Improving Teamwork in Agile Software Engineering Education: the ASEST+ Framework

Daymy Tamayo Avila, Wim Van Petegem, *Member, IEEE,* Monique Snoeck, *Member, IEEE*

Abstract—**Contribution: This paper presents Agile Software Engineers Stick Together (ASEST+), an improved version of a framework called ASEST that aims to develop team cohesion, leading to better team learning and software engineering student teams. Background: Effective teamwork is crucial for agile software development's success and is, therefore, a key topic of current software engineering education. In previous work, a preliminary proposal for ASEST+ was presented. Here, an improved version, more suitable for agile practice education and considering cohesion antecedents, is described. Intended outcome: A teaching-learning framework to support teamwork in agile software education. Application design: ASEST+ 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 agile practices. ASEST+ addresses personality traits, conflict resolution, and task interdependence as the antecedents identified as the most important. Findings: 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 the control group.**

***Index Terms*— software engineering education, teamwork training, team cohesion, team performance, team learning, team dynamics, team development, team-based learning, agile software development**

I. Introduction

The education of software engineers to effectively operate in agile teams is extremely important given the popularity of agile methods in the software industry [1]. Agile software development (ASD) bases its success on self-managed teams able to prioritize and quickly respond to changes and satisfy customers and users through early and continuous quality software delivery. The construction and validation of the ASEST (Agile Software Engineers Stick Together) framework were discussed in [2]. ASEST aims to develop and enhance the cohesion of agile teams to improve team learning and performance. Team cohesion is related to the team's social attachment and its connection to the project itself, both at the individual and team levels [3]. Team learning is "activities carried out by team members through which a team obtains and processes data allowing it to adapt and improve" [4], while team performance is the "degree to which the team satisfies client needs and expectations" [4]. This framework's fundamentals were established by identifying teamwork trends in software engineering education, for example, collaborative learning, games, and gamification, among others [2]. The first version of ASEST (hereafter referred to as ASEST0) was developed following an IMO (input-mediator-outcome) research model [5] adapted to software engineering education. This model is widely used to study the factors affecting teamwork effectiveness [5]: the inputs are antecedents enabling and constraining team member interactions. The mediators are emergent states influencing input-output relationships and the outputs of the team activity's results and by-products. Guided by this model, ASEST0 was established. 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. Establishing and agreeing upon team rules was found in a meta-analysis among the most important antecedents for well-functioning student teams in higher education [6]. ASEST0, however, did not include the antecedents of team cohesion.

The core of ASEST0 is establishing agreed-upon rules related to communication and conflict resolution to regulate team behavior. ASEST0 was tested using two studies [7] [2]. The work in [7] reports on the results of a pilot 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. The work in [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. Despite their limitations, these studies showed ASEST0 to be effective. However, they also contributed to identifying difficulties with the approach. The research in this paper reports on an improved version of the ASEST0 framework: ASEST+. The ASEST+ framework addresses the previously identified difficulties and the lack of cohesion antecedents.

The rest of this paper is structured as follows. Section II briefly describes the ASEST0 framework and explains the identified difficulties. Section III discusses the methodology followed in this work. Sections IV and V report on the studies conducted to identify improvements. Section VI describes how these studies' findings were incorporated into ASEST+.

Section VII reports on a new quasi-experiment performed to validate ASEST+. Section VIII discusses some limitations of this research. Section IV concludes the paper and points to future work.

II. The ASEST0 framework

The ASEST0 framework combines team-based learning, project-problem-based learning, and role-playing game learning strategies in three phases and eight steps (a more detailed description of ASEST0 can be found in [2]). Its components and activities are also included in Appendix I of this paper, i.e. all items NOT underlined. The first phase aims to establish a learning environment based on agile teamwork and project-problem resolution. Teams are formed, and capstone projects are assigned. The students identify by themselves which roles they want to assume, and they self-coordinate task assignments. The students diagnose their teamwork skills (step 2) with the Team Knowledge Test (TKT) questionnaire [8]. Next, training (step 3) based on a role-playing game and focused on communication and conflict resolution aims to improve their teamwork skills. Conflicting situations are simulated, and the students are asked to solve them cooperatively by playing different software engineering roles in the team. Belbin's team roles [9] are used as well.

The core of the framework is the second phase, i.e., the implementation phase. After evaluation of team functioning (step 4) via the Team Process Check (TPC) questionnaire [10], an agreement on team rules that support communication and conflict management (step 5) attempts to further improve the team's collaboration. The team functioning diagnosis and the individual diagnoses (step 2) and lessons learned from the teamwork training (step 3) assist students in writing the agreements. Step 6 aims at the establishment of these agreements.

The third phase focuses on adjusting the agreement. A self and peer evaluation of team member contributions via the Comprehensive Assessment of Team Member Effectiveness (CATME-B) questionnaire [11] (step 7) serves as feedback for updating the agreements (step 8). This evaluation on member contributions aims to make students aware of the team's successes or failures with regard to collaboration and avoid conflicts that can arise from social loafing, that is, team members not carrying their weight in the work effort. According to high, medium, and low levels of team performance behaviors, team member contributions are evaluated in five areas, as proposed by [11].

In order to get a better understanding of the perceptions of the students who participated in the studies that were conducted to test ASEST0 [7] [2], a survey with a random sample of participants was conducted. As detailed in [2], participants expressed appreciation for ASEST0 but also pointed to some areas that required improvement. The students' most important requests concerned the need for more guidance in reaching and tracking the rules agreements and having the rules be more focused on project tasks. Indeed, despite the use of team rules agreements that supported self-organization (an important characteristic of agile teams [12]), the rules agreements in ASEST0 focused on teamwork rather than agile practices. Students recognized that, even though not all members engaged with the agreements from the beginning, after evaluation of team members' contributions, teammates compromised and participated more. However, the assessment was considered by at least one student to be at times unfair, and, further, participants proposed evaluating contributions several times to overcome this sort of criticism. ASEST+ aims to solve these difficulties, as will be described in the following sections.

III. Methodology

The methodology followed in this work is presented in table 1 and Fig 1. Table I summarizes the research questions, design, and techniques used in this work. Figure 1 shows the research process.

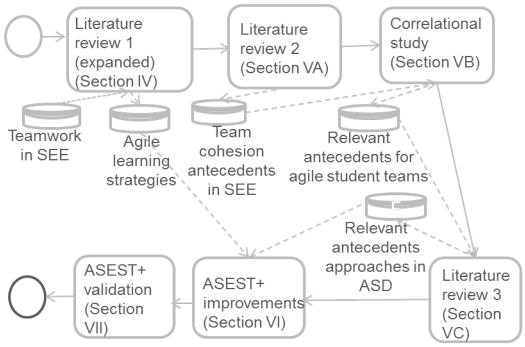

Fig. 1. Research process

TABLE I. Research questions, research design, and data analysis techniques used in this work

Section	Research questions	Research design	Data analysis techniques
IV	What are approaches regarding Scrum teamwork in SEE to be included in ASEST+ to improve its learning strategies?	Literature review 1 (expanded)	Descriptive (narrative) synthesis [13]
VA	What are the team cohesion antecedents for collocated teams in SEE?	Literature review 2	Descriptive (narrative) synthesis [13]
VB	What are the most relevant identified antecedents for agile student teams?	Survey, Correlational study	Analysis of variance [14], Person correlation analysis [14] Multiple regression analysis [14]
VC	What are approaches in ASD regarding the relevant antecedents to be included in ASEST+?	Literature review 3	Descriptive (narrative) synthesis [13]
VIII	Does the application of ASEST+ improve team cohesion, team learning, and team performance as perceived by software engineering student teams?	Quasi-experiment	Shapiro-Wilk test [14] Mann-Whitney U test [14]

*SEE: Software Engineering Education

The identification of improvements for ASEST+ proceeded

in two directions. The first stream of improvements aims to solve the difficulties identified from the studies [7] [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 a new quasi-experiment.

IV. AGILE LEARNING STRATEGIES

This section identifies approaches regarding agile teamwork in software engineering education (SEE) that enhance ASEST+ learning strategies, offering improvement over the strategies of ASEST0 (summarized in Table II).

TABLE II. SELECTED STUDIES AND IMPROVEMENTS ON LEARNING STRATEGIES

Study	Contribution/ Criteria	Improvements
[15]	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
[16]	Presents an adapted educational framework to teach Scrum. The original framework [17], 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
[18]	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 [19], [20], [21]	The phases of this Lego game are used to set up the main flow of the game during training (Step 3)
[21]	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

The original literature review reported in [2] has been expanded and focused on Scrum. The original review showed Scrum to be the most widespread methodology used to teach ASD, concurring with the review in [22]. An industrial report on the latest state of ASD shows the use of Scrum is a trend for companies as well [1].

The expanded literature review includes studies on teamwork in SEE focused on Scrum and published after the date of the original review. A search for publications containing the terms "Scrum," "software engineering education," and "team" in their title, abstract, or keywords and were published in 2017 yielded three additional studies [16], [20], [21]. In all, 15 studies focused on Scrum were (re)analyzed, 12 of which were included in the original literature review. Table II summarizes the selected studies' main contribution to improving ASEST+, the criteria leading to their selection, and how the identified improvements are included in ASEST+. Section VI further explains how these approaches were combined.

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.

V. TEAM COHESION ANTECEDENTS

This section contains two literature reviews and reports on a correlational study, all of which were conducted to identify further directions for improvement of ASEST+ regarding team cohesion antecedents.

A. Team cohesion antecedents in SEE

The first literature review is focused on identifying cohesion antecedents for collocated teams in SEE. A preliminary discussion of this literature review based on a Web of Science database search can be found in [23]. That study was expanded by adding Scopus as a research database using the same query terms as in [23]: TOPIC: (cohesion and team* and (software engineering or software development)). This led to the analysis of 23 additional papers.

A total of 10 antecedents were found to be studied for student teams: software engineering methodology [24], [25], personality [25]–[27], conflicts [26], [28], task interdependence [26], [27], task autonomy [26], [27], feedback [29], communication [30], meeting-flow [31], diversity [32] and team formation [33] (the last two antecedents were found through the expanded review).

Six antecedents were finally selected to be studied: software engineering methodology, team formation, personality, conflicts, task interdependence, and task autonomy. While some of the others may be relevant, they were not included because they pertain to virtual teams, whereas agile methods were originally developed for collocated teams [12]. While agile development is not restricted to collocated teams, studies have demonstrated the effectiveness of agile methods in collocated environments [34], showing better results in crucial issues such as productivity [35]. The antecedents 'diversity' and 'meeting-flow' had been studied for collocated teams in educational settings but were not included for the following reasons: the study in [32] found that diversity was more significant for teams supported by collaborative technologies;

meeting-flow was removed from the list because it does not refer to a specific construct but rather to a whole framework, including several concepts and approaches.

B. Relevant antecedents for agile student teams

A survey addressing these six antecedents individually and team cohesion as a whole was presented to a sample of 61 software engineering students working in 17 agile teams. The students were performing in bachelor and master programs from the Faculty of Informatics and Mathematics at the University of Holguin. The data were collected between April and May 2017. The criteria to measure the variables can be found in Appendix II. Using SPSS version 25, relationships between each antecedent and team cohesion were tested. No significant associations were found for the two non-numerical variables: the analysis 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 III summarizes the data statistics, correlations and the results of a regression analysis performed to test relationships between the other four antecedents and team cohesion. Pearson correlation confirms a significant correlation between team cohesion and personality ($r = 0.44$; $p < 0.01$), task interdependence ($r = 0.34$; $p < 0.01$), conflicts ($r = -0.46$; $p < 0.01$). There was no significant correlation between task autonomy and cohesion ($p = 0.02$, n.s).

TABLE III DATA STATISTICS, CORRELATIONS AND REGRESSION WEIGHTS

	Mean	Std	Pearson correlation	Regression weight
			r	β
Personality	194.69	17.52	.44**	.38**
Task Interdependence	18.48	3.79	.34**	.23*
Task Autonomy	29.63	7.24	.02	.05
Conflicts	31.75	3.78	-.46**	-.36**

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

Multiple regression analysis was conducted to test if these aspects significantly predicted cohesion. This technique enables determining a correlation between a criterion variable (cohesion, in this case) and the best combination of several predictor variables [36]. The coefficient of determination indicated that 40% of the variance in cohesion was explained by the identified predictors ($R2 = 0.40$, $F (4,56) = 9.42$, $p < 0.01$). Personality was found to significantly predict cohesion ($\beta = .38$, $p < 0.01$), as did task interdependence ($\beta = 0.23$, $p < 0.05$) and conflicts ($\beta = - 0.36$, $p < 0.01$). Task autonomy was not found to be a good predictor ($p > 0.05$); therefore, it could be removed from the regression model. As a result, personality, task interdependence, and conflicts were the most relevant antecedents to improve the framework.

C. Relevant antecedents approaches in ASD

The second literature review focused on how the relevant antecedents of task interdependence, conflicts, and personality were addressed for agile development. Searches in the Scopus and Web of Science databases were performed. The search string was ("software development" or "software engineering") AND agile AND ("task interdependence" or personality or conflict). In order to assess the quality of the query, the researchers verified that studies they knew to be relevant for this search (e.g., [37]) appeared in the results. The search string was ("software development" or "software engineering") AND agile AND ("task interdependence" or personality or conflict). In order to assess the quality of the query, the researchers verified that studies they knew to be relevant for this search (e.g., [37]) appeared in the results. The query resulted in a collection of 147 papers (99 from Scopus and 104 from Web of Science, many of which were duplicated in both databases) published before July 2017. The papers for which the full text was not available were excluded, as were proceedings or papers that mentioned these aspects from a too narrow or irrelevant perspective for our study. A total of 52 papers were retained.

Personality has been studied concerning its influence on job satisfaction and product quality [27], [26], productivity [38], team performance [39], preference for agile methods [40], the outcome of Scrum teams [41], and pair programming [38], [42]. Individuals' personality traits, characteristics, and temperaments have been considered for the formation of agile teams [43]–[45], team structures [46], to predict whether people are suited for agile methodologies [47], to effectively allocate and rotate developers in pairs [42], [48], for group development and maturity [34] and for Scrum roles [37].

Two papers about aspects influencing conflicts in ASD were found: they discussed socio-cultural differences [49] and emotional contagion [50]. Conflicting priorities were identified as obstacles for decision-making in ASD [51]. Another study showed that conflicts among team members influence the development of real projects in an agile academic environment [16]. It was shown in[52] that task conflicts impact the collaborative software development processes and related outcomes. Other studies addressed project management conflicts [53], [54], conflicts in funding processes [55], conflicts between long-term quality and short-term progress [52], conflicts between project and departmental tasks [56], conflicting project prioritization [56], planning, monitoring and control conflicts [57], communication conflicts [57], information sharing conflicts [58], results and scheduling conflicts [59] and conflicts between organizational control and flexibility [60]. Conflicts between stakeholders [51], [61]–[63] and requirements conflicts [64]–[66] were the most addressed issues.

While personality and conflict aspects were found to be widely addressed in ASD, only two papers about task interdependence were found. One study focused on the relationships between task interdependence and teamwork quality, and project performance [67]. The other speaks of the potential effects of trust on interdependence [68]. No strategies addressing task interdependence for agile teams were found. Table IV 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+. Section VI further explains how these approaches were combined.

TABLE IV. SELECTED STUDIES AND IMPROVEMENTS ON COHESION ANTECEDENTS

Study	Contribution/ Criteria	Improvements
[37]	The findings show how personality traits based on the Five Factor Model [69] are relevant for Scrum teams. This model provides a better understanding than the Myers-Briggs Type Indicator questionnaire, the other widely used approach for studying personality traits of software developers [70]	Their proposal on the relevance of personality traits for Scrum teams is used for role allocation (step 1)
[61]	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 teamworking skills (step 3)

VI. ASEST+ IMPROVEMENTS

This section describes the improvements of ASEST+ regarding four aspects: course design, role allocation, a Win-Win Lego game, and 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 [71]. Therefore, Scrum is used as a management framework and some development practices from extreme programming (XP) are included.

The students need programming and modeling skills as prerequisites. The course design is based on an educational framework that contains inputs, guidelines and outputs [17], based on and adapted from proposals in [21], [18] and [15]. The inputs refer to the resources needed for developing the activities. In contrast to the proposal in [17], the software tools here are open to be 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 with respect to 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 V) 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 [17], technical and collaboration practices are included based on the proposal in [15]. 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). Teamworking skills for conflict resolution are included as well. Scrum is taught through a workshop based on [21] and [18]. 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 [16], this course's output includes skills in teamwork and agile practices.

Role allocation: During team formation, role allocation is based on personality traits. To that end, the criteria set out in [37] 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 [72]. In ASEST+, the NEO Personality Inventory [69] is used to assess the presence and extent of these traits (See Appendix II). 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 [73] was adopted, as can be seen in Fig. 2.

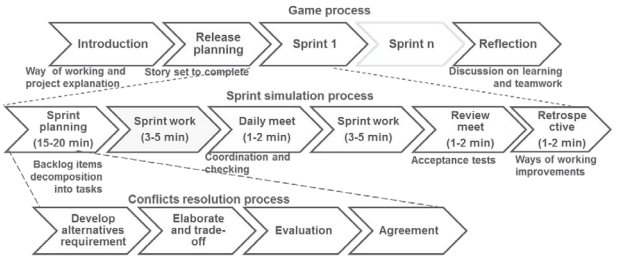

Fig. 2. Training flow

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 [61]. The release planning process incorporates the proposal of [21]. 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 in order to resolve conflicts. The teachers introduce conflicts regarding requirements priorities, introducing new requirements conflicting with the sprint backlog as well.

By means of a win-win negotiation process, team members conduct a negotiation 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 agreement, an evaluation of how the teams communicate and resolve conflicts is conducted through the TPC questionnaire [10]. 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 [74] to assist students. 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. 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 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) in order 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. Appendix I shows the final version of ASEST+ with the new activities underlined.

TABLE V: 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 factors affecting communication and conflicts in agile teams. Give examples of requirements and stakeholders conflicts	Lecture	Reading assignments, Short quizzes
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

VII. ASEST+ VALIDATION

This section reports a quasi-experiment conducted to observe the effects of ASEST+ on students' perceptions of team cohesion, team performance, and team learning. A different sample than that used in the correlational study was observed. Table VI describes the composition and demographics per subgroup for both the experimental and control groups. The experimental group was observed over eight weeks in the period September-November, 2017. A control group from the same program was observed in the period of September-November, 2019. The students were enrolled in the Informatics Engineering program at the University of Holguin in Cuba. Convenience sampling was used. All the students in the third and fourth year of the program at that time were included.

TABLE VI. GROUP COMPOSITION AND DEMOGRAPHICS

	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 students enrolled in two courses (Software Engineering II for subgroups 1, and Software Engineering III for subgroups 2), working 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 the students presented their own proposals in the control group. The projects were modules of larger projects in progress (new for the experimental group students). The experimental group students followed Scrum with some minor changes because students could not work on the project every day due to other responsibilities. The control group students used Scrum and Iconix, another ASD methodology, 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. Table VII shows the course activities per week for the experimental and control groups linked to the intervention activities (ASEST+ steps).

TABLE VII. COURSE SCHEDULES AND INTERVENTIONS

Week	Control group	Experimental group	
	Course activities	Course activities	ASEST+ steps
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
	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

Both the experimental and control group students in Software Engineering II learned about integration and acceptance tests, while in Software Engineering III, the students learned about clean code and unit test practices. In the experimental group, phases 2 and 3 were deployed two times, concurring with two project sprints. The study only covered eight weeks of the 16-week course to not 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 basically slides and books. Their assessment tasks included just practical exercises in the classroom in addition to the capstone project activities. The teaching in the control group did not include any teamwork activity beyond the capstone project development itself. These students were not coached.

The Group Environment Questionnaire (GEQ) [3], adapted to software engineering teams, was used to measure team cohesion. Team performance and team learning were measured using the instruments proposed by [4]. These instruments were 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 conducted the revision. Just few minor changes regarding cultural issues were suggested to adjust the translation for all instruments. The full adapted questionnaires are included in Appendix III.

In order to test the reliability of the instruments, Cronbach's Alpha tests were run. For the three instruments, the Cronbach's Alpha values were above the cut-off point (0.7), indicating good internal consistency (0.892 for team cohesion, 0.807 for team performance, and 0.886 for team learning). As the students had already worked together in the same teams in previous courses, the cohesion of the teams before using ASEST+ could serve as a baseline. Thus, both groups' variables were measured during the first week and right after the last session.

A Shapiro-Wilk test (recommended as the best choice for testing the normality of data [75]) was conducted on the students' perceptions for the experimental and control group at the end of the intervention period. It showed p values larger than 0.05, indicating that there is no evidence that the values correspond to a normal distribution for team cohesion, team learning, and team performance. To check the significance of increased team cohesion, performance and learning, the non-parametric Mann-Whitney U test was used. This test showed a significant difference between the students' perceptions regarding team cohesion ($p = 0.002$), team performance ($p = 0.002$), and team learning ($p = 0.003$) in the experimental group compared to the perceptions of students in the control group. Table VIII shows the means for the pre/post measurements of team cohesion, team performance, and team learning and their deltas.

For both the experimental and control groups, team cohesion, team 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. The 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). These results suggest that the increase can be attributed to the treatment, although some limitations should be considered. Section VIII dicusses the most important limitations.

TABLE VIII. MEANS AND DIFFERENCES FOR TEAM COHESION, TEAM PERFORMANCE, AND TEAM LEARNING

	Control group			Experimental group		
	Mean before	Mean after	Diff.	Mean before	Mean after	Diff.
Team cohesion	57.74	59.26	1.53	59.86	75.09	15.19
Team performance	14.21	17.90	3.69	14.64	21.62	6.98
Team learning	17.48	21.53	4.05	17.52	29.96	12.44

VIII. DISCUSSION ON VALIDITY

This section discusses the most important limitations of the validity of this work. First of all, the literature reviews are limited by the fact that unavailable papers were excluded; thus, relevant articles may have been missed. Further, the quasi-experiment's most important limitations are the sample size and not having a random sample. In order to deal with the small sample size, however, the statistical analysis included bootstrapping techniques. According to [76], 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. 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 [77]. However, [36] 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.

The quasi-experiment was also limited by the location in which the investigation took place. The local conditions and the socio-economic status and cultural background of the participants might have influenced the study. 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. Teachers were assigning projects to the students in the experimental group while the students in the control group presented their own proposals might have had an influence too. However, the intervention of teachers was necessary as the students had to develop additional new features for an existing modular project. To extend these projects, they had to delve into existing code, which increased the difficulty level of their tasks. Thus, the teachers had to intervene to guarantee the assigned work was feasible in the allotted time. Another factor to consider is the students' varying levels of expertise in the software methodologies used. While the control group students already knew Scrum and Iconix because they were already applying these methodologies in projects before the study, the experimental group students had to learn Scrum before starting the project. This favored the control group students.

IX. CONCLUSIONS AND FUTURE WORK

This paper reports on improvements to the ASEST0 framework to derive the new ASEST+ framework, a proposal to improve teamwork in terms of team learning and team performance, and the framework's validation. The improvements focus on four aspects: 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-play 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 have a dynamic nature and are established regarding communication and conflict management linked to agile practices. The results of a study of two groups of students applying ASEST+ 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.

Although this study's findings have limitations in terms of generalization to other populations, the ASEST+ framework might benefit students in other software engineering programs. In order to apply ASEST+, time constraints, resources availability, and participants' characteristics should be carefully taken into account, including their fit with limitations dictated by the program/course schedules. Availability of real 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. Keeping in mind these matters and the potential limitations discussed, it would be interesting to study the effects of ASEST+ in other situations.

ASEST+ might also help to improve similar learning environments though knowledge transfer of educational content and case studies derived from its application. In addition, the information gathered could contribute to better understand software engineering students' teams in order to further improve agile teamwork in SEE. Future work will further explore the validity of the ASEST+ framework and focus on new and larger samples. In addition, further research will explore the mediational role of team cohesion between the antecedents and the outputs.

APPENDIX I. THE ASEST+ FRAMEWORK WITH THE NEW ISSUES UNDERLINED[1]

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

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
Overview explanation of the framework (10 min)
Formation of 3-7 member teams (10 min)
Personality traits identification through the NEO PI questionnaire (30 min)
Assignation of roles based on personality traits (15 min)
Assignation of projects to the teams (5 min)
Lesson on introduction to ASD and agile teamwork (60 min)
Reading/watching assignments (Individual assignments) (30 min)
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
Diagnosis of teamwork knowledge via the TKT questionnaire (30 min, Individual assignment)
Analysis of the TKT questionnaire results (15 min, Team assignment)
Diagnosis of conflict-handling style through the TKI questionnaire. (20 min, Individual assignment)
Analysis of team weaknesses and strengths with respect to 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 teamworking 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
Reading/watching assignments on Scrum (60 min, Individual assignment)
Short quiz on Scrum (20 min, Individual assignment)
Explanation of the training (10 min)
Identification of Belbin's roles (optional, 20 min, Individual assignment)
Scrum process simulation via Lego projects. Resolution of conflicting situations (120 min)
Analysis of the teamwork demonstrated in the game (20 min, Team assignment)
Lesson on introduction on agile practices (60 min)
Reading/watching assignment (30 min, Individual assignment)
Short quiz on agile practices (20 min, Individual assignment)
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
Diagnosis of team functioning via the TPC questionnaire (10 min, Individual assignment)
Elaboration of the joint perception matrix for team functioning. (15 min, Team assignment)
Reading/watching assignment (60 min)
Hands-on laboratory on automated build and test environments (90 min)
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. 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
Setting up the agreements. Identification of team rules regarding communication and conflict resolution linked to systematic agile practices. (30 min, Team assignment)
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 agreement establishment. Agile practices application
Input: Team rules agreement, Assignment guide, Lesson contents, Scrum artifacts, Automated build and test environments, Computers and collocated environment, Scrum templates, Conflicts template
Assessment of team rules in order to track their effectiveness and improve upon agreements (15 min, Individual assignment)
Discussion in teams about rules implementation (30 min, Team assignment).
Reading/watching assignment (60 min)
Hands-on laboratory on agile practices (45 min)
Team project. Development of the project. (5 hours)
Output: Team agreement assessment report, Practical exercises solved, Product increment, Unit/Integration/Acceptance tests, Conflicts record, Scrum artifacts

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
Self and peer evaluations of team member contributions. Completing the CATME-B questionnaire. (20 min, Individual assignment)
Elaboration of the joint perception matrix including the individual average scores of each member's contribution assessment (15 min, Team assignment)
Discussion in teams of the matrix scores in order to identify problems. (30 min, Team assignment)
Team project. Development tasks. Sprint Review & Retrospective (5 hours)
Reading/watching assignments (60 min)
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
Identification of new team rules (30 min)
Updating the rules agreements (15 min)
Presentation of the project (2 hours)
Output: Team agreement updated, Project solution

APPENDIX II. CRITERIA TO MEASURE VARIABLES IN THE CORRELATIONAL STUDY

Variables	Instrument or criteria	Values
Software engineering methodology	Software engineering methodology used	Iconix, Scrum, XP
Team formation	Criteria of [78]	self-selection; random; by command
Personality	NEO Personality Inventory, Spanish version [69]	openness to experience, conscientiousness, extraversion, agreeableness, neuroticism
Conflicts	Survey of [79]	Likert scale
Task autonomy	Survey of [80]	Likert scale
Task interdependence	Survey of [81]	Likert scale
Team cohesion	GEQ questionnaire [3]	Likert scale

APPENDIX III. FULL QUESTIONNAIRES*

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 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 or other 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 [3] [4], **Reverse scored

ACKNOWLEDGMENTS

We would like to thank the participants in these studies, especially the assistant teachers Ivet Challenger Pérez and Antonio López Zaragoza, to Professors Jenny Ruiz de la Peña and Marcia E. Noda Hernández, as well as the editor and reviewers of this journal for their valuable feedback. The authors would also like to thank Enago (www.enago.com) for the English language review.

REFERENCES

[1] VersionOne, "14th annual state of agile report," 2020.

[2] D. Tamayo Avila, W. Van Petegem, and A. Libotton, "ASEST framework : a proposal for improving teamwork by making cohesive software engineering student teams," *Eur. J. Eng. Educ.*, pp. 1–15, 2020, doi: 10.1080/03043797.2020.1863339.

[3] A. V. Carron, W. N. Widmeyer, and L. R. Brawley, "The Development of an Instrument to Assess Cohesion in Sport Teams: The Group Environment Questionnaire," *J. Sport Psychol.*, vol. 7, no. 3, pp. 244–266, 1985, doi: 10.1123/jsp.7.3.244.

[4] A. Edmonson, "Psychological Safety and Learning Behavior in Work Teams," *Adm. Sci. Q.*, vol. 44, no. 2, pp. 350–383, 1999.

[5] J. Mathieu, M. T. Maynard, T. Rapp, and L. Gilson, "Team Effectiveness 1997-2007: A Review of Recent Advancements and a Glimpse Into the Future," *J. Manage.*, vol. 34, no. 3, pp. 410–476, 2008, doi: 10.1177/0149206308316061.

[6] M. Zarraga-Rodriguez, C. Jaca, and E. Viles, "Enablers of team effectiveness in higher education: Lecturers' and students' perceptions at an engineering school," *Team Perform. Manag.*, vol. 21, no. 5/6, pp. 274–292, 2015.

[7] D. Tamayo Avila and W. Van Petegem, "An experience using team rules for improving team work in software engineering undergraduate education," in *EDULEARN 2017*, 2017, pp. 2685–2690.

[8] J. E. Sims-Knight, R. L. Upchurch, T. A. Powers, S. Haden, and R. Topciu, "Teams in software engineering education," in *32nd Frontiers in Edu Conf*, 2002, pp. 17–22.

[9] J. B. Kidd and R. M. Belbin, "Management Teams - Why They Succeed or Fail," *J. Oper. Res. Soc.*, vol. 33, no. 4, p. 392, 2006, doi: 10.2307/2581652.

[10] H. Heesen *et al.*, "Assessing Team Functioning in Engineering Education," *ASEE Annu. Conf*, vol. 6, no. 2, pp. 199–216, 2002, doi: 10.1023/A:1018988827405.

[11] M. W. Ohland *et al.*, "The Comprehensive Assessment of Team Member Effectiveness: Development of a Behaviorally Anchored Rating Scale for Self- and Peer Evaluation," *Acad. Manag. Learn. Educ.*, vol. 11, no. 4, pp. 609–631, 2012, doi: 10.5465/amle.2010.0177.

[12] K. Beck *et al.*, "Manifesto for Agile Software Development," 2001. http://www.agilemanifesto.org.

[13] J. Popay *et al.*, "Guidance on the Conduct of Narrative Synthesis in Systematic Reviews." 2006.

[14] S. L. Weinberg and S. K. Abramowitz, *Statistics using spss: an integrative approach*, Second. Cambridge University Press, 2008.

[15] M. Kropp, A. Meier, and R. Biddle, "Experience Report of Teaching Agile Collaboration and Values: Agile Software Development in Large Student Teams," 2016, doi: 10.1109/CSEET.2016.30.

[16] M. Villavicencio, E. Narvaez, E. Izquierdo, and J. Pincay, "Learning scrum by doing real-life projects," in *Proc. - EDUCON 2017*, 2017, no. April, pp. 1450–1456, doi: 10.1109/EDUCON.2017.7943039.

[17] M. Villavicencio, "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, 2014.

[18] M. Paasivaara, V. Heikkilä, C. Lassenius, and T. Toivola, "Teaching students scrum using LEGO blocks," 2014, pp. 382–391, doi: 10.1145/2591062.2591169.

[19] T. D. Lynch, M. Herold, J. Bolinger, S. Deshpande, T. Bihari, and J. Ramanathan, "An Agile Boot Camp : Using a LEGO ® -Based Active Game to Ground Agile Development," in *Proc- Frontiers in Educ. Conf.*, 2011, pp. 1–6.

[20] M. Paasivaara, J. Vanhanen, V. T. Heikkila, C. Lassenius, J. Itkonen, and E. Laukkanen, "Do high and low performing student teams use scrum differently in capstone projects?," in *Proc.- ICSE-SEET 2017*, 2017, pp. 146–149, doi: 10.1109/ICSE-SEET.2017.22.

[21] J. Steghöfer, H. Burden, H. Alahyari, and D. Haneberg, "No silver brick : Opportunities and limitations of teaching Scrum with Lego workshops," *J. Syst. Softw.*, vol. 131, pp. 230–247, 2017, doi: 10.1016/j.jss.2017.06.019.

[22] V. Mahnič, "Scrum in software engineering courses: An outline of the literature," *Glob. J. Eng. Educ.*, vol. 17, no. 2, pp. 77–83, 2015.

[23] D. Tamayo, W. Van Petegem, Y. Cruz Ochoa, and M. Noda Hernández, "A correlational study on factors that influence the cohesion of software engineering students teams," in *INTED2018 Proc.*, 2018, pp. 5697–5702, doi: 10.21125/inted.2018.1354.

[24] C. A. Wellington, T. Briggs, and C. D. Girard, "Comparison of student experiences with plan-driven and agile methodologies," in *35th Frontiers in Educ. Conf.*, 2005, pp. 18–23, doi: 10.1109/FIE.2005.1611951.

[25] J. S. Karn, S. Syed-Abdullah, A. J. Cowling, and M. Holcombe, "A study into the effects of personality type and methodology on cohesion in software engineering teams," *Behav. Inf. Technol.*, vol.

26, no. 2, pp. 99–111, Mar. 2007, doi: 10.1080/01449290500102110.

[26] S. T. Acuña, M. Gómez, and N. Juristo, "How do personality , team processes and task characteristics relate to job satisfaction and software quality ?," *Inf. Soft. Tech.*, vol. 51, no. 3, pp. 627–639, 2009, doi: 10.1016/j.infsof.2008.08.006.

[27] S. T. Acuña, M. N. Gómez, and J. de Lara, "Empirical study of how personality, team processes and task characteristics relate to satisfaction and software quality," in *Proc.- ESEM '08*, 2008, p. 291, doi: 10.1145/1414004.1414056.

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

[29] A. Castro-Hernández, K. Swigger, and M. Ponce-Flores, "Effects of Cohesion-Based Feedback on the Collaborations in Global Software Development Teams," 2014, doi: 10.4108/icst.collaboratecom.2014.257332.

[30] A. Castro-Hernandez, K. Swigger, M. P. Ponce-Flores, and J. D. Teran-Villanueva, "Measures for predicting task cohesion in a global collaborative learning environment," in *Proc. - ICGSEW 2016*, 2016, pp. 31–36, doi: 10.1109/ICGSEW.2016.23.

[31] C. Y. Chen, Y. C. Hong, and P. C. Chen, "Effects of the meetings-flow approach on quality teamwork in the training of software capstone projects," *IEEE Trans. Educ.*, vol. 57, no. 3, pp. 201–208, 2014, doi: 10.1109/TE.2014.2305918.

[32] L. Chidambaram and T. Carte, "Diversity: Is there more than meets the eye? A longitudinal study of the impact of technology support on teams with differing diversity," 2005.

[33] M. K. Shaikh, A. Raza, and K. Ahsan, "Software project management as team building intervention," *J. Basic Appl. Sci.*, 2016, doi: 10.6000/1927-5129.2016.12.56.

[34] L. Gren, R. Torkar, and R. Feldt, "Group development and group maturity when building agile teams: A qualitative and quantitative investigation at eight large companies," *J. Syst. Softw.*, vol. 124, pp. 104–119, 2017, doi: 10.1016/j.jss.2016.11.024.

[35] S.D. Teasley ; L.A. Covi ; M.S. Krishnan ; J.S. Olson, "Rapid software development through team collocation," *IEEE Trans. Softw. Eng.*, vol. 28, no. 7, pp. 671–683, 2002.

[36] J. R. Fraenkel and N. E. Wallen, *How to design and evaluate research in education*, 7th Ed. New York: McGraw-Hill Higher Education, 2009.

[37] R. Baumgart, M. Hummel, and R. Holten, "Personality Traits of Scrum Roles in Agile Software Development Teams - A Qualitative Analysis," in *ECIS 2015 Proceedings*, 2015, pp. 0–15.

[38] K. S. Choi, F. P. Deek, and I. Im, "Exploring the underlying aspects of pair programming: The impact of personality," *Inf. Soft. Tech.*, vol. 50, no. 11, pp. 1114–1126, 2008, doi: 10.1016/j.infsof.2007.11.002.

[39] M. Omar and S.-L. Syed-Abdullah, "Identifying Effective Software Engineering (SE) Team Personality Types Composition using Rough Set Approach," in *Int. Symposium on Inf. Tech.*, 2010, pp. 1499–1503.

[40] D. Bishop and A. Deokar, "Toward an understanding of preference for agile software development methods from a personality theory perspective," in *Int. Conf. on System Sciences*, 2014, pp. 4749–4758, doi: 10.1109/HICSS.2014.583.

[41] D. Branco, R. Prikladnicki, and T. Conte, "A preliminary study on personality types in teams Scrum," *ClbSE 2012 Proc.*, 2012.

[42] P. Sfetsos and I. Stamelos, "Improving Quality by Exploiting Human Dynamics in Agile Methods," in *Agile Software Development Quality Assurance*, 2011, pp. 154–170.

[43] S. Licorish, A. Philpott, and S. G. MacDonell, "Supporting agile team composition: A prototype tool for identifying personality (in)compatibilities," in *Proc.- CHASE 2009*, 2009, pp. 66–73, doi: 10.1109/CHASE.2009.5071413.

[44] M. Omar and N. Khasasi, "Designing an agile Scrum team formation model," in *Conf. on Inf. Systems 2017*, 2017, pp. 145–154.

[45] E. Papatheocharous, M. Belk, J. Nyfjord, P. Germanakos, and G. Samaras, "Personalised continuous software engineering," in *RCoSE'14*, 2014, pp. 57–62, doi: 10.1145/2593812.2593815.

[46] M. Yilmaz, R. V. O'Connor, R. Colomo-Palacios, and P. Clarke, "An examination of personality traits and how they impact on software development teams," *Inf. Softw. Technol.*, vol. 86, pp. 101–122, 2017, doi: 10.1016/j.infsof.2017.01.005.

[47] R. Bhannarai and C. Doungsaard, "Agile person identification through personality test and kNN classification technique," *ICSITech 2016 proccedings*, pp. 215–219, 2017, doi: 10.1109/ICSITech.2016.7852636.

[48] V. Venkatesan and A. Sankar, "Investigation of student's personality on pair programming to enhance the learning activity in the academia," *J. Comput. Sci.*, vol. 10, no. 10, pp. 2020–2028, 2014, doi: 10.3844/jcssp.2014.2020.2028.

[49] H. Ozawa and L. Zhang, "Adapting agile methodology to overcome social differences in project members," *Proc. - Agil. 2013*, pp. 82–87, 2013, doi: 10.1109/AGILE.2013.13.

[50] A. Alhubaishy and L. Benedicenti, "Toward a model of emotional contagion influence on agile development for mission critical systems," in *Proc. - HPCS 2017*, 2017, pp. 541–544, doi: 10.1109/HPCS.2017.86.

[51] M. Drury, K. Conboy, and K. Power, "Obstacles to decision making in Agile software development teams," *J. Syst. Softw.*, vol. 85, no. 6, pp. 1239–1254, 2012, doi: 10.1016/j.jss.2012.01.058.

[52] N. B. Moe, "Key challenges of improving agile teamwork," in *Lecture Notes in Business Information Processing*, 2013, vol. 149, no. 7465, pp. 76–90, doi: 10.1007/978-3-642-38314-4_6.

[53] J. Noll, M. A. Razzak, J. M. Bass, and S. Beecham, "A study of the Scrum master's role," in *Lecture Notes in Comp. Science*, 2017, vol. 10611 LNCS, pp. 307–323, doi: 10.1007/978-3-319-69926-4_22.

[54] K. J. Taylor, "Adopting Agile software development: the project manager experience," *Inf. Technol. People*, vol. 29, no. 4, pp. 670–687, 2016, doi: 10.1108/ITP-02-2014-0031.

[55] L. Cao, K. Mohan, B. Ramesh, and S. Sarkar, "Adapting funding processes for agile IT projects: An empirical investigation," *Eur. J. Inf. Syst.*, vol. 22, no. 2, pp. 191–205, 2013, doi: 10.1057/ejis.2012.9.

[56] H. Salameh and L. Alnaji, "Challenges Leading to Projects Struggle in IT Project Management Office," *WSEAS Trans. Bus. Econ.*, vol. 11, no. 1, pp. 262–271, 2014.

[57] J. Pechau, "Rafting the agile waterfall," in *Proc. - EuroPLoP '11*, 2012, pp. 1–15, doi: 10.1145/2396716.2396731.

[58] J. Pechau, "Conflicting value systems in agile software development projects," 2011, doi: 10.1145/2578903.2579160.

[59] R. Rodin, J. Leet, M. Azua, and D. Bygrave, "A pattern language for release and deployment management," 2011, doi: 10.1145/2578903.2579147.

[60] J. E. Hannay and H. C. Benestad, "Perceived productivity threats in large agile development projects," in *Proc. - ESEM 2010*, 2010, pp. 16–17, doi: 10.1145/1852786.1852806.

[61] U. Z. Khan, F. Wahab, and S. Saeed, "Integration of Scrum with Win-Win requirements negotiation model," *Middle - East J. Sci. Res.*, vol. 19, no. 1, pp. 101–104, 2014, doi: 10.5829/idosi.mejsr.2014.19.1.11770.

[62] J. Abdelnour-Nocera and H. Sharp, "Understanding Conflicts in Agile Adoption through Technological Frames," *Int.Jour.of Sociotechnology Knowl. Dev.*, vol. 4, no. 2, pp. 29–45, 2012, doi: 10.4018/jskd.2012040104.

[63] R. V. Anand and M. Dinakaran, "Handling stakeholder conflict by agile requirement prioritization using Apriori technique," *Comput. Electr. Eng.*, vol. 61, pp. 126–136, 2017, doi: 10.1016/j.compeleceng.2017.06.022.

[64] V. Sachdeva and L. Chung, "Handling non-functional requirements for big data and IOT projects in Scrum," in *Proc. - Confluence 2017*, 2017, pp. 216–221, doi: 10.1109/CONFLUENCE.2017.7943152.

[65] P. Busetta, "Addressing team awareness by means of a requirement prioritization tool," 2017.

[66] P. Chetankumar and M. Ramachandran, "Story card Maturity Model (SMM): A process improvement framework for agile requirements engineering practices," *J. Softw.*, vol. 4, no. 5, pp. 422–435, 2009, doi: 10.4304/jsw.4.5.422-435.

[67] K. F. Kuthyola, J. Y.-C. Liu, and G. Klein, "Business Information Systems," *Lecture Notes in Business Info. Processing, vol 288*. Springer, Cham, 2017.

[68] I. Barbosa, M. Oliveira, P. Reis, T. Gomes, and F. Da Silva,

"Towards understanding the relationships between interdependence and trust in software development: A qualitative research," in *Proc. - CHASE 2017*, 2017, pp. 66–69, doi: 10.1109/CHASE.2017.12.

[69] P. T. J. Costa and R. R. McCrae, "NEO Personality Inventory," *Psychological Assessment Resources*. Spanish version, TEA Ediciones, Madrid, 2002, Odessa, Florida, 1992.

[70] V. Balijepally, R. Mahapatra, S. P. Nerur, and S. Nerur, "Assessing Personality Profiles of Software Developers in Agile Development Teams," *Commun. Assoc. Inf. Syst.*, vol. 18, pp. 55–75, 2006, [Online]. Available: https://aisel.aisnet.org/cais/vol18/iss1/4.

[71] K. Schwaber and J. Sutherland, "The Scrum Guide." 2020.

[72] P. T. Costa and R. R. Mccrae, "Domains and Facets: Hierarchical Personality Assessment Using the Revised NEO Personality Inventory," *J. Pers. Assess.*, vol. 64, no. 1, pp. 21–50, 1995, doi: 10.1207/s15327752jpa6401.

[73] M. Paasivaara, V. Heikkilä, C. Lassenius, and T. Toivola, "Teaching students scrum using LEGO blocks," in *ICSE Companion'14*, 2014, pp. 382–391, doi: 10.1145/2591062.2591169.

[74] C. So and W. Scholl, "Perceptive Agile Measurement: New Instruments for Quantitative Studies in the Pursuit of the Social-Psychological Effect of Agile Practices," in *Proc- Int. Conf. on Agile Processes and Extreme Programming in Soft. Eng.*, pp. 83–93.

[75] N. M. Razali and Y. B. Wah, "Power comparisons of Shapiro-Wilk, Kolmogorov-Smirnov, Lilliefors, and Anderson-Darling tests," *J. Stat. Model. Anal.*, vol. 2, no. 1, pp. 21–33, 2011, doi: doi:10.1515/bile-2015-0008.

[76] A. J. Onwuegbuzie, "Expanding the Framework of Internal and External Validity in Quantitative Research," 2000.

[77] P. Kline, *The Handbook of Psychological Testing Second Edition*, Second. NewYork: Routledge, 2000.

[78] R. Decker, "Management team formation for large scale simulations," *Dev. Bus. Simul. Exp. Exerc.*, vol. 22, pp. 128–129, 1995.

[79] K. A. Jehn, "A multimethod examination of the benefits and detriments of intragroup conflict," *Adm. Sci. Q*, vol. 40, pp. 256–282, 1995.

[80] E. Molleman, "Worker Flexibility and Its Perceived Contribution to Performance : The Moderating Role," *Hum. Factors Ergon. Manuf.*, vol. 17, no. 2, pp. 117–135, 2007, doi: 10.1002/hfm.

[81] G. S. van der Vegt, B. E. Emans, and E. Vuert, "Patterns of interdependence in work teams: a two-level investigation of the relations with job and team satisfaction," *Pers. Psychol.*, vol. 54, pp. 51–69, 2001.

Daymy Tamayo Avila is a doctoral candidate at KU Leuven and an associate professor at the University of Holguin. She has been teaching courses on software engineering and management for 15 years.

Wim Van Petegem is an associate professor at the Faculty of Engineering Technology at KU Leuven and policy coordinator learning technologies. He holds a PhD in Electrical Engineering from KU Leuven, Belgium. He has worked at the University of Alberta, Edmonton (Canada), at the Open University of the Netherlands and at the Leuven University College (Belgium). He is actively involved in different networks of universities like SEFI (Fellow), EDEN (Senior Fellow), and MEDEA (President). His research interests are in the field of multimedia production, new educational technology, networked e-learning, virtual mobility, lifelong learning, open and distance learning, engineering education, professional and intercultural engineering skills.

Monique Snoeck is a full professor at the KU Leuven, Research Center for Management Informatics (LIRIS), and visiting professor at UNamur. Her research focuses on smart learning environments, enterprise modeling, requirements engineering, model-driven engineering, and business process management and learning analytics. Her main guiding research themes are the integration of different modeling approaches into a comprehensive approach, the quality of models through formal grounding, model to code transformations, educational aspects of conceptual modeling and technology-enhanced learning. She has published over 130 peer-reviewed papers.