Computers in Human Behavior 36 (2014) 56–64

Contents lists available at ScienceDirect

Computers in Human Behavior

journal homepage: www.elsevier.com/locate/comphumbeh

Are learning styles useful indicators to discover how students use Scrum for the first time?

Ezequiel Scott, Guillermo Rodríguez, Álvaro Soria*, Marcelo Campo

ISISTAN Research Institute (CONICET-UNICEN), Campus Universitario, Paraje Arroyo Seco, Tandil, Buenos Aires, Argentina

ARTICLE INFO

Article history:

Keywords:
Software Engineering
Agile software development
Software Engineering education
Learning styles

ABSTRACT

Teaching agile practices is in the cutting-edge of Software Engineering education since agile methodologies are widely used in the industry. An effective strategy to teach agile practices is the use of a capstone project, in which students develop requirements following an agile methodology. To improve students' learning experience, professors have to keep track and analyze the information generated by the students during the capstone project development. The problem here arises from the large amount of information generated in the learning process, which hinders professors to meet each student's learning profile. Particularly, to know the students skills and preferences are key aspects on a learner-centered approach of education in order to personalize the teaching. In this work, we aim to discover the relationships between students' performance along a Scrum-based capstone project and their learning style according to the Felder–Silverman model, towards a first step to build the profiles. To address this issue, we mined association rules from the interaction of 33 Software Engineering students with *Virtual Scrum*, a tool that supports the development of the capstone project in the course. In the present work we describe promising results in experiments with a case-study.

1. Introduction

The adoption of agile methodologies for software development has increased for the last few years. In particular, several surveys carried out by companies, as well as scientific evidence (Ambler, 2006; Hossain, Babar, & young Paik, 2009; Paasivaara, Durasiewicz, & Lassenius, 2009; Salo & Abrahamsson, 2008), has revealed that Scrum is one of the most used frameworks in software development industry because of its potential improvements in productivity, quality, and client satisfaction. In line with industry, academia also has focused on teaching agile practices, leading Scrum to become the cutting-edge of Software Engineering education as an effective strategy to prepare students for facing challenges in professional contexts (Chookittikul, Kourik, & Maher, 2011; Devedzic & Milenkovic, 2011; Mahnic, 2010; Melnik & Maurer, 2003).

A widespread adopted strategy to teach agile practices is the use of capstone projects (Devedzic & Milenkovic, 2011; Mahnic, 2010; Mahnic & Rozanc, 2012; Schroeder, Klarl, Mayer, & Kroiss, 2012). In this kind of courses, professors make emphasis on the experience acquired by students, who are motivated to tackle real problems rather than resolve traditional exercises or tests. Students are divided into groups and are given a list of requirements (also called User Stories in the Scrum jargon) to develop a software product, whereas professors train students in skills related to problem solving, communication and project management. Therefore, capstone projects allow students to put Scrum into practice in a quasi-real controlled environment, in which the students synchronize the team-work, communicate problems and challenges and assess the quality of the resulting product through testing and validation activities. To support these activities, students are encouraged to interact with a development environment, in the same way that development teams do in the industry (Azizyan, Magarian, & Kajko-Matsson, 2011). Most of the surveyed companies that apply Scrum also use support tools to manage, test and monitor projects, since they can be a rich source of information to extract metrics and diagnosis about projects, processes and teams (Hartmann & Dymond, 2006).

In order to help students to interact with the development environment, given that they learn in many ways, professors tend to personalize the teaching by means of learner-centered principles. Thus, professors should detect the profiles of their students, the way in which they learn, their strengths and weaknesses (Dick, Carey, & Carey, 2005; McCombs & Whisler, 1997). For instance, the professor should monitor the student's performance with

* Corresponding author. Tel.: +54 2409442224553.
E-mail addresses: ezequiel.scott@isistan.unicen.edu.ar (E. Scott), guillermo.rodriguez@isistan.unicen.edu.ar (G. Rodríguez), alvaro.soria@isistan.unicen.edu.ar (Á. Soria), marcelo.campo@isistan.unicen.edu.ar (M. Campo).

http://dx.doi.org/10.1016/j.chb.2014.03.027
0747-5632/© 2014 Elsevier Ltd. All rights reserved.

Scrum by analyzing the artifacts generated, the feedback provided, and their interaction with development tools. However, the large amount of information resulting from the capstone project may prevent professors from achieving their goals (Antunes, 2010). Thus, there is a need of exploring new approaches that allow professors to bear in mind the students' different ways of learning easily.

In this context, learning styles arise as useful indicators as they are defined as the characteristic cognitive, affective, and psychological behavior that serve as relatively stable indicators of how learners perceive, interact with, and respond to the learning environment (Keefe, 1988). Many studies report that the usage of learning styles in teaching is an important factor that can improve the quality of education (Felder & Spurlin, 2005; Hawk & Shah, 2007). Among these studies, Layman, Cornwell, and Williams (2006) propose an assessment of the didactic based on learning styles and the personality types. Others (Graf & Liu, 2010; Limongelli, Sciarrone, & Vaste, 2008; Popescu, 2009; Zaina, Bressan, Rodrigues, & Cardieri, 2011) focus on applying learning styles to learning environments, such as web based environments. Thus, there is an increasing interest in analyzing the students' behavior by considering their learning preferences. Out of the learning style models, we use the Felder–Silverman Learning Style Model (FSLSM) (Felder & Silverman, 1988) since it has been widely applied in engineering education and research related to learning technologies (Kuljis & Liu, 2005).

In this work, we formulate the hypothesis that there is a relationship between the way students perform Scrum practices along a capstone project and the students' learning style according to FSLSM. On the one hand, the Scrum practices under study are the definition and specification of user stories, estimation of user stories by means of *Planning Poker*, and tracking of user stories, among others. On the other hand, FSLSM comprises four dimensions that contribute to explain the preferences that students have when they receive and process information: *perception*, *processing*, *understanding* and *input*. *Perception* relates to the type of information a student prefers to perceive; *processing* describes how perceived information is converted into knowledge; *understanding* describes the way students' progress towards understanding; finally, *input* considers the way in which students prefer to receive external information.

In this context, the first step is to know students' learning styles, which are gathered from the Index of Learning Style (ILS) (Felder & Spurlin, 2005) based on FSLSM. Then, our approach focuses on tracking students' interaction with *Virtual Scrum* (Rodriguez, Soria, & Campo, 2013), a tool that supports the Scrum process and tracks off students' prioritization and estimates of user stories, and time spent on tasks along the capstone project. To discover relationships, we mine the students' interaction log along with their learning style by applying association rules so as to show the frequency between the way in that students perform the agile practices and different styles.

To corroborate our hypothesis, we carried out an analysis of the behavior of 33 students from a Software Engineering course during 2011 at Universidad Nacional del Centro de la Provincia de Buenos Aires (UNICEN). The experimental results show there is a considerable correlation between the students' behavior when they use Scrum for the first time and their learning styles according to de FSLSM; supported by a confidence of 86.75%, a support of 22.50% and lift of 1.44, on average for the association rules.

The remaining of this paper is organized as follows. Section 2 introduces the background topics in the area of Scrum and learning styles, and also reports on related works. Section 3 presents our approach to discover the relationship between students' learning style and their performance along the capstone project. In Section 4, the results obtained from the experiments, as well as lessons learned and threats to validity, are presented. Finally, in Section 5, we state our conclusion and discuss future lines of work.

2. Background

Due to the current trend of using Scrum in the software development industry, universities have begun to teach Scrum in Software Engineering courses. Many studies have shown promising results when professors include teaching strategies that enhance students' comprehension of agile practices (Reichlmayr, 2003; Rico & Sayani, 2009). Recently, the use of capstone projects based on Scrum has been adopted as a vehicle for teaching the basic concepts in software engineering. This kind of project is developed in the classroom and supervised by professors; this strategy aims to increase student's participation in the learning process and address not only common problems found in the development of software systems but also several values proposed by the Agile Manifesto (Beck et al., 2001). Mahnic (2012) reports an experience of teaching Scrum using capstone projects in a Software Engineering course. The author identifies both the students' behavior and their perception of Scrum when performing activities, and proposes recommendations to achieve a successful capstone project; furthermore, even the behavior of students using Scrum for the first time has been observed. In the same line, Zualkernan, Al Darmaki, and Shouman (2008) proposes a simulation model in order to analyze deviations from the Scrum development process. Then, students are shown the explanations of the deviations so that they can enhance their conceptual learning of Scrum. Other authors, such as Devedzic and Milenkovic (2011), recommend some practices that are the result of the use of both Scrum and Extreme Programming (XP). These practices facilitate the adoption of the methodology and allow students to avoid several problems.

Most of the works mentioned above agree with the idea that the individual characteristics of students affect their performance in the agile development process. However, they fail to obtain evidence of how students behave toward their first contact with Scrum according to their learning preferences. This is mainly due to the omission of strategies that allow for the analysis of students' interaction with software development tools in the Scrum course. In this context, several approaches have shed light on the way students learn by considering their individual characteristics. For instance, Kay (2010) have proposed the identification of behavioral patterns from a group of students that used Extreme Programming. These patterns are oriented to groups instead of individuals. Similarly, Talavera and Gaudioso (2004) have identified patterns using data mining; however, this approach has attempted to build collaborative user profiles. Our work differs in that the analysis is focused on each student's learning styles to obtain individual preferences. Along this line, Prata et al. (2009) have detected a relationship between interpersonal conflicts and how they arise inside the course; Barros and Verdejo (2000), with the DEGREE system, have analyzed students' interaction processes from their text manipulation. However, these works fail to analyze behavior from the students' learning styles point of view. Thus, our work is based on the FSLSM, which is widely suggested for the improvement of the teaching quality Saracho (1997), Felder and Silverman (1988), and Kuljis and Liu (2005). Then, many studies have discerned the behavior of students during a course in order to identify patterns of behavior or characteristics associated to each learning style (Graf & Liu, 2010; Graf & Viola, 2009; Huang, Lin, & Huang, 2012; Ocepek, Bosnić, Nančovska Šerbec, & Rugelj, 2013; Slack & Norwich, 2007). These works have focused on the analysis of students' behaviors while performing different web-based actions such as navigating the web, using the email, and taking part in forums. As a difference, our work identifies the students' behavior associated with their learning styles when they perform agile practices in a 3D environment as *Virtual Scrum* (Rodriguez et al., 2013).

The selection of FSLSM is supported by two main reasons. Firstly, there are a lot of related works that have showed the use of the model contributes to improve the quality of learning (Carver Jr, Howard, & Lane, 1999; Essalmi, Ayed, Jemni, & Graf, 2010; Graf, Liu, Chen, & Yang, 2009; Kuljis & Liu, 2005). Secondly, the learning styles can be obtained by means of the Index of Learning Styles instrument, a questionnaire based on 44 items to which each student responds according to their learning preferences (Felder & Spurlin, 2005). The results obtained from the ILS distinguish the learners' preferences according to the four dimensions of the model, and they allow describing trends about stronger and weaker preferences by means of a numeric scale. The dimensions of the FSLSM are *perception*, *understanding*, *organization* and the *input of information*, and they are in line with the *sensing/intuitive*, *sequential/global*, *active/reflexive*, *visual/verbal* learning styles, respectively. *Sensing* students prefer to learn using concrete material, while *intuitive* students prefer more abstract material. *Sequential* students learn better in linear and well-defined steps, while *global* students prefer long steps with more freedom. *Active* students prefer doing tasks or talking about concepts, while *reflexive* students are likely to manipulate and examine the information introspectively. Lastly, *visual* students prefer to learn through images or other visual representations, instead of narratives or sounds that explain concepts as *verbal* students do (Felder & Silverman, 1988).

This learning-style model allows for a better understanding of students' differences when learning. Bearing learners' preferences in mind for Software Engineering courses will eventually enhance the quality of teaching. For this reason, our work explores the relationship between the students' behavior while they perform Scrum practices and their learning style. This work attempts to shed light on understanding the impact of the way students learn on the teaching of software engineering practices in the context of capstone projects.

3. The course under study

This section presents our approach to discover relationships between the students' performance of Scrum and their learning style. We validated the approach in a Software Engineering course that covered one semester in the last year of the Systems Engineering curriculum at UNICEN. Students attending the course have been trained in software system design, object-oriented programming, operating systems and networks and database management. The aim of the course is to teach current concepts of Software Engineering by means of Scrum and provide students with opportunities to put their acquired theoretical concepts into practice along a capstone project. This kind of project allows students to make use of their knowledge and skills on communication and management by simulating a professional context (Soria, Campo, & Rodriguez, 2012). We have chosen Scrum because it is an iterative and incremental methodology that organizes projects to make them manageable for small, self-organized and cross-functional teams, and also systematizes software projects, pursuing successful software development practices by emphasizing teamwork interaction (Schwaber & Beedle, 2002).

The left part of Fig. 1 shows our approach to conduct the course. In line with Scrum, we assign Scrum roles to professors and students. A professor plays the role of the Product Owner, who owns project requirements (i.e. User Stories) and leads teams to clarify the requirement specifications, whereas students are divided into Scrum Teams in which a student plays the role of the Scrum Master. Along the iterations (i.e. Sprints), the Scrum Teams develop User Stories assigned by the Product Owner, and the Scrum Master facilitates the process and must ensure the productivity of the team.

The course lasts 16 weeks and is divided into 4 Sprints. The first Sprint is meant to set-up the required activities to be done before starting with the project, such as loading the User Stories onto the Product Backlog, configuring the development environment and distributing the workstations to team members. In this Sprint, students are taught to perform the Scrum practices supported by the development environment. This environment is *Virtual Scrum*, which aims to help Scrum students set up a virtual working environment in which the visual metaphors required by Scrum are effectively displayed such as Product and Sprint Backlogs, Task Boards, and Burnt-down chart, among others (Rodriguez et al., 2013).

Once the training Sprint ends, the development process follows the subsequent Sprints. At the beginning of each of them, there is a meeting in which the Product Backlog is negotiated with the Product Owner so as to define the Sprint Backlog, which exposes the User Stories to be developed along a Sprint. After that, the Scrum Teams estimate the User Stories by means of Planning Poker (Cohn,

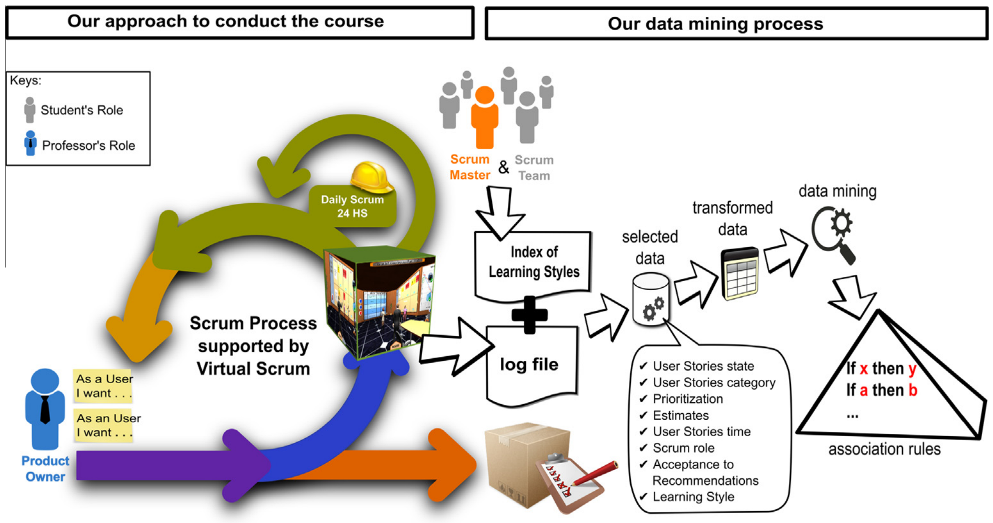

Fig. 1. Our approach to discover relationships between students' leaning style and their performance of Scrum practices.

2005). This technique is based on assigning a value of the Fibonacci series, which represents an estimate of the effort or time required for each User Story. In each Sprint, the teams are responsible for developing a working increment of the product. Each day of the Sprint, each Scrum Team meets in a Daily Meeting; during a 15-min time-boxing period, each team member answers three questions, namely, "What will you do today?" "What did you do yesterday?" and "Are there any impediments?" At the end of the Sprint, Scrum Teams hold two meetings: the *Sprint Review* and the *Sprint Retrospective*. In the first one, students present the product increment, once it has been inspected and assessed, to the Product Owner in order to obtain their approval. In the second one, the teams reflect on how they have performed the Scrum practice during the Sprint, and discuss the lessons learned and the improvements to be done in subsequent sprints. Moreover, professors provide students with a suitable context to facilitate reflection, communication and engagement taking into account the skills and preferences of students.

In the same line of thought, considering learning styles becomes relevant not only for the analysis of the students' performance when following the Scrum-based process associated with the capstone project, but also to the personalization of the Software Engineering teaching so as to maximize students' learning experience. To tackle the analysis of the students' performance, our approach uses *Virtual Scrum*, which generates a log file gathering the students' interactions from each of the supported Scrum practices. This log file contains attributes related to the following Scrum practices:

- *User Stories definition:* the execution of this practice involves the definition of the requirements as User Stories. User Stories contain title, description, state, priority, and category. The state describes the progress of User Stories from their conception in the Product Backlog to their acceptance as a working product (Table 1, row 1). The priority means the importance level of the User Story (Table 1, row 2). According to Felder, a sensing student is careful and detailed-oriented but they may be slow as well so that an expected situation could be that they updated their User Stories status to DONE by giving them HIGH priority. In the same way, the opposite situation could be expected for intuitive students.
- *Estimation by Planning Poker:* to estimate User Stories, students use the Planning Poker technique (Cohn, 2005). *Virtual Scrum* supports this practice virtually, in which students assign numeric values (i. e. story points) to User Stories according to their effort estimation. Since the numeric values are determined by the Fibonacci series and they can be discretized (Table 1, row 3). FSLSM describes that sensing students are aware of their surroundings, pay attention to details and are meticulously experimental. Thus, an expected situation could be that sensing students assign HIGH estimates to their User Stories.
- *User Stories Tracking:* students take on User Stories as the time spent on them is daily recorded in a log (Table 1, row 4). According to FSLSM, sensing students are careful yet may be slow to perform their tasks, and usually exceed the assigned time. Then, an expected situation could be that sensing students complete their User Stories during HIGH intervals of time.
- *Team Organization:* each student belongs to a Scrum Team; thus, they can perform the Scrum Master role with different responsibilities than others students (Table 1, row 5). Furthermore, the students perform either development or support tasks; development tasks are the ones related to design, implementation and testing, among others, while support tasks are related to configuration and maintenance tasks (Table 1, row 6). In the FSLSM, intuitive students are good at grasping new concepts and performing exploratory tasks; also, these students like innovation in the same way they dislike repetition. Then, an expected situation could be that intuitive students perform development tasks as a software developer does when solving a problem.
- *Assistance by Recommendations: Virtual Scrum* contains an agent that provides recommendations and textual reminders to students, which are related to what students do. For instance, when the time that students spend on completing their User Stories surpasses the estimated time, the agent indicates an overestimate (Table 1, row 7). According to the FSLSM, sensing students are likely to prefer concrete information since they observe and gather data through their senses. Then, an expected situation could be that sensing students accept all kind of recommendations.

However, the analysis of the students' performance becomes a burdensome and time-consuming task since the professor has to deal with large amount of data generated along the course. To address this issue, our approach follows a standard data mining process (Fayyad, Piatetsky-Shapiro, & Smyth, 1996) and aims at extracting new knowledge as association rules, one of the most widely spread data mining techniques. The right part of Fig. 1 shows how this process is involved in our approach. An association rule can be defined as an implication of the form $X \Rightarrow Y$, where X and Y are sets of elements also known as antecedent and consequent respectively. These elements can take any previously defined value from the logs. Therefore, this kind of rules contribute as new knowledge about the frequent behavior that students perform when they are doing their agile practices and about how this behavior is related to the student's learning styles.

Table 1
Variables under study, their possible values and meanings.

Variable	Possible values	Meaning
State	TO DO	The label assigned to User Stories from the Product Backlog to be developed
	DOING	The label assigned to in-progress User Stories along the Sprint
	DONE	The label assigned to finished User Stories in the Sprint
Prioritization	HIGH, MEDIUM or LOW	The possible values associated with the assigned level of importance for Scrum members and Product Owner
Estimation	HIGH, MEDIUM or LOW	The possible values associated with the predicted effort by Scrum members necessary to complete each User Story
Time	HIGH, MEDIUM or LOW	The time required by each Scrum member to complete a User Story
Role	DEV	The role played by Scrum members when performing development tasks
	SUPPORT	The role played by Scrum members when performing supporting tasks
Scrum Master	YES or NO	The Scrum Master role performed by the Scrum members
Recommendations	OK	The label assigned when Scrum members accept the suggestions provided by the tool
	IGNORED	The label assigned when students disregard the suggestions provided by the tool

In order to get the association rules, we take the *Virtual Scrum* log as input and select the aforementioned attributes as our variables under study. Then, we transform data in order to make it suitable for data mining by filtering out data tracked off from students that have not interacted correctly with the tool. Additionally, we augment the information about the recorded behavior with the learning style of each student obtained from the ILS. Once we have the log along with the students' learning style, we mine association rules to analyze the relationship between the learning style dimensions of FSLSM and the students' Scrum performance. To extract a set of rules, we use the Apriori (Agrawal & Srikant, 1994) algorithm with the set of instances in the database log and the learning style as a class.

To select the suitable association rules, we use three measures of interest: *support*, *confidence* and *lift*. *Support* shows the proportion of log instances containing all the items in the rule; this shows how relevant the rule is. *Confidence* can be interpreted as the estimated probability to find the consequent of a rule in a log entry when it has the antecedent too; this gives a measure of how accurate the rule is. *Lift* shows the expected proportion of the observed *support* as long as antecedent and consequent are under independence conditions. The instances in the log have different attributes related to the students' performance of agile practices such as estimation of User Stories, and time spent on Tasks, among others. Redundant rules are deleted, and only high-quality rules, with minimum thresholds of *support* and *confidence* of 5% and 60%, respectively, are taken into account. Therefore, the obtained rules could be considerably useful to help teachers bear in mind different students' ways of learning. An evaluation of our approach is presented in the next section.

4. Experimental results

To corroborate our hypothesis, we evaluated our approach after the Software Engineering course in 2012 within the Systems Engineering BSc program at the Faculty of Exact Sciences (Universidad Nacional del Centro de la Provincia de Buenos Aires). 33 students attended the course and were divided into 3 teams of 8 and another one of 9 members, who were required to develop a capstone project. The project consisted of a list of 12 User Stories of similar complexity that were distributed and assigned to the teams. The User Stories described desired features of *Universidad3D*,[1] a virtual world that allows users to navigate the facilities of the UNICEN University, play thematic games and make social activities by using chat, e-mail and forum.

From the capstone project, *Virtual Scrum* recorded a log with 434 interactions from the students. The log was augmented with the learning style of each student, which was obtained at the beginning of the course by means of the ILS questionnaire. Table 2 shows the distribution of the learning style dimensions of the course. As a result, 19 out of 33 were sensing students from whom we obtained 32.25% of the interactions, 23 out of 33 were active students from whom we obtained 66.12% of interactions, 19 out of 33 were sequential students from whom 61.29% of interactions were obtained, and 29 out of 33 were visual students from whom we obtained 95.16% of interactions. In this context, we discarded the input dimension because the number of verbal students and their interactions are not great enough to carry out the experiment, and thus, might introduce a bias towards visual students. Therefore, only *sensing/intuitive*, *active/reflexive* and *sequential/global* students were analyzed. The analysis of the obtained association rules related to each learning style dimension is described as follows.

[1] *Universidad3D* Home Page. http://www.isistan.unicen.edu.ar/?page_id=386.

4.1. Sensing/intuitive students

To visualize the association rules, we used a simplified version of the grouped matrix proposed by Hahsler and Chelluboina (2011) to facilitate their interpretation. For instance, Fig. 2 shows the obtained association rules for sensing/intuitive students. The columns of the matrix represent the unique antecedents, also known as left hand side (LHS) of the rules, while the rows represent the consequents, also known as right hand side (RHS). Both of them are grouped by the *k-means* clustering algorithm (MacQueen, 1967). In the intersection between the columns and the rows there could be a circle representing a cluster of rules; both color and diameter are proportional to the average *support* and *lift* values respectively. Then, the largest circle indicates that LHS and RHS are highly frequent in the analyzed sample. Likewise, the darkest circle indicates clusters of rules in which their items appear more frequently than expected under independence conditions. The size of the circle varies proportionally according to the support value as the color does with the lift value of the rule, whereas the intersections without circles show that there are no association rules between LHS and RHS. For example, in Fig. 2 there is a circle in the intersection of the row $perception = intuitive$ and the column $role = dev$ in the matrix, which indicates the rule $perception = intuitive \Rightarrow role = dev$. Additionally, Table 1 summarizes the variables under study.

As a result of the rule "$perception = intuitive \Rightarrow role = dev$", we observed that intuitive students frequently play the developer role. Additionally, the existence of an inverted relationship "$role = dev \Rightarrow perception = intuitive$" results in a bidirectional relationship between $perception = intuitive$ and $role = dev$. By this reciprocal relationship, we mean that the occurrence of one of the connected statements requires the occurrence of the other. The size of the circle of the latter rule shows a *support* value that is lower than in the former one, indicating that the former has more evidence in the analyzed sample. With respect to the way students develop their User Stories, intuitive students tend to complete them in a lower interval of time than sensing students($time = low \Rightarrow perception = intuitive; perception = intuitive \Rightarrow time = low$).This observation is in concordance with the FSLSM, which describes that intuitive students are faster than sensing ones to perform tasks, since intuitive students prefer to be innovative. It is worth noting that the rule "$perception = intuitive \Rightarrow time = low$" has more *support* than the former one since the size of the circle is considerably larger. Furthermore, intuitive students tend to accept more textual recommendations ($perception = intuitive \Rightarrow recommendations = OK$) than sensing students ($perception = sensing \Rightarrow recommendations = OK$). This observation stems from the fact that the size of the circle that represents *support* of the former rule is larger than the latter one. In line with the FSLSM, intuitive students feel more comfortable with symbols and words than sensing ones.

On the other hand, sensing students tend to spend more time completing User Stories than intuitive students ($perception = sensing \Rightarrow time = high; time = high \Rightarrow perception = sensing$). Remarkably, there is a high *lift* value associated with that relationship as shown by the darkness of the circle, which indicates a strong dependency between the antecedent and the consequent. This dependency suggests that the rule could be useful to predict the consequent in new data sets. As the FSLSM states, sensing students are slow to perform their tasks; this attribute may be associated with their patience to perform tasks with a high level of detail. Furthermore, students that prioritize their tasks as HIGH are frequently sensing ($prioritization = high \Rightarrow perception = sensing$). It is worth noting that there is no inversion of the rule due to the fact that there is not enough

Table 2
Distribution of students' learning styles.

	Perception		Processing		Understanding		Input	
	Sensing	Intuitive	Active	Reflexive	Sequential	Global	Visual	Verbal
Students	19	14	23	10	19	14	29	4

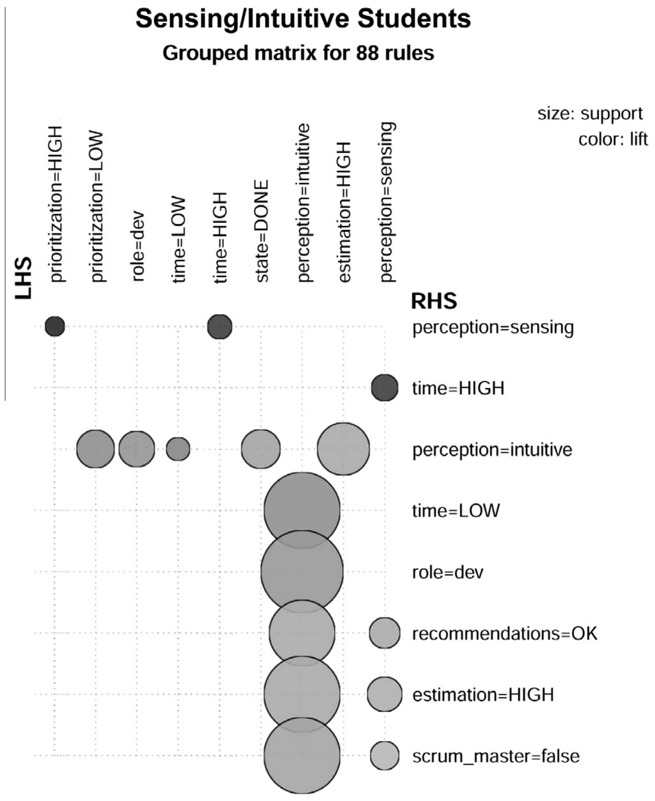

Fig. 2. Grouped matrix for association rules related to sensing/intuitive students.

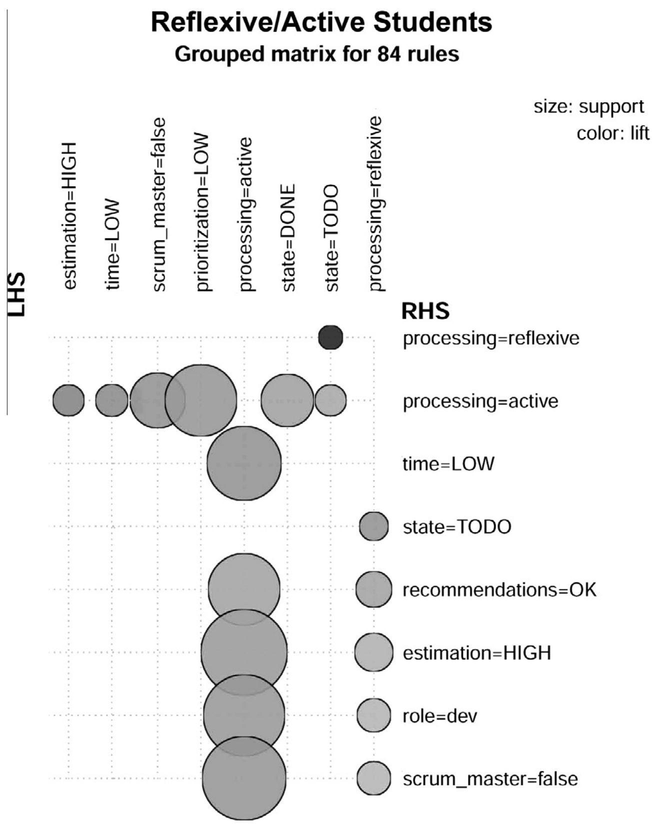

Fig. 3. Grouped matrix for association rules related to reflexive/active students.

evidence to asseverate that sensing students assign high priority to their USs.

4.2. Reflexive/active students

Fig. 3 shows the matrix that groups the association rules related to reflexive/active students. On the one hand, active students usually take little time to complete their assigned tasks, moving them to DONE status onto the *taskboard* ($processing = active \Rightarrow time = low; time = low \Rightarrow processing = active$). We attributed this result to the fact that active students prefer to process the information directly in order to research on it, validate it or experiment with it. Moreover, active students tend to both estimate highly and perform tasks associated with software development ($processing = active \Rightarrow estimation = high; processing = active \Rightarrow position = dev$). However, their tasks remain in DONE status onto the *taskboard* ($state = done \Rightarrow processing = active$). On the other hand, backed up by a high *support* value, the rules show that active students tend to take the recommendations provided by the agent of *Virtual Scrum* ($processing = active \Rightarrow recommendations = ok$) and rarely play the Scrum Mater role ($scrum\_master = false \Rightarrow processing = active; processing = active \Rightarrow scrum\_master = false$). We state that these results are in line with the idea that active students feel more comfortable if they work in groups; these students tend to take part in discussions actively, and interact proactively with their partners.

In terms of accepting recommendations provided by *Virtual Scrum*, both reflexive and active students tend to take them ($processing = reflexive \Rightarrow recommendations = ok; processing = active \Rightarrow reccommendations = ok$). However, the low *support* of the former rule shows that reflexive students seldom accept recommendations in comparison with active students since reflexive students are usually introspective and analyze the information with no external stimuli. Supported by a high *lift* value but low *support* one, the rule ($processing = reflexive \Rightarrow state = todo; state = todo \Rightarrow processing = reflexive$) shows that the tasks assigned to reflexive students usually remain in TODO status. Additionally, reflexive are likely to play the developer role ($processing = reflexive \Rightarrow position = dev$) as well as estimate with high values ($processing = reflexive \Rightarrow estimation = high$). It is worth mentioning that a high value of *lift* indicates that the facts are correlated, while a low *support* is attributed to the low number of reflexive students.

4.3. Global/sequential students

Fig. 4 depicts the matrix that groups the association rules related to global/sequential students. We observed that sequential students rarely move their assigned tasks to DONE status; instead, they remain in TODO onto the *taskboard* ($understanding = sequential \Rightarrow state = todo$). However, DONE tasks took little time to be completed by the students and also were estimated with high values of complexity ($understanding = sequential \Rightarrow time =$

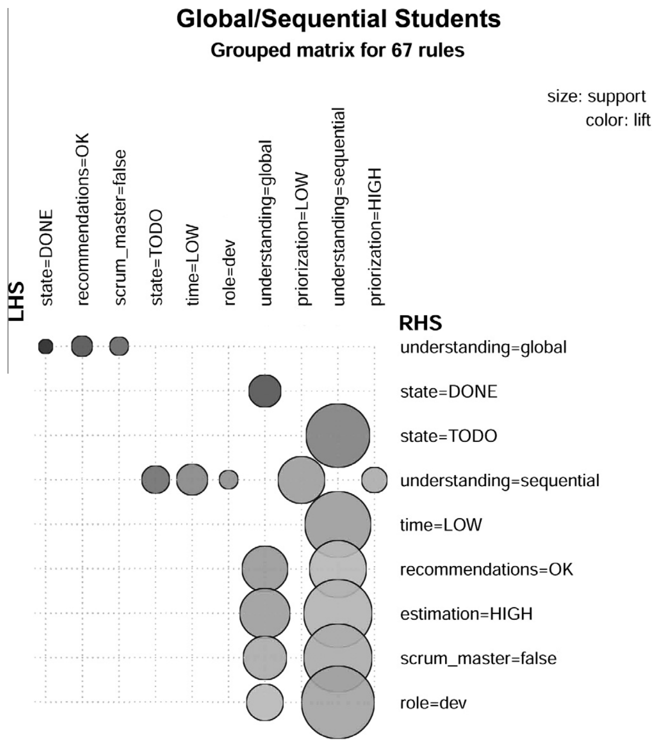

Fig. 4. Grouped matrix for association rules related to global/sequential students.

low; *understanding* = *sequential* ⇒ *estimation* = *high*). We attributed this result to the ease with which sequential students learn step by step. On the one hand, they tend to perform tasks rapidly since this kind of students clearly understands the information provided even though it is incomplete or partially structured. On the other hand, sequential students tend to accept recommendations provided within *Virtual Scrum* (*understanding* = *sequential* ⇒ *recommendations* = *ok*) and play a developer role (*understanding* = *sequential* ⇒ *role* = *dev*).

As for global students, they tend to move their assigned tasks to DONE status onto the *taskboard* (*understanding* = *global tarrow state* = *done*). Moreover, these students tend to assign high values of estimates (*understanding* = *global* ⇒ *estimation* = *high*) and play a developer role (*understanding* = *global* ⇒ *role* = *dev*) associated with implementation tasks (*category* = *implementation* ⇒ *understanding* = *global*). In line with these observations, FSLSM states that global students perform a thorough comprehension of the given information before starting to solve a problem. Therefore, these students estimate that their tasks will take considerable time since they have to analyze and understand the problem completely. Additionally, global students tend to accept less recommendations than sequential students (*understanding* = *global* ⇒ *recommendations* = *ok*; *understanding* = *sequential* ⇒ *recommendations* = *ok*).

4.4. Lesson learned

Our work is a step towards discovering the relationship between the behavior of students when they perform Scrum for the first time and their learning style according to FSLSM. In our experiment, we observed that sensing students spent more time on finishing User Stories than intuitive ones; sensing students mostly prioritized their User Stories as high, and they set higher estimates to complete them than intuitive ones. On the other hand, intuitive students tended to play the role of software developers, discarding Scrum Master responsibilities. Sensing and intuitive students considered textual recommendations given by *Virtual Scrum* as important. Likewise, both kinds of students estimated User Stories with high values so there was not enough evidence to relate this behavior to the sensing/intuitive dimension of FSLSM. We attribute these results to the fact that students did not have much experience in software development. From a professor standpoint, these observations allow for the understanding of how students with different learning styles use Scrum. Gathering this kind of information is crucial to research on how to enhance the teaching of Scrum practices when students perform them for their first time.

Active students updated their tasks status to DONE onto the *taskboard* more frequently than reflexive students, whose tasks remained in TODO status. Further, active students assigned low prioritization to their tasks and they also finished them in short periods of time. On the contrary, there is no enough evidence to state that reflexive students prioritize or finish their tasks in a particular way. Moreover, active and reflexive students accepted the recommendations given by *Virtual Scrum*, played the role of software developers and discarded Scrum Master responsibilities. However, for all the cases, the evidence showed that active students performed these behaviors more frequently than reflexive ones.

As for global students, their tasks remained in TODO status in contrast to sequential students, who usually updated their tasks status to DONE onto the *taskboard*. Sequential students spent short periods of time in their tasks, while there is not enough evidence to conclude about how long global students spent on their tasks. Furthermore, both kinds of students accepted recommendations, assigned high values of estimates and usually played the role of software developers. In short, sequential students exhibited these behaviors more frequently in comparison with global ones.

To support our aforementioned ideas, we carried out a descriptive statistics analysis of association rule measures by means of *box-plot* charts. Fig. 5 shows the *box-plot chart* that summarizes *confidence*, *support* and *lift*. *Box-plots* related to the same measure of interest were grouped, and those ones related to same FSLSM dimension were colored with the same color. For instance, perception *box-plots* are in dark gray, processing ones are in middle gray and understanding ones are in light gray. The central box shows the data between the upper and lower quartiles, with the median represented by a horizontal line. Furthermore, each plot is crossed by a vertical line that determines the data dispersion between the maximum and minimum values. The points that are misplaced far away from the main part of the *box-plot* are known as *outliers*. As shown in Fig. 5, the group of box-plots related to *support* has a mean and standard deviation values near to 0.2 and 1.4 respectively. As expected, the *support* is low and indicates that rules were extracted from an heterogeneous data set. Similarly, the low standard deviation indicates that rules are slightly scattered. By comparing the *support* values, we can state that the perception dimension reached the highest value of *support*, while the processing dimension reached the minimum *support* as expected. The *confidence* has a mean and standard deviation values near to 0.8 and 0.1 respectively. In the same way as *support*, the perception dimension reached the highest value of *confidence*. Although the processing dimension had a low *confidence* value, in comparison with the other dimensions, the understanding dimension reached the minimum *confidence* as expected. The *lift* values had a mean and standard deviation values of 1.3 and 0.3 respectively. The *lift box-plot* of the perception dimension obtained the largest number of *outliers* due to the high value of their standard deviations. The high value of *lift*, on average greater than 1, shows that the items of the rules are not independent, having a high probability of being mined on new data sets. In this context, the understanding dimension reached the highest value of *lift*, even though the *lift* value of

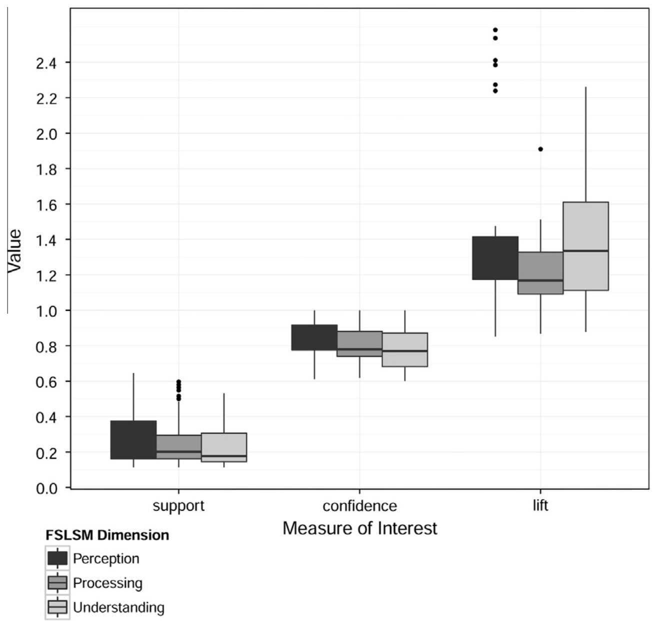

Fig. 5. *Box-plot chart* that considers metrics of interest about the association rules mined.

perception was high too. Finally, the processing dimension reached the minimum *lift* as expected.

4.5. Threats to validity

The present research explores FSLSM dimensions in isolation and how they are related to the Scrum practices, yielding promising results which are a significant contribution towards understanding students' behaviors when performing agile practices. Nonetheless, there are some aspects of our approach that may bias the results of the experiments. Firstly, each student has a learning style that is described by the four FSLSM dimensions simultaneously (Felder & Silverman, 1988; Keefe, 1988). As a consequence, further studies are required to analyze not only how the dimensions are related to the performance of Scrum practices, but also how the four dimensions interact simultaneously affecting decision making. Then, new relationships may arise from analyzing the combination of the four FSLSM dimensions and could help teachers to be aware of students' behaviors related to their learning style. Secondly, we utilized Scrum to develop a capstone project and defined a set of variables that represent good software engineering practices (Kulpa & Johnson, 2008). Thus, as these practices are general, our approach could be applied to analyze students' behaviors when they are learning not only Scrum, but also any other Software Engineering methodology. For instance, estimating the effort to develop a software task is a typical software engineering practice; moreover, consensus-based estimation techniques have been widely used in different software engineering contexts. Particularly, we have chosen Planning Poker because it is a technique commonly used in Scrum. Along this line, monitoring the tasks progress is a crucial practice in project monitoring and control; in a Scrum context, this practice is carried out by labeling software tasks as "To Do", "Doing" or "Done". As regards team organization, in any software development context, software development teams have a member that plays the leader role and is responsible for guiding, monitoring and facilitating resources for the team; in Scrum, this role is called Scrum Master. As a consequence, all the variables utilized in our research are linked to a Scrum context. Thirdly, although we have used *Virtual Scrum* as a tool for project management within Scrum teams, we believe that our approach can be applicable to other case-studies, in which other tools (JIRA[2] or Teamwork[3]) and other software development methodologies (Rational Unified Process or Extreme Programming) can be orchestrated, as long as students who attend the course receive training in the proposed combination of the methodology and tools. If methodology and tools were changed data associated with the variables researched should allow comparable analysis. Finally, it is worth noting that our approach is sensitive to the characteristics of the academic context such as the students' motivation, the Product Owner's pressure, and the contents of the course.

5. Conclusion

In this work, we have presented an approach that uses association rules to discover relationships between how students use Scrum and their learning style. Scrum was used throughout the development of a capstone project in the context of a Software Engineering course at UNICEN, where a professional environment was simulated. A virtual world equipped with Scrum artifacts, *Virtual Scrum*, supported the Scrum process during the capstone project. The tool made it possible to record all students' activities, such as the time spent on User Stories, the way in that they set priorities, the estimates of their User Stories, and the role they played in the project. The analysis of the association rules corroborated our hypothesis about the existence of relationships between the way in which students perform Scrum practices and their learning' style. Thus, our work showed fruitful results that could be useful to understand the students' differences when learning as the first step to enhance and personalize the teaching of agile methodologies in Software Engineering courses.

To bring this paper to a closure, as future work, we are planning to carry out more experiments and test with other project management tools and teaching contexts, aiming to build a framework to analyze the impact of development tools on students' behavior. This analysis would allow us to obtain information about the personality of students in order to make optimal working teams. In the same line, we will attempt to analyze the performance of professional teams with much more experience in software development as well as other software development tools so as to confirm whether our hypothesis remains successfully supported in other contexts.

Acknowledgments

We wish to thank the reviewers for providing valuable feedback to improve our work.

References

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th international conference on very large data bases. VLDB '94* (pp. 487–499). San Francisco, CA, USA: Morgan Kaufman Publishers Inc. ISBN 1-55860-153-8.
Ambler, S. (2006). Survey says: Agile works in practice. *Dr. Dobb\'s Journal, 31*(9), 62–64.
Antunes, C. (2010). *Anticipating student's failure as soon as possible*. New York, NY: Chapman & Hall/CRC Press.
Azizyan, G., Magarian, M., & Kajko-Matsson, M. (2011). Survey of agile tool usage and needs. In *Agile conference (AGILE), 2011* (pp. 29–38).
Barros, B., & Verdejo, M. (2000). Analysing student interaction processes in order to improve collaboration. The DEGREE approach. In *International journal of artificial intelligence in education*.
Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., et al. (2001). *Manifesto for agile software development*. <http://www.agilemanifesto.org/>.

[2] https://www.atlassian.com/software/jira.
[3] http://www.twproject.com/home.page.

Carver Jr, C. A., Jr., Howard, R. A., & Lane, W. D. (1999). Enhancing student learning through hypermedia courseware and incorporation of student learning styles. *IEEE Transactions on Education, 42*(1), 33–38.

Chookittikul, W., Kourik, J. L., & Maher, P. E. (2011). Reducing the gap between academia and industry: The case for agile methods in Thailand. In *Proceedings of the 2011 eighth international conference on information technology: New generations. ITNG '11* (pp. 239–244). Washington, DC, USA: IEEE Computer Society. ISBN 978-0-7695-4367-3.

Cohn, M. (2005). *Agile estimating and planning*. Upper Saddle River, NJ, USA: Prentice Hall PTR. ISBN 0131479415.

Devedzic, V., & Milenkovic, S. (2011). Teaching agile software development: A case study. *IEEE Transactions on Education, 54*(2), 273–278. http://dx.doi.org/10.1109/TE.2010.2052104. ISSN 0018-9359.

Dick, W., Carey, L., & Carey, J. (2005). *The systematic design of instruction*. Pearson/Allyn & Bacon. ISBN 9780205412747.

Essalmi, F., Ayed, L. J. B., Jemni, M., Graf, S., et al. (2010). A fully personalization strategy of E-learning scenarios. *Computers in Human Behavior, 26*(4), 581–591.

Fayyad, U. M., Piatetsky-Shapiro, G., & Smyth, P. (1996). From data mining to knowledge discovery: An overview. In Fayyad, U. M., Piatetsky-Shapiro, G., Smyth, P., & Uthurusamy, R. (Eds.), *Advances in knowledge discovery and data mining, chap. From data mining to knowledge discovery: An overview* (pp. 1–34). Menlo Park, CA, USA: American Association for Artificial Intelligence. ISBN 0-262-56097-6.

Felder, R. M., & Silverman, L. K. (1988). Learning and teaching styles in engineering education. *Engineering Education, 78*(7), 674–681.

Felder, R., & Spurlin, J. (2005). Applications, reliability, and validity of the index of learning styles. *International Journal of Engineering Education, 21*(1), 103–112.

Graf, S., & Viola, S. (2009). Automatic student modelling for detecting learning style preferences in learning management systems. In *Proc. international conference on cognition and exploratory learning in digital age* (pp. 172–179).

Graf, S., & Liu, T.-C. (2010). Analysis of learners' navigational behaviour and their learning styles in an online course. *Journal of Computer Assisted Learning, 26*(2), 116–131. ISSN 02664909.

Graf, S., Liu, T.-C., Chen, N.-S., Yang, S. J., et al. (2009). Learning styles and cognitive traits–Their relationship and its benefits in web-based educational systems. *Computers in Human Behavior, 25*(6), 1280–1289.

Hahsler, M., & Chelluboina, S. (2011). Visualizing association rules in hierarchical groups. In *Computing science and statistics, 42nd symposium on the interface: Statistical, machine learning, and visualization algorithms (Interface 2011)* (Vol. 42). The Interface Foundation of North America.

Hartmann, D., & Dymond, R. (2006). Appropriate agile measurement: Using metrics and diagnostics to deliver business value. In *Agile conference* (Vol. 6, pp. 134). http://dx.doi.org/10.1109/AGILE.2006.17, 2006.

Hawk, T. F., & Shah, A. J. (2007). Using learning style instruments to enhance student learning. *Decision Sciences Journal of Innovative Education, 5*(1), 1–19. ISSN 1540-4609.

Hossain, E., Babar, M., & young Paik, H. (2009). Using scrum in global software development: A systematic literature review. In *Fourth IEEE international conference on global software engineering, 2009. ICGSE 2009* (pp. 175–184).

Huang, E. Y., Lin, S. W., & Huang, T. K. (2012). What type of learning style leads to online participation in the mixed-mode e-learning environment? A study of software usage instruction. *Computers & Education, 58*(1), 338–349. <http://www.sciencedirect.com/science/article/pii/S0360131511001813>. http://dx.doi.org/10.1016/j.compedu.2011.08.003. ISSN 0360-1315.

Judy Kay, Irena Koprinska, & Kalina Yacef. (2010). *Handbook of educational data mining, Chap. educational data mining to support group work in software development projects*. (pp. 173–186). Taylor and Francis.

Keefe, J. W. (1988). *Profiling & utilizing learning style*. ERIC.

Kuljis, J., & Liu, F. (2005). A comparison of learning style theories on the suitability for elearning. In *Proceedings of the IASTED conference on web technologies, applications, and services* (pp. 191–197).

Kulpa, M. K., & Johnson, K. A. (2008). *Interpreting the CMMI: A process improvement approach*. Auerbach Pub.

Layman, L., Cornwell, T., & Williams, L. (2006). Personality types, learning styles, and an agile approach to software engineering education. In *Proceedings of the 37th SIGCSE technical symposium on computer science education – SIGCSE '06 38* (pp. 428–432). http://dx.doi.org/10.1145/1121341.1121474.

Limongelli, C., Sciarrone, F., & Vaste, G. (2008). LS-plan: An effective combination of dynamic courseware generation and learning styles in web-based education. In *Proceedings of the 5th international conference on adaptive hypermedia and adaptive web-based systems. AH '08* (pp. 133–142). Berlin, Heidelberg: Springer-Verlag. ISBN 978-3-540-70984-8.

MacQueen, J. B. (1967). Some methods for classification and analysis of MultiVariate observations. In L. M. L. Cam & J. Neyman (Eds.). *Proc. of the fifth Berkeley symposium on mathematical statistics and probability* (Vol. 1, pp. 281–297). University of California Press.

Mahnic, V. (2010). Teaching scrum through team-project work: Students' perceptions and teacher's observations. *International Journal of Engineering Education, 26*(1), 96–110.

Mahnic, V. (2012). A capstone course on agile software development using scrum. *IEEE Transactions on Education, 55*(1), 99–106. ISSN 0018-9359.

Mahnic, V., & Rozanc, I. (2012). Students' perceptions of Scrum practices. In *MIPRO, 2012 proceedings of the 35th international convention* (pp. 1178–1183).

McCombs, B. L., & Whisler, J. S. (1997). *The learner-centered classroom and school: Strategies for increasing student motivation and achievement. The Jossey–Bass education series*. Jossey–Bass Publishers.

Melnik, G., & Maurer, F. (2003). Introducing agile methods in learning environments: Lessons learned. In V. Malyshkin (Ed.), *Parallel computing technologies. Lecture notes in computer science* (Vol. 2753, pp. 172–184). Berlin–Heidelberg: Springer. ISBN 978-3-540-40673-0.

Ocepek, U., Bosnić, Z., Nančovska Šerbec, I., & Rugelj, J. (2013). Exploring the relation between learning style models and preferred multimedia types. *Computers & Education, 69*(0), 343–355. <http://www.sciencedirect.com/science/article/pii/S0360131513001966>. http://dx.doi.org/10.1016/j.compedu.2013.07.029. ISSN 0360-1315.

Paasivaara, M., Durasiewicz, S., & Lassenius, C. (2009). Using scrum in distributed agile development: A multiple case study. In *Fourth IEEE international conference on global software engineering, 2009. ICGSE 2009* (pp. 195–204). http://dx.doi.org/10.1109/ICGSE.2009.27, 2009.

Popescu, E. (2009). Evaluating the impact of adaptation to learning styles in a web-based educational system. In *Proceedings of the 8th international conference on advances in web based learning. ICWL '009* (pp. 343–352). Berlin, Heidelberg: Springer-Verlag. ISBN 978-3-642-03425-1.

Prata, D. N., d Baker, R. S., Costa, E. D. B., Rosé, C. P., Cui, Y., & de Carvalho, A. M. (2009). Detecting and understanding the impact of cognitive and interpersonal conflict in computer supported collaborative learning environments. In: T. Barnes, M. Desmarais, C. R., S. Ventura, E. (Eds.), *Educational data mining* (pp. 131–140). Cordoba, Spain.

Reichlmayr, T. (2003). The agile approach in an undergraduate software engineering course project. In *Frontiers in education, 2003. FIE 2003 33rd annual* (Vol. 3). ISSN 0190-5848, S2C – 13–18.

Rico, D., & Sayani, H. (2009). Use of agile methods in software engineering education. In *Agile conference, 2009. AGILE '09* (pp. 174–179). http://dx.doi.org/10.1109/AGILE.2009.13.

Rodriguez, G., Soria, A., & Campo, M. (2013). Virtual scrum: A teaching aid to introduce undergraduate software engineering students to scrum. *Computer Applications in Engineering Education*.

Salo, O., & Abrahamsson, P. (2008). Agile methods in European embedded software development organisations: A survey on the actual use and usefulness of extreme programming and scrum. *IET Software, 2*(1), 58–64. ISSN 1751-880.

Saracho, O. N. (1997). The relationship between matching teachers' and students' cognitive styles and the students' academic achievement. *Early Child Development and Care, 137*(1), 21–29. http://dx.doi.org/10.1080/0300443971370102. <http://www.tandfonline.com/doi/abs/10.1080/0300443971370102>.

Schroeder, A., Klarl, A., Mayer, P., & Kroiss, C. (2012). Teaching agile software development through lab courses. In *Global engineering education conference (EDUCON), 2012* (pp. 1–10). IEEE. http://dx.doi.org/10.1109/EDUCON.2012.6201194, ISSN 2165-9559.

Schwaber, K., & Beedle, M. (2002). *Agile software development with scrum* (Vol. 1). Upper Saddle River: Prentice Hall.

Slack, N., & Norwich, B. (2007). Evaluating the reliability and validity of a learning styles inventory: A classroom-based study. *Educational Research, 49*(1), 51–63.

Soria, A., Campo, M., & Rodriguez, G. (2012). Improving software engineering teaching by introducing agile management In *Proceedings of ASSE 2012 Argentine symposium on software engineering*.

Talavera, L., & Gaudioso, E. (2004). Mining student data to characterize similar behavior groups in unstructured collaboration spaces. In *Workshop on artificial intelligence in CSCL. 16th European conference on artificial intelligence, (ECAI 2004)* (pp. 17–23).

Zaina, L., Bressan, G., Rodrigues, J., & Cardieri, M. (2011). Learning profile identification based on the analysis of the user 's context of interaction. *Latin America Transactions, IEEE (Revista IEEE America Latina), 9*(5), 845–850. http://dx.doi.org/10.1109/TLA.2011.6030999. ISSN 1548-0992.

Zualkernan, I. A., Al Darmaki, H., & Shouman, M. (2008). A methodology for building simulation-based e-learning environments for Scrum. In *International conference on Innovations in information technology, 2008. IIT 2008* (pp. 357–360). IEEE.