rules. I thought the students would not stick to the agreements. But instead, I could see that students liked to have some way to norm the teamwork. I now believe that rule agreement are an effective mechanism to guide team behaviors
2. Do you see potential benefits in the use of this approach? If so, what are a few? *Teacher 1:* ASEST+ combines the current approach to teach students how to work and stick together as a team. It levers up teamwork in an agile environment and its application does not require too many resources *Teacher 2:* ASEST+ is a novel teaching framework that can potentially benefit the quality of our graduates. This framework considers how several important aspects for software teams like conflict management, personality types among others, have been addressed in the industry. Therefore, the students can be better prepared to face their professional life while working in teams. The course design considers industry experiences as well and is easy to adapt to agile development courses
3. Do you see potential costs in the use of this solution? If so, what are a few? *Teacher 1:* I don’t see any costs beyond the time required for teachers to be prepared to apply it, adapt the framework to their courses, and prepare some materials *Teacher 2:* The Lego blocks needed for the training are too expensive in our country if not available at all. The

activities could take more time than the initially planned if not well scheduled

4. Do the potential benefit(s) will outweigh the potential cost (s)?

Teacher 1:

Yes, for sure. It would help to share recommendations from other teachers already applied the framework

Teacher 2:

Definitely yes. The Lego blocks could be substituted with lower-cost materials. Some activities could be assigned to be done outside of the classroom

5. Do you see the ASEST+ as a more strategic approach (addressing key issues) or brute force approach (merely another approach)?

Teacher 1:

I think ASEST+ is a strategic approach as teamwork is the main aspect of software development. Besides, it includes current trends from software engineering education and practices from the industry as well. Therefore, ASEST+ is in line with current pathways to deliver well and updated engineers to the industry

Teacher 2:

I consider ASEST+ as a strategic approach as it tackles the main issue for software engineering in a novel way. Moreover, I think it is more strategic for our program as teams generally remain working together along with several courses. I see ASEST+ as a flexible approach to be adapted in several courses

Concerns: what aspects of the approach might cause them to hesitate using the approach

6. What do you dislike about the solution?

Teacher 1:

I don't have any complaints. I found ASEST+ easy to apply and innovative. Besides students felt motivated and showed good performance

Teacher 2:

Sometimes it is a bit difficult to track the information generated individually and collectively. It would help to have software to support information management along with the steps

7. What modifications would you make to the approach?

Teacher 1:

It would be of great benefit to provide some guidance to teachers to adapt and evaluate the learning environment. I believe that involving students to identify opportunities to improve might be good as well

Teacher 2:

I would highlight the most effective rules each time they are evaluated (for instance putting them in a visible place) in such a way the team feels them part of their achievements. A software tool would facilitate tracking the process and managing information. This could allow incorporating mechanisms to recognize patterns of behaviors, anticipate changes needed, etc