Virtual Scrum: A Teaching Aid to Introduce Undergraduate Software Engineering Students to Scrum

GUILLERMO RODRIGUEZ, ÁLVARO SORIA, MARCELO CAMPO

ISISTAN—CONICET *Research Institute*, UNCPBA, *Paraje Arroyo Seco* S/N, *Tandil*, *Buenos Aires* B7001BBO, *Argentina*

Received 22 October 2012; accepted 20 October 2013

ABSTRACT: Teaching agile practices is becoming a priority in Software engineering curricula as a result of the increasing use of agile methods (AMs) such as Scrum in the software industry. Limitations in time, scope, and facilities within academic contexts hinder students' hands-on experience in the use of professional AMs. To enhance students' exposure to Scrum, we have developed *Virtual Scrum*, an educational virtual world that simulates a Scrum-based team room through virtual elements such as blackboards, a Web browser, document viewers, charts, and a calendar. A preliminary version of *Virtual Scrum* was tested with a group of 45 students running a capstone project with and without *Virtual Scrum* support. Students' feedback showed that *Virtual Scrum* is a viable and effective tool to implement the different elements in a Scrum team room and to perform activities throughout the Scrum process. Comput Appl Eng Educ; View this article online at wileyonlinelibrary.com/journal/cae; DOI 10.1002/cae.21588

Keywords: software engineering education; agile methodologies; scrum; virtual worlds

INTRODUCTION

Agile methods (AM) are becoming main-stream in the software industry. Annual Surveys[1] have shown that the use of Scrum has increased in the last few years. In 2007, 37% of respondents used Scrum, and nowadays, more than 50% of the surveyed companies are adopting Scrum as an effective vehicle toward agility [1]. The knowledge of Scrum is crucial to most companies because it emphasizes the importance of team effort and social activities in the development of software. Scrum concentrates on project management practices and includes monitoring and feedback activities that ensure transparency [2,3]. This emerging use of Scrum has opened a gap between the skills taught in academic contexts and the ones required by the software industry [4].

Several approaches have focused on bridging this gap by teaching Scrum in software engineering courses through capstone projects. A capstone project is a cooperative assignment that aims to provide students with an opportunity to integrate the concepts learned previously, deepen their understanding of those concepts, extend their area of knowledge; students are expected to apply their knowledge and skills in a quasi-real world experience [5]. Alfonso and Botia [6] compared the results of teaching Scrum practices in an undergraduate software course with a previous experience using a waterfall-like rigid process, resulting in a decrease in risks and process overhead. Tan et al. [7] reported on the use of capstone projects to initiate future software professionals in the importance of agility, flexibility, and adaptability in professional contexts. By carrying out a project-work, the students followed an iterative and incremental teaching approach based on XP, Scrum and FDD. Mahnic [2] taught Scrum through a capstone project and described the course details, students' perceptions and teachers' observations after the course. However, these approaches fail to address the teaching constraints in a university course such as large classes, multiple groups working at a time, limited space, and limited number of tutors. Using a room for multiple teams may jeopardize the effective implementation of the Scrum process since each team may require customized configurations of the room, whereas

Correspondence to: A. Soria (asoria@exa.unicen.edu.ar).

[1] http://www.versionone.com/pdf/2010_State_of_Agile_Development_Survey_Results.pdf.

physical space and teaching materials for a personalized class may prove unviable.

Each team of students in a course, to effectively learn the Scrum process, requires to set up an individual Scrum-like team room, with different visual management strategies [8,9]: planning cards for estimating user stories, whiteboards for organizing user stories in Product and Sprint Backlog, burn-down charts for accessing team performance and room for performing Daily Meetings.

To this end, we propose to exploit the richness of the 3D metaphor of virtual worlds to support the Scrum artifacts. A virtual world is a computer-simulated persistent environment similar to the real world in which users, represented by avatars, interact each other [10]. This kind of 3D environments may provide a physical topology of the organization of a team by accessing the artifacts generated during the implementation of Scrum regardless the synchrony of place.

In this work, we present *Virtual Scrum*, a prototype tool that aims to help each team of Scrum students set up a virtual working environment in which the visual metaphors required by Scrum are effectively supported. The students have information readily available through whiteboards that display the Product and Sprint Backlogs, and trace requirements easily through task boards that show team progress and performance metrics into the virtual world.

To evaluate the performance of *Virtual Scrum*, we carried out an experiment involving 45 undergraduate students from the UNICEN University. The students had to develop a capstone project by following Scrum in a two-phase homework. During the first phase, the students utilized currently available tools such as Excel, Jira, Microsoft Project, among others[2] to develop a defined set of user requirements, whereas during the second phase the same students utilized *Virtual Scrum* to develop another set of similar user requirements. At the end of the phases, we asked the students to complete a survey to collect their opinions. The data collected showed that the students agreed that the tool is useful to enhance the comprehension of Scrum (67%), to perform planning meetings (80%), and to check the progress status of a project (60%). Finally, students provided us with ideas and suggestions in order to improve our approach.

The rest of the article is organized as follows. Next section describes the main characteristics of *Virtual Scrum*. The Experimental Results section shows the results of experiments carried out to test the tool with students. Related Work section reports the related work. Finally, the last section outlines the conclusions of the article as well as some lines for future work.

VIRTUAL SCRUM

This research work explores the hypothesis that a virtual world is an adequate solution to the problem of teaching the Scrum framework, through a capstone project, within the university premises when lack of physical space and resources prevent the creation of Scrum team rooms. In this virtual environment, students use avatars to communicate through gesture, sound, icons, text, among others; and manipulate elements in the modeled room of Scrum. The simulated team room, equipped with the Scrum artifacts and rules, is crucial to engage users in a hands-on experience going through a 3D experience, participate in collaborative work, and improve their comprehension of this AM.

To test our hypothesis, we have developed *Virtual Scrum*, which follows an iterative and incremental life cycle based on Scrum as defined by Schwaber and Beedle [9]. Scrum defines and structures the software development in fixed length iterations called Sprints. On each day of the Sprint, project and product management take part in Scrum meetings. Each meeting allows communication among the Scrum members in order to identify and plan corrective actions to possible impediments in the development process. *Virtual Scrum* allows users, i.e., avatars, to play each of the Scrum roles: Product Owner, Scrum Master, and Scrum Team [9]. The Product Owner owns the user stories and leads teams to clarify the requirement specifications. The Scrum Master leads the team into applying the Scrum practices (Daily Meeting, Planning session, Retrospective Meeting, etc.), helps make decisions or acquire resources as needed, and ensures that the Scrum process is followed as intended. The Scrum Team is responsible for developing and testing the requirements. All the avatars are notified of the meeting, since they can be programmed in the *Calendar View*, which indicates different kinds of meetings that were programmed, namely Sprint Planning, Sprint Review, and Sprint Retrospective by means of green-colored cells.

Virtual Scrum is a tool to enrich the learning experience of Scrum within the framework of the capstone project in a Scrum course. In fact, we envision *Virtual Scrum* to enhance an agile teaching model like the one proposed in [11], in which the Scrum roles are assigned as follows. A professor plays the Product Owner role while students play the roles of Scrum Master and Scrum Team. The role of Scrum Master was assigned to a student with the most advance knowledge of AMs so as to preserve the self-organizing principle of this AM. This way, we expected to contribute to the student's experiential learning about management and leadership [12].

To understand the use of *Virtual Scrum* during the course, we describe a step-by-step example. For instance, the objective of the course is that students develop a virtual world named *Universidad3D*.[3] *Universidad3D* allows users to navigate the facilities of the UNICEN, play thematic games, attend to virtual courses, and communicates each other by chat, e-mail, and forum mechanisms.

Organizing and Preparing User Stories

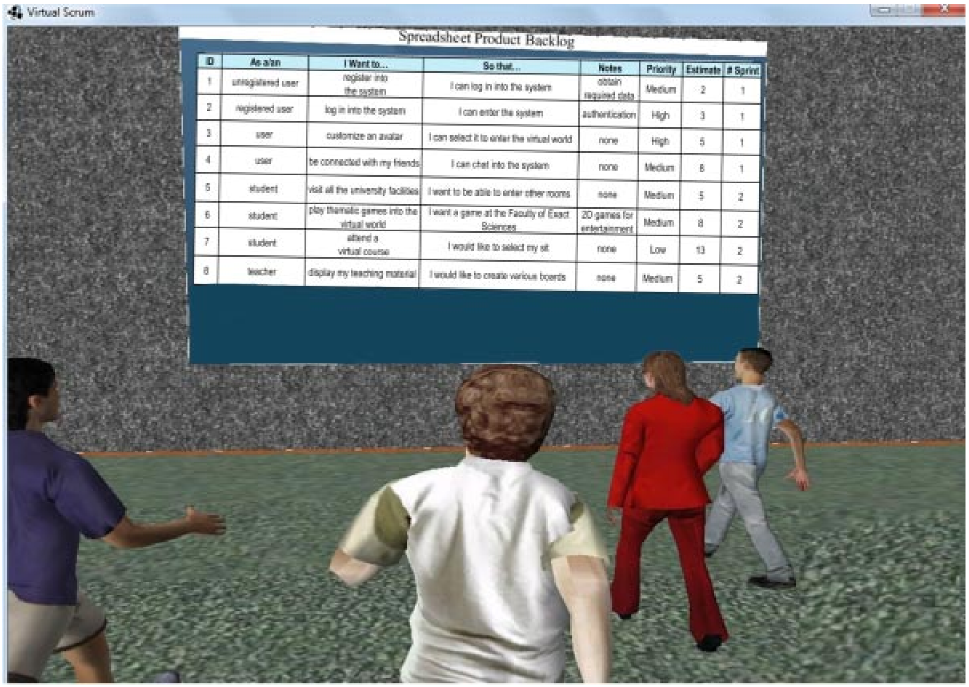
The first step in the development of *Universidad3D* is the load and prioritization of the user stories into the Product Backlog, which is the master list of the desired product features. Figure 1 shows an example of the Product Backlog loaded with the user stories for *Universidad3D*.

Virtual Scrum supports the Product Backlog as a spreadsheet that contains the backlog items. Each row of the spreadsheet represents a user story (US). A US describes a desired functionality involving role ("As…"), product features ("…I want to…") and the benefit provided to the user ("…so that…"). Row 3 represents the US "*As* a User, when I log in the virtual world *I want to* customize my avatar *so that* I can select it and dress it to enter the virtual world." Also, each row contains the identification of the US (column "ID") and some extra comments about it (column "Notes").

We use a spreadsheet because it creates a bit of history and makes the work transparent; additionally, this document can be

[2] http://www.versionone.com/pdf/2012_State_of_Agile_Development_Survey_Results.pdf.

[3] http://isistan.exa.unicen.edu.ar/u3d/.



Spreadsheet Product Backlog

ID	As a/an	I Want to...	So that...	Notes	Priority	Estimate	# Sprint
1	unregistered user	register into the system	I can log in into the system	obtain required data	Medium	2	1
2	registered user	log in into the system	I can enter the system	authentication	High	3	1
3	user	customize an avatar	I can select it to enter the virtual world	none	High	5	1
4	user	be connected with my friends	I can chat into the system	none	Medium	8	1
5	student	visit all the university facilities	I want to be able to enter other rooms	none	Medium	5	2
6	student	play thematic games into the virtual world	I want a game at the Faculty of Exact Sciences	2D games for entertainment	Medium	8	2
7	student	attend a virtual course	I would like to select my sit	none	Low	13	2
8	teacher	display my teaching material	I would like to create various boards	none	Medium	5	2

Figure 1 Spreadsheet-based Product Backlog sample. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

easily shared [13] and comprehended by the Scrum Team and the Product Owner because of its simplicity and accessibility [6]. On the spreadsheet, the professor wearing the Product Owner hat clarifies and prioritizes (column “Priority” in Fig. 1) the user stories, and the Scrum Team asks the Product Owner for the necessary information by using questionnaires, interviews, and prototypes.

After that, the Scrum Team is ready to move on to the next step, in which the user stories are estimated (column “Estimate” in Fig. 1) in order to select the ones that will be developed during each Sprint (column “# Sprint” in Fig. 1).

Planning the Sprint Backlog

The Sprint initiates with the specification of achievable and expected outcomes of that Sprint. Thus, the Scrum Team holds a planning meeting to estimate a set of user stories and reach a decision to start the activities of the Sprint. Considering the recommendations from the literature review [8,14], *Virtual Scrum* provides the *Planning Poker View* component to perform the Planning Poker technique [15,16]. Figure 2 shows the Planning Poker estimation of US 2 conducted by the Scrum Team. The team members voted the candidate estimates for US 2 using cards. Each card contains the story points proposed as the US estimation. Here, a story point represents a working day consisting of 6 h of uninterrupted work. As story point values, we decided to utilize the following Fibonacci sequence 0.5, 1, 2, 3, 5, 8, 13, and 20.

For each US, team members select their cards simultaneously by using the *Planning Poker View*. Members with high and low story points have to justify their estimates and then all the team members vote again. The estimation process for each US is repeated until a consensus is reached or the Scrum Master decides to finish the process after three iterations and then, the average of story points is calculated. The resulting estimate for the US 2 was a value of 3 story points, which meant 18 h of uninterrupted work for this US.

At the end of the planning meeting, the team selects the user stories that will be developed by defining the Sprint Backlog and disaggregates the user stories into smaller tasks. Then, the team starts the development of the user stories in the Sprint backlog, which cannot be changed along the Sprint.

Controlling and Monitoring Sprint Work

Every day of the Sprint at the same time, the team meets in a Daily Meeting to review and discuss the task progress. During a 15-minute’s time-boxing period, each team member answers three questions through the *Daily Meeting View* artifact: What have you done? Do you have any problems? What are you doing? Here, a team member answered the question “What have you done?” by expressing that he has been implementing a scripting prototype. The team members can support his/her reply by surfing the Web and presenting documents or reports. Notice that these views are also helpful to integrate the team room with external traditional development tools.

After answering the three questions, the Scrum Team updates the figures of the tasks in the *Task Board* (Fig. 3). Here, the tasks from US 2 (Fig. 1) are displayed on the *Task Board*. As the development of the tasks evolves, the tasks move along the DOING list on the *Task Board* and team members are able to load hours worked into the assigned tasks. When the team member completed a task, moves it to the DONE list. Then, the team member selects a new task from the *Task Board*, usually chosen when answering “What are you doing?”

Notice that the *Task Board* is a very powerful tool that creates communication among team members easily, since they can glance at it and review the work in progress, completed, or pending. The *Task Board* both helps team members visualize tasks and shows

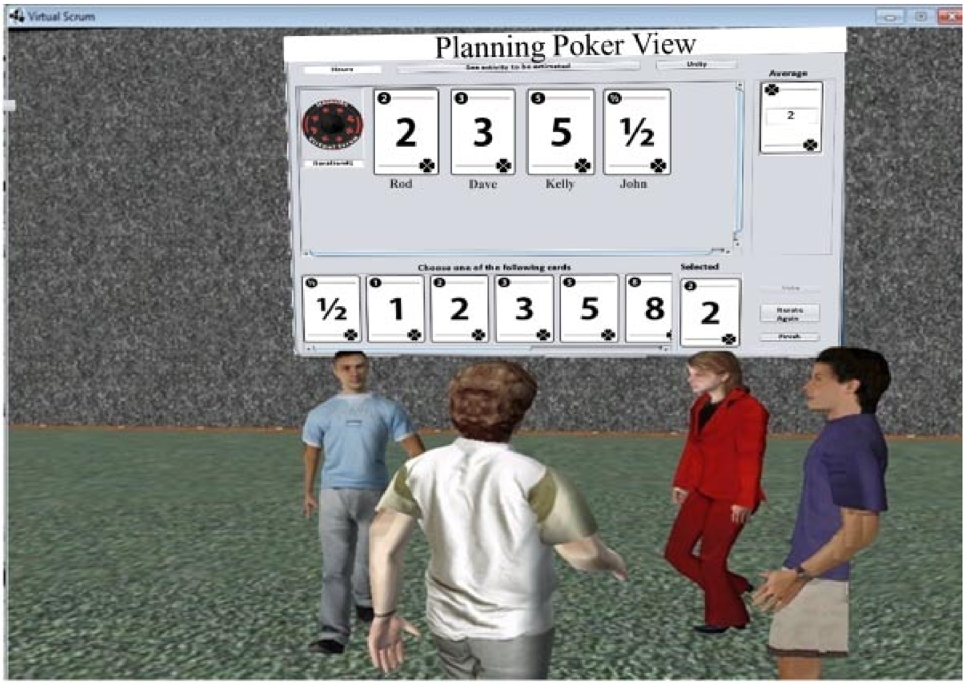
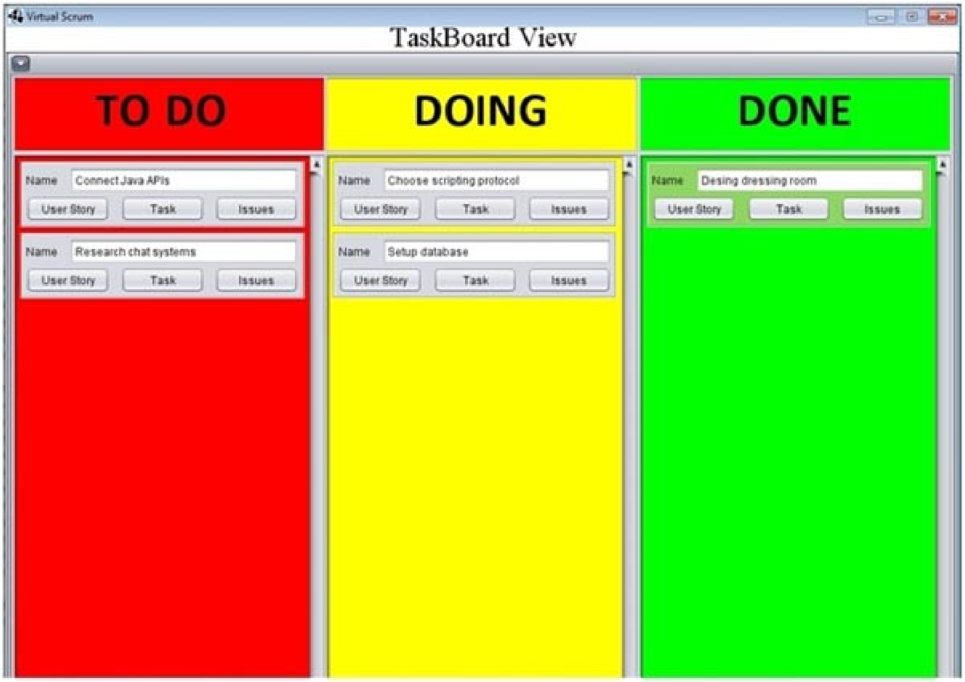


Figure 2 Estimation of User Story 2 by using the *Planning Poker View*. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

what they have accomplished, encouraging them to move on to new tasks. The *Task Board* is a useful aid to embrace the agile values such as transparency, collaboration, focus, and self-organization, among others. The Daily Meeting is ended by the Scrum Master when the timer reaches the 15-min period.

Closing the Sprint

The Sprint concludes with a review of the deliverables to be evaluated by the Product Owner in a Sprint Review meeting. Then, a Sprint Retrospective meeting takes place to assess achievement



TO DO	DOING	DONE
Name: Connect Java APIs	Name: Choose scripting protocol	Name: Desing dressing room
Name: Research chat systems	Name: Setup database	

Figure 3 *Task Board View* containing the status of User Story 2 tasks. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

of initial goals, review risks, and carefully define the process aspects to be improved.

In the Sprint Review, the Scrum Team displays the user stories completed during the Sprint by a demo. The objective of the demo is to examine the work done and get feedback from the Product Owner and other stakeholders. During the demo, the team shows how the developed features pass the acceptance tests and the Product Owner may want improvements to the features or new features, after interacting with the real software. It is worth mentioning that in our Scrum course, we decided to perform the demo in a face-to-face meeting. The reason for this kind of meeting stems from the necessity of augmenting *Virtual Scrum* with a desktop sharing tool to present the software product increment. The simultaneous use of both tools results in a high overhead and technological cost as well as [17] complex interaction that decreases team members' sense of presence and commitment [18]. A further reason for holding face-to-face meetings during demos is that the distributed agile software development may lead to misunderstandings between team members and stakeholders [19], and mistrust between the students and the professor.

After the Sprint Review, the team performs the Sprint Retrospective. This meeting represents an opportunity to reflect on what occurred during the iteration, and identify the problems that may prevent the team from improving their productivity. To do so, each team member mentions the problems that need to be addressed. *Virtual Scrum* supports the impediment list artifact, for which team members identify solutions and members responsible for implementing them. The Scrum Master is in charge of monitoring the effective adoption of those solutions so as to reduce the list of impediments.

To augment the process analysis with performance indicators, *Virtual Scrum* supports a *Burn-down chart* (Fig. 4), which represents the estimated story points ("Ideal Effort") and the *burnt* story points ("Team Effort") during the 15-day sprint for the *Universidad3D* project. Using this chart, the team knows both when the pending user stories in the Sprint Backlog should be completed and when the team deviates from the ideal effort at a specific moment.

At end of the course, the team presents the final integrated product to the Product Owner. Here, the professor assesses the way students followed Scrum during the development of the product such as evaluations on students' cooperative work, maintenance of the Sprint Backlog, and meeting of deadlines. There is no formal final exam, and the students' grades are determined on the basis of the number of user stories accomplished in the Product Backlog, the quality of the software and documentation developed, and the fulfillment of releases and Sprint plans.

To assist the professor in the students' assessment, *Virtual Scrum* provides the *Task Board* to evaluate the progress of the user stories, *Planning Poker View* to evaluate students' estimates, *Burn-down chart* and *Daily Meeting View* to evaluate students' performance, and the *Sprint Retrospective artifacts* to evaluate the quality of products and documents completed by the students.

To sum up, we believe that *Virtual Scrum* will help students improve their learning of Scrum practices and get a concrete outlook of how to setup a suitable Scrum-based team room to develop a software project.

EXPERIMENTAL RESULTS

This section describes the experiments that were carried out to assess whether *Virtual Scrum*, which focuses on implementing a Scrum-like team room, is suitable for introducing undergraduate students to the agile development. The experiments involved 45 undergraduate students and the development of a capstone project, after which the students were asked to complete a survey to gather their opinions. It is worth clarifying that for the sake of impartiality,

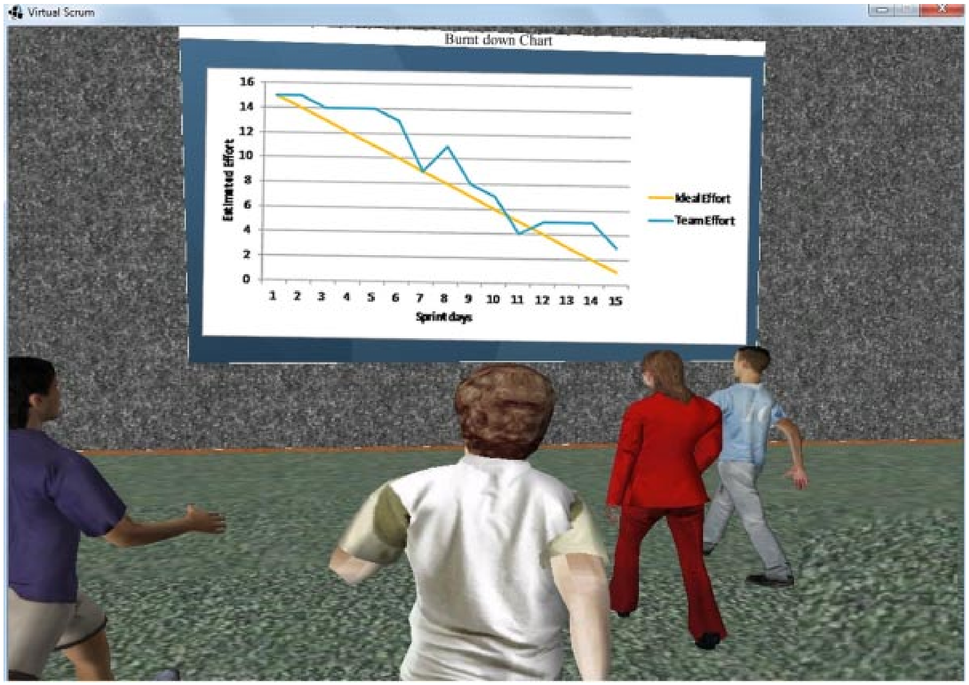


Figure 4 Showing team performance by displaying the *Burn-down chart*. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Table 1 Survey Statements With *Virtual Scrum*

Query item	Totally agree	Agree	Somewhat agree	Somewhat disagree	Disagree	Totally disagree
1. *Virtual Scrum* eases the communication among team members	0 (0%)	8 (17.8%)	16 (35.6%)	8 (17.8%)	10 (22.2%)	3 (6.7%)
2. *Virtual Scrum* helps students to enhance their comprehension of Scrum	13 (28.9%)	17 (37.8%)	15 (33.3%)	0 (0%)	0 (0%)	0 (0%)
3. *Virtual Scrum* aids students to keep the traceability of the user stories	1 (2.23%)	5 (11.1%)	14 (31.1%)	8 (17.8%)	12 (26.7%)	5 (11.1%)
4. *Virtual Scrum* facilitates teams making estimates through the Planning Poker technique	4 (8.9%)	14 (31.1%)	18 (40%)	8 (17.8%)	1 (2.23%)	0 (0%)
5. *Virtual Scrum* clarifies the daily work by having the group and individual work always available	1 (2.23%)	6 (13.3%)	11 (24.45%)	11 (24.45%)	15 (33.3%)	1 (2.23%)
6. *Virtual Scrum* provides students with metrics to present the team performance through the Sprint Retrospective meeting	6 (13.3%)	20 (44.5%)	16 (35.6%)	3 (6.7%)	0 (0%)	0 (0%)
7. *Virtual Scrum* facilitates the availability and accessibility of software artifacts in each meeting	3 (6.7%)	9 (2%)	14 (31.1%)	10 (22.2%)	5 (11.1%)	4 (8.9%)
8. *Virtual Scrum* simplifies the Daily Meetings when students are geographically distributed	2 (4.45%)	9 (2%)	10 (22.2%)	8 (17.8%)	12 (26.7%)	4 (8.9%)

at no time students were informed that the second objective of the use of *Virtual Scrum* was part of an experiment.

The experiments were performed during 2011 in the context of the Software Engineering course of the System Engineering BSc program at Faculty of Exact Sciences (Department of Computer Science—UNICEN). The course requirements are appropriate design and programming skills, object-oriented programming basics, management of databases, concepts of operating systems, and some experience with Java development.

As mentioned in previous sections, students developed a set of user stories of similar complexity corresponding to *Universidad3D*. For the experiments, the students were randomly divided into six Scrum teams of seven members. The teams were responsible for developing the aforementioned eight user stories in two phases. The first phase consisted in building four user stories during a 15-day sprint following Scrum supported by a set of long-established open source and academic-licensed software tools. The second phase consisted in building the next four user stories during a 15-day sprint, but using *Virtual Scrum*. All of the teams received training in the use of these tools.

At the end of the second phase, we conducted a survey to gather the students' opinions about *Virtual Scrum*. The survey contained eight items on a six-point Likert scale [20]. The use of an even-numbered scale aimed to better gather the students' opinions to avoid processing neutral mid-points. The survey was anonymous and answered by the 45 students. The survey was focused on assessing whether *Virtual Scrum* facilitates the development of a software product following Scrum, and it was a list of statements to which the students could either totally agree, agree, somewhat agree, somewhat disagree, disagree o totally disagree. Totally disagree represents the most negative opinion (score 1) and totally agree the most positive one (score 6). Additionally, we reserved a textbox for further comments or suggestions.

Analysis of Students' Opinions About *Virtual Scrum*

Results of the survey are shown in Table 1. Firstly, we analyzed the communication among team members along the experience (query item 1). Out of 45 surveyed students, 53.33% agreed or somewhat agreed with the idea of "*Virtual Scrum* eases the communication among team members." Moreover, 87.5% out of that 53.33% expressed that "*Virtual Scrum* simplifies the Daily Meetings when students are geographically distributed." In contrast, the remaining 46.66% disagreed with our aforementioned idea; this dissatisfaction may be attributed to the preliminary-stage of *Virtual Scrum* regarding the usability of the 3D interfaces such as avatar movement and camera handling, as well as the lack of audio and video support. This group of students found the tool uncomfortable, and stated that performance could improve if the communication media facilitated more collaborative environments during meetings. To overcome this drawback, the students employed Skype[4] as a complement to the communication media.

With regard to traceability of user stories 44.4% of the students agreed or somewhat agreed to "*Virtual Scrum* aids students to keep the traceability of the user stories." On the other hand, 57.78% of the students agreed or somewhat agreed to "*Virtual Scrum* facilitates the availability and accessibility of software artifacts in each meeting." Similarly, we were also interested in assessing the performance of *Virtual Scrum* on letting both individual and group work remain permanently visible (query item 5). Table 1 shows that 40% of the students agreed or partially agreed with this idea, since the work visibility allows students to trace the progress of the team, and the Scrum Master to follow each member' advance.

Observing the results concerning query items 3, 5, and 7, we noted that the levels of agreement and disagreement of the students remained even. We attributed these results to the fact that the 3D metaphor was not appropriate for the aforementioned interaction. Thus, the solution obtained from 2D tools matched the results obtained with *Virtual Scrum*.

Highlighting the salient aspects of *Virtual Scrum* that reinforce our hypothesis, it is possible to observe that the students favored the 3D metaphor when evaluating the knowledge acquired during the interaction with the tool, the commitment gained during planning sessions and self-reflection from the display of students' performance. Interestingly, all the surveyed students agreed or somewhat disagreed to the idea that "*Virtual Scrum* helps students to enhance their comprehension of Scrum." From a conceptual

[4] http://www.skype.com.

point of view, the students stated that *Virtual Scrum* allowed them to strengthen the software engineering concepts studied in previous courses, in which problem-solving skills, necessary in professional contexts, were developed through simulations. As for the planning sessions, 80% of the students agreed or somewhat agreed to "*Virtual Scrum* facilitates teams to make estimates through the Planning Poker technique." On the contrary, the remaining 20% of the students preferred face-to-face meetings to discuss and estimate the user stories.

One interesting finding was the students were satisfied with the display of metrics and progress status in *Virtual Scrum*; 93.33% of the students agreed or somewhat agreed with the idea that "*Virtual Scrum* provides students with metrics to present the team performance through the Sprint Retrospective meeting," because this meeting encouraged observation of students' progress and reflection on the mistakes made during the sprint, which in turn allowed them to avoid those interferences in subsequent iterations. This strategy was considered effective because some uncooperative students became more proactive. Surprisingly, only 3 out 45 students thought that showing performance metrics failed to cause the desired effect on the team members.

Quantitative Analysis of the Students' Opinions

The Likert scores per query item and student were computed to quantify the opinions of all the participants. The scores per query item were used in the one-sample Student's *t*-test to evaluate how far opinions deviate from the null hypothesis, which in this experiment is that the students' perception of *Virtual Scrum* is neutral: with arithmetic mean value 3 for all query items.

Table 2 shows the results of the survey. Columns represent the arithmetic mean value (Mean), the standard deviation (Std. dev.) and the *t*-value of the one-sample Student's *t*-test.

Table 2 Analysis of Likert Scores From the Survey

	Scrum implementation with *Virtual Scrum*		
Query item	Mean	Std. dev.	One-sample *t*-test (*t*-value)
1	3.39	1.21	2.17*
2	4.91	0.83	15.46*
3	3.06	1.32	0.36
4	4.28	0.93	9.30*
5	3.17	1.17	1
6	4.65	0.79	14.09*
7	3.58	1.35	2.92*
8	3.30	1.39	1.47

*$P < 0.05$.

The *t* values show that three out of eight hypotheses were rejected. As students grades were significantly higher than 3, we can accept the alternative hypothesis that the students' perceptions of *Virtual Scrum* is positive. We observed a statistically significant difference in query items 2, 4, and 6 when using *Virtual Scrum*. That is to say, most students agreed that *Virtual Scrum* is a friendly alternative to compensate for the lack of physical space and resources within the university premises. These students conveyed that the use of *Virtual Scrum* allowed more accurate comprehension of Scrum, enhanced commitment during the planning session, and stimulated improve performance through Sprint Retrospective Meetings. These remarks ratify the highlighted results observed in the sub-section above.

Regarding students' overall perception of *Virtual Scrum*, we computed the Likert score per student considering the eight query items of the survey. Therefore, our Likert scale was in the range of 8–48, with 8 being strongly disagree with the statements about *Virtual Scrum* mentioned above (Table 1) and with 48 being strongly agree with them. Figure 5 shows the score histogram, in which each bar contains the number of students who had the same score. Moreover, the score histogram was smoothed using Bézier curves as it is shown by the blue curve in Figure 5. This depicts that the results also tended toward a normal distribution with an average $\mu = 30.37$ and a standard deviation $\sigma = 4.60$. Then, the normal distribution test and the Chevyshev theorem [21] were applied, so that we concluded that 78% of the students scored between $[\mu - \sigma, \mu + \sigma]$. Thus, 35 students scored in the range of [25.27, 34.96], which manifests a very good perception of *Virtual Scrum* from the experience.

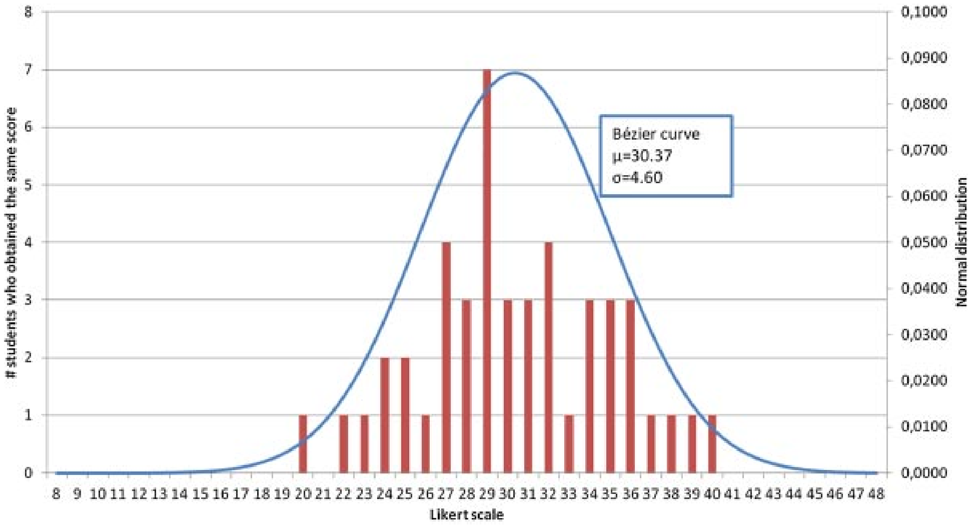


Figure 5 Likert Scale: Frequency of the scores of *Virtual Scrum*. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

RELATED WORK

In order to prepare students for their professional experiences in commercial software development, where AMs are widely used, some researchers have introduced Scrum in software engineering courses by means of a capstone project. However, teaching through a capstone project is a challenging task, due to drawbacks such as complexity of projects, lack of physical resources, numerous groups of students and packed schedules, which affect the normal running of the course.

Another increasing trend in teaching Scrum practices is the use of simulation games, which may act as a bridge from academic knowledge to the industry [22]. Along this line, we found a number of agile-based games[5] commonly used in industries and coaching, designed to strengthen and fix concepts and practices. These games require physical presence and are useful to learn how to adapt to changes in software requirements, customer management strategies and self-organizing dynamics in agile teams, among others; since they are cornerstone of Scrum. In the academic context, we found PlayScrum [23], a card game to allow university-level students to learn Scrum; just as in *Virtual Scrum*, students play different Scrum roles in a software development project in which the AMology is followed. Unlike PlayScrum, *Virtual Scrum* uses a virtual world to simulate a real work environment handling 3D displays of the Scrum artifacts. Games such as Agile Hour [24] and XP Game [25] teach XP[6] practices through a web site; these practices complement the Scrum ones, by covering programming practices, and their incorporation in *Virtual Scrum* would maximize students' programming skills. However, the use of 2D games results in unsuitable strategies to provide an immersion experience and realistic display of a Scrum environment [26,27].

In this context, virtual reality arises as an innovative technology that enhances students' motivation, understanding and creative learning [28]. Certainly, virtual worlds have great potential to make the learning experience both challenging and appealing by providing visuals that are easier to retain [23,29,30]. Moreover, the user representation through avatars allows more active participation by manipulating the 3D artifacts [31].

The use of virtual reality has been addressed in several approaches to teach in various fields of science and engineering in university courses. In [32], the authors explored a 3D learning tool to easily observe and handle internal structures of a generating unit of energy, reducing the gap between the real experience and theoretical concepts in the area of systems of power. In [33], a virtual-lab relied on a HTML Web page for control education is presented to illustrate the dynamic behavior of an industrial boiler in a user-friendly way. Diedro-3D is an application that addresses the challenges that architecture students face when using descriptive geometry [34], giving students greater autonomy to study geometry. In [35], a project to develop, build up, and evaluate solutions for virtual mobility and e-learning in biomedical engineering is presented. The authors included different educational materials to provide students with the possibility to revise course contents at any time, place, or pace. In [36], the authors studied the feasibility of introducing SimuSurvey in regular surveyor training courses to support students' learning experience. Along this line, the use of virtual reality simulations has also been utilized in [28] to teach earth science concepts, improving students' spatial abilities. In addition, a 3D virtual hydraulic model to facilitate the teaching of hydraulic engineering is presented in [37], resulting in a reduction of time and effort in comparison with experiments in a traditional laboratory.

All the above-mentioned approaches support the notion that virtual worlds and simulations cater for teachers' needs in traditional courses, such as acquisition and maintenance of equipment for each group of students, the influence of location and time of the class on the effectiveness of the lesson, and the difficulty of clearly demonstrating every step to each student in the class at a suitable pace. What is more, virtual worlds provide students with hands-on experimentation without hazardous and costly laboratories.

In the context of our research, *Virtual Scrum* uses a virtual world to harness the 3D interfaces for training students in their performance in a simulated Scrum environment. As far as software teaching with virtual worlds is concerned, some approaches have been introduced. For instance, Ye et al. [10] used a virtual world to enhance software engineering education by exploiting communication and collaboration tools to teach practices such as problem solving, plan formulation, interpretative analysis, and adaptation to rapid change, in a virtual office provided by Second Life [10,38]. In addition, in order to evaluate the effects of Second Life when playing games, the authors developed the Second Life version of two so-called games to introduce software engineering practices: *Groupthink* and *SimSE*. The former focuses on teaching software specification practices, and the latter aims to train in project management skills. Likewise Ye et al. [10], Parsons et al. used a 3D virtual world based on Open Wonderland[7] (originally Project Wonderland) to support a workshop activity based on agile software development processes. This workshop enables students to take part in this activity despite their geographical distribution [39]. The approach tackles problem solving by defining and specifying user stories, and employing a shared board, in which users may sketch prototypes of designs. Improving the aforementioned approaches, *Virtual Scrum* supports a task board for planning and tracing user stories; a daily meeting artifact for solving problems, adapting rapidly to changes and removing impediments; and a burn-down chart for reflecting on the past sprint and making continuous process improvements during retrospective meetings. Furthermore, unlike Parsons et al., we concentrate on using *Virtual Scrum* to teach Scrum assisting professors and students in the development of a capstone project.

CONCLUSIONS

In undergraduate courses, there is an increasing need for effective ways of teaching the fundamentals of AMs such as Scrum; yet constraints within educational premises demand creative use of resources. In order to tackle these limitations, this article has introduced *Virtual Scrum*, a simulated environment, in which students experienced the use of a suitable Scrum-based team room while playing the roles of software developers in a capstone project. This work showed that *Virtual Scrum* is an effective tool for teaching up-to-date software engineering procedure and training students in the use of valuable skills to work in professional contexts through a virtual world.

According to the data collected from the 45 students surveyed, we found that using *Virtual Scrum*, as a teaching aid, was helpful in improving students' comprehension of the

[5] http://www.dosideas.com/wiki/Juegos_Agiles.

[6] http://www.extremeprogramming.org/.

[7] http://openwonderland.org/.

fundamentals of agile practices and principles of developing software with Scrum. It is worth noting that the tool outperformed students' expectations with regard to the *Virtual Scrum* support for planning meetings, which increased students' commitment; and follow-up metrics, which allowed students to self-reflect on their performance in the Sprint Retrospective meetings.

The students also provided valuable feedback on user interactions and traceability of the user stories through *Virtual Scrum*. Based on this feedback, we will intend to improve user interactions by upgrading the support of media aids, specially the avatar integration with current social networks. As for traceability of user stories, we found that students preferred using 2D tools for dealing with configuration management rather than a 3D representation of the artifacts; for this reason, we will complement *Virtual Scrum* with conventional and open-source development tools.

As a further stage in this research, we are planning to incorporate an intelligent agent to offer students personalized assistance during the use of Scrum. To tackle this issue, we will equip *Virtual Scrum* with performance indicators to obtain more information about students' interactions with the tool. This information will be useful to proactively assist the student by suggesting him/her personalized courses of action that will help them during the Scrum process.

ACKNOWLEDGMENTS

We thank the students who participated in the evaluation of *Virtual Scrum* for their good predisposition in the experiments.

REFERENCES

[1] V. Mahnic, Teaching scrum through team-project work: Student's perceptions and teacher's observations, Int J Eng Educ 26 (2010), 96–110.

[2] V. Mahnic, A capstone course on agile software development using scrum, IEEE Trans Educ 55 (2011), 96–106.

[3] C. Coupal and K. Boechler, Introducing agile into a software development capstone project, Proceedings Agile Conference, 2005, 289–297.

[4] T. Dingsøyr, T. Dybå, and P. Abrahamsson, A preliminary roadmap for empirical research on agile software development, Proceedings of the Agile Conference, 2008, 83–94.

[5] S. Tilley, S. Huang, K. Wong, and S. Smith, Report from the 2nd International Workshop on Software Engineering Course Projects (SWECP 2005), Proceedings of the 19th Conference on Software Engineering Education and Training, Turtle Bay, HI, 2006, 87–94.

[6] M. I. Alfonso and A. Botia, An iterative and agile process model for teaching software engineering, Proceedings of the 18th Conference on Software Engineering Education and Training, 2005, 9–16.

[7] C. Tan, W. Tan, and H. Teo, Training students to be agile information systems developers: A pedagogical approach, Proceedings of the 2008 ACM SIGMIS CPR Conference on Computer Personnel Doctoral Consortium and Research, 2008, 88–96.

[8] M. Cohn, Agile estimating and planning, Prentice Hall, United States, 2006.

[9] K. Schwaber and M. Beedle, Agile software development with scrum, Prentice Hall, United States, 2002.

[10] E. Ye, C. Liu, and J. A. Polack-Wahl, Enhancing software engineering education using teaching aids in 3-d online virtual worlds, Proceedings of the 37th ASEE/IEEE Frontiers in Education Conference, Milwaukee, WI, 2007, T1E-8-T1E-13.

[11] Á. Soria, M. Campo, and G. Rodríguez, Improving software engineering teaching by introducing agile management, Proceedings of the 13th Argentine Symposium on Software Engineering, ISSN: 1850-2792, 2012, 215–229.

[12] S. Davies, Appointing team leads for students software development projects, J Comput Sci Coll 25 (2009), 92–99.

[13] J. Grenning, Planning poker or how to avoid analysis paralysis while release planning, Available from: http://renaissancesoftware.net/files/articles/PlanningPoker-v1.1.pdf [Accessed: September 2012], 2002.

[14] V. Mahnic and T. Hovelja, On using planning poker for estimating user stories, J Syst Softw 85 (2012), 2086–2095.

[15] E. Oliveira and R. Lima, State of the art on the use of scrum in distributed development software, Rev Sist Comput Salv 1 (2011), 106–119.

[16] M. Paasivaara, S. Durasiewicz, and C. Lassenius, Using scrum in distributed agile development: A multiple case study, Proceedings of the Fourth IEEE International Conference on Global Software Engineering, 2009, 195–204.

[17] S. V. Shrivastava and H. Date, Distributed agile software development: A review, J Comput Sci Eng 1 (2010), 10–16.

[18] R. Likert, A technique for the measurement of attitudes, Arch Psychol 22 (1932), 5–55.

[19] C. H. Brase and C. P. Brase, Understandable statistics, 8th ed., Heath and Company, Washington, DC, 2005.

[20] A. A. Deshpande and S. H. Huang, Simulation games in engineering education: A state-of-the-art review, Comput Appl Eng Educ 19 (2011), 399–410.

[21] D. Parsons and M. Cranshaw, Agile technique hour, Available from: http://www.massey.ac.nz/∼dpparson/agilehour.htm [Accessed: September 2012], 2008.

[22] B. Dalgarno and M. J. W. Lee, What are the learning affordances of 3-D virtual environments? Br J Educ Technol 41 (2010), 10–32.

[23] M. J. Jacobson, C. Miao, B. Kim, Z. Shen, and M. Chavez, Research into learning in an intelligent agent augmented multi-user virtual environment, Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, 3, 2008, 348–351.

[24] M. Wrzesien, and M. Alcañiz Raya, Learning in serious virtual worlds: Evaluation of learning effectiveness and appeal to students in the E-Junior project, Comput Educ 55 (2010), 178–187.

[25] G. Jimenez-Diaz, P. A. Gonzalez-Calero, and M. Gomez-Albarran, Role-play virtual worlds for teaching object-oriented design: the ViRPlay development experience, Softw Pract Exper 42 (2011), 235–253.

[26] J. Hedberg and S. Alexander, Virtual reality in education: defining researchable issues, Educ Media Int 31 (1994), 214–220.

[27] E. Castronova, Synthetic worlds: The business and culture of online games, University of Chicago Press, United States, 2005.

[28] V. Peeters and P. V. Cauwenberghe, The XP game, Available from: http://www.xp.be/xpgame.html [Accessed: April 2012], 2008.

[29] D. Parsons and R. Stockdale, Cloud as context: virtual world learning with Open Wonderland, Proceedings of the 9th World Conference on Mobile and Contextual Learning (mLearn 2010), Valetta, Malta, 2010, 123–130.

[30] Z. Gao, S. Liu, M. Ji, and L. Liang, Virtual hydraulic experiments in courseware: 2D virtual circuits and 3D virtual equipments, Comput Appl Eng Educ 19 (2011), 315–326.

[31] Y. Shin, Virtual reality simulations in web-based science education, Comput Appl Eng Educ 10 (2002), 18–25.

[32] M. Cohn, User stories applied, Addison-Wesley, United States, 2004.

[33] K. E. Kanyarusoke and J. Uziak, Spreadsheets: The ideal tool for distance learning in engineering education, Comput Appl Eng Educ 19 (2011), 216–223.

[34] M. P. Alves de Sousa, M. Ribeiro Filho, M. V. Alves Nunes, and A. da Costa Lopes, A 3D learning tool for a hydroelectric unit, Comput Appl Eng Educ 20 (2012), 269–279.

[35] C. Martin-Villalba, A. Urquia, and S. Dormido, Development of an industrial boiler virtual-lab for control education using Modelica, Comput Appl Eng Educ 21 (2013), 36–45.

[36] J. Martin-Gutiérrez, F. A. Gil, M. Contero, and J. L. Saorín, Dynamic three-dimensional illustrator for teaching descriptive geometry and training visualisation skills, Comput Appl Eng Educ 21 (2013), 8–25.

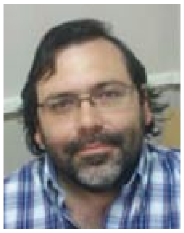
[37] A. Kybartaite, J. Nousiainen, and J. Malmivuo, Technologies and methods in virtual campus for improving learning process, Comput Appl Eng Educ 21 (2013), 185–192.

[38] H. Kuol, S. Kang, C. Lu, S. Hsieh, and Y. Lin, Using virtual instruments to teach surveying courses: Application and assessment, Comput Appl Eng Educ 19 (2011), 411–420.

[39] S. De Freitas, Serious virtual worlds. A scoping study, Prepared for the JISC e-learning programme, 2008.

BIOGRAPHIES

Guillermo Rodríguez received the Computer Engineer degree from Universidad Nacional del Centro de la Provincia de Buenos Aires (UNICEN), Tandil, Argentina, in 2010, and is currently pursuing the Ph.D degree in Computer Science at the same University. Since 2008, he has been part of ISISTAN Research Institute (CONICET - UNICEN), where has worked on projects related to software engineering, virtual reality and games for education. His research interests include software architecture materialization. Mr. Rodríguez has obtained a scholarship from CONICET to complete his doctoral studies.



Álvaro Soria received the Computer Engineer degree from Universidad Nacional del Centro de la Provincia de Buenos Aires (UNICEN), Tandil, Argentina, in 2001, and the Ph.D degree in Computer Science at the same university in 2009. Since 2001, he has been part of ISISTAN Research Institute (CONICET - UNICEN). His research interests include Software Architectures, Quality-driven Design, Object-oriented Frameworks and Fault Localization and Virtual Learning Environments and Agile Methodologies.

Marcelo Campo received the Computer Engineer degree from Universidad Nacional del Centro de la Provincia de Buenos Aires (UNICEN), Tandil, Argentina, in 1988, and the Ph.D degree in Computer Science from Instituto de Informática de la Universidad Federal de Rio Grande do Sul (UFRGS), Brazil, in 1997. He is currently an Associate Professor at Computer Science Department and Director of the ISISTAN Research Institute (CONICET - UNICEN). His research interests include intelligent aided software engineering, software architecture and frameworks, agent technology and software visualization.