Comprendiendo la Aplicabilidad de Scrum en el Aula: Herramientas y Ejemplos

Antonieta Kuz[1], Mariana Falco[1], Roxana S. Giandini[1,2]

[1] LINSI, Facultad Regional La Plata, Universidad Tecnológica Nacional, La Plata, Buenos Aires, Argentina
[2] LIFIA, Facultad de Informática, Universidad Nacional de La Plata, La Plata, Buenos Aires, Argentina

akuz@frlp.utn.edu.ar, mfalco@frlp.utn.edu.ar, rgiandini@frlp.utn.edu.ar

Recibido: 02/03/2017| Corregido: 23/03/2018| Aceptado: 30/03/2018

Resumen

Las metodologías ágiles permiten flexibilizar y gestionar el desarrollo de software siendo Scrum un marco que permite el desarrollo y mantenimiento de productos complejos, reduciendo el riesgo durante la realización de un proyecto trabajando de manera colaborativa. Debido a su capacidad para potenciar la autoorganización y cooperación, son cada vez más los profesionales que están explorando cómo aplicar Scrum en la educación. Consecuentemente, el objetivo que persigue el presente artículo es evidenciar los distintos tópicos que se consideran a la hora de trabajar con Scrum, presentando tres ejemplos de aplicación de metodologías ágiles en el aula.

Palabras clave: Metodologías ágiles; Scrum; Aula; Ingeniería de software; Ambiente educativo; eduScrum; Aprendizaje ágil.

Abstract

Agile methodologies allow to make more flexible and to manage software development being Scrum a framework that enables the development and maintenance of complex products, reducing the risk during the realization of a project working collaboratively. Because of its ability to enhance self-organization and cooperation, more and more practitioners are exploring how to apply Scrum in education. Consequently, the objective of this article is to highlight the different topics that are considered when working with Scrum, showing three examples of applying agile methodologies in the classroom.

Keywords: Agile methodologies; Scrum, Classroom; Software engineering; Educational environment; eduScrum; Agile learning.

1. Introducción

El mundo actual se encuentra en una etapa de cambio continuo, lo que conlleva a que la educación también lo esté. Desde antaño las aulas encarnan un medio donde pueden difundirse conocimientos, pero hoy en día son necesarios espacios fuera del aula donde los alumnos formulen sus propias nociones y saberes. De esta manera, la escuela inteligente es aquella que logra entretejer los conocimientos dentro y fuera del aula. En la actualidad, existen diversas escuelas que se re-plantean la forma tradicional de enseñanza, por lo que en este contexto es fundamental estudiar diversos ejemplos de escuelas en diferentes países del mundo como Estados Unidos, India, España, Finlandia, entre otros [1] con el fin explícito de conocer para poder implementar eventual y paulatinamente un proceso de cambio y mejora educativa.

Según la IEEE [2], la Ingeniería de Software es la aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento de software. Se desprende una rama denominada Ingeniería de Software Educativo que brinda apoyo al momento de realizar aplicaciones de software que implementen procesos de aprendizaje. Por añadidura, una metodología de desarrollo de software hace referencia a un marco que es utilizado para estructurar, planificar y controlar el proceso de desarrollo de sistemas [3]. Son por demás conocidas las metodologías tradicionales, pero en el

Cita sugerida: A. Kuz, M. Falco, R. S. Giandini, "Comprendiendo la Aplicabilidad de Scrum en el Aula: Herramientas y Ejemplos," *Revista Iberoamericana de Tecnología en Educación y Educación en Tecnología*, Nº 21, pp. 62-70, 2018.
DOI: 10.24215/18509959.21.e07

último tiempo se han puesto en auge las metodologías ágiles.

El movimiento ágil busca alternativas a la gestión de proyectos tradicionales, debido a que estos enfoques ayudan a los equipos a responder al comportamiento errático mediante cadencias de trabajo incrementales e iterativas y retroalimentación empírica [4]. La metodología implementada proporciona oportunidades para evaluar la dirección del proyecto a lo largo del ciclo de vida, mediante los sprints o iteraciones; y esto conlleva a que pueda construirse el producto adecuado.

El aprendizaje en este contexto se focaliza en las competencias para el siglo XXI, donde la prioridad es aprender a aprender. Por ello, la incorporación al ámbito educativo de las metodologías ágiles requiere una adaptación al contexto de la enseñanza, en general, y al de la institución educativa y las materias, en particular. De esta manera, las instituciones educativas [5-6] han comenzado a utilizar Scrum para ayudar a los equipos de estudiantes a aprender más eficazmente, de una forma más agradable desarrollando en mayor medida sus capacidades y el trabajo colaborativo, fortaleciendo la actividad docente desde una visión más amplia y renovada.

El objetivo del presente artículo es evidenciar los distintos tópicos que se consideran a la hora de trabajar con Scrum dentro del contexto áulico, vislumbrando los roles, actividades, sprints y artefactos a través de eduScrum. El artículo se estructura de la siguiente manera: en la sección 2, se contextualiza la Ingeniería de Software enfatizando la descripción de las metodologías ágiles. La sección 3 presenta un breve estudio de los marcos XP, Crystal y Scrum. La sección 4 describe Scrum en el aula, describiendo brevemente tres ejemplos de aplicación: eduScrum, Blueprint y aprendizaje ágil. La sección 5, contempla una descripción amplia de eduScrum. Finalmente, se presentan las conclusiones.

2. Ingeniería de Software: Contexto y Metodologías

Debido a las características particulares de los desarrollos educativos, es necesario considerar los aspectos pedagógicos y de la comunicación con el usuario para cada caso en particular junto con la adaptación de los paradigmas a las teorías educativas [7]. Pressman describe a la Ingeniería de Software como una disciplina o área de las Ciencias de la Computación, que ofrece métodos y técnicas que permiten desarrollar y mantener software de calidad [8].

Se compone por una serie de modelos que abarcan los métodos, las herramientas y los procedimientos. Por ello, la Ingeniería de Software Educativo apoya y sustenta el desarrollo de aplicaciones de software que buscan implementar procesos de aprendizaje, y que contendrán aspectos didácticos y pedagógicos en las fases de análisis y diseño [9]. El ciclo de vida de una aplicación educativa propuesta por Galvis [10] presenta dos maneras de ejecución: recorriendo en el sentido de los agujas del reloj, se procede a diseñar, desarrollar y testear; mientras que en el sentido contrario, se testea aquello que puede satisfacer la necesidad.

2.1. Metodologías de desarrollo de software

Ahora bien, no es viable hablar de metodologías universales que puedan ser aplicadas con éxito en cualquier proyecto de software, sino que toda metodología debe ser adaptada al contexto en el que se encuentra el proyecto como el tiempo de desarrollo, los recursos humanos, entre otros.

Por un lado, existen las metodologías tradicionales que como plantean Canós y otros [11], desde sus orígenes abordaron un gran número de contextos de proyectos, que sin dudarlo exigía un esfuerzo relativamente alto con el fin de que puedan ser adaptadas considerando asimismo la existencia de requisitos cambiantes. Un ejemplo sumamente conocido es RUP (por sus siglas en inglés, *Rational Unified Process*) que puede describirse como un proceso de ingeniería de software, que proporciona un enfoque disciplinado para asignar tareas y responsabilidades dentro de una organización de desarrollo [12]. Por el otro, encontramos las metodologías ágiles que proporcionan una solución a medida para un gran número de proyectos. En la Tabla 1, puede encontrarse un esquema comparativo de las metodologías ágiles y tradicionales, en una adaptación de [11].

Tabla 1. Comparación metodologías ágiles vs tradicionales

Tradicionales	Ágiles
Resistencia a los cambios	Preparados para cambios
Impuestas por el equipo	Impuestas externamente
Arquitectura esencial, expresada mediante modelos	Menos énfasis en la arquitectura del software
Más roles	Pocos roles
Más artefactos	Pocos artefactos
Grupos grandes y distribuidos	Grupos pequeños, en el mismo sitio
Proceso controlado, con muchas normas y políticas	Proceso menos controlado, con pocos principios
Proceso rígido	Proceso flexible con adaptación
El cliente interactúa con el equipo de desarrollo	El cliente es parte del equipo de desarrollo

Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo	Basadas en heurísticas provenientes de prácticas de producción de código

Al comienzo del año 2001, diecisiete expertos de la industria del software se reunieron en Snowbird, Utah; con el objetivo de aunar y describir los principios y valores que creían debían permitir a los equipos desarrollar software de forma rápida, respondiendo a los cambios que podrían llegar a presentarse en el transcurso del proyecto. Buscaban ofrecer una alternativa a los procesos de desarrollo de software tradicionales, por lo que en dicha reunión se acuñó el término ágil y esbozaron el manifiesto ágil, un documento que resume dicha filosofía [11]. Tras esta reunión se creó una organización sin ánimo de lucro dedicada a promover los conceptos del desarrollo ágil de software, denominada The Agile Alliance[2].

2.2. Manifiesto y principios ágiles

El manifiesto ágil valora [13] al individuo y las interacciones del equipo de desarrollo sobre el proceso y las herramientas (las personas son el principal factor de éxito de un proyecto software por lo que es fundamental construir un buen equipo de trabajo que el entorno), el desarrollar software que funciona más que conseguir una buena documentación (no producir documentos si no son necesarios), la colaboración con el cliente más que la negociación de un contrato (busca la interacción constante entre el cliente y el equipo de desarrollo) y responder a los cambios más que seguir estrictamente un plan (la planificación debe ser flexible y abierta).

Los valores que dan forma al manifiesto, sustentan a los principios [14] del mismo siendo estos características de los procesos ágiles:

1) La prioridad es satisfacer al cliente mediante tempranas y continuas entregas de software que le aporte un valor.
2) Dar la bienvenida a los cambios y se capturan los cambios para que el cliente tenga una ventaja competitiva.
3) Frecuentemente entregar el software que funcione desde un par de semanas a un par de meses, con el menor intervalo de tiempo posible entre entregas.
4) La gente del negocio y los desarrolladores deben trabajar juntos a lo largo del proyecto.
5) Construir el proyecto en torno a individuos motivados, darles el entorno y el apoyo que necesitan y confiar en ellos para conseguir finalizar el trabajo.
6) El diálogo cara a cara es el método más eficiente y efectivo para comunicar información dentro de un equipo de desarrollo.
7) El software que funciona es la medida principal de progreso.
8) Los procesos ágiles promueven un desarrollo sostenible.
9) La atención continua a la calidad técnica y al buen diseño mejora la agilidad.
10) La simplicidad es esencial.
11) Equipos organizados por sí mismos.
12) En intervalos regulares, el equipo reflexiona respecto a cómo llegar a ser más efectivo, y según esto ajusta su comportamiento.

3. Frameworks ágiles

Se describirán brevemente tres ejemplos de marcos ágiles: Extreme Programming (XP), Crystal y Scrum.

3.1. Extreme Programming

El primer proyecto de programación extrema (en inglés, *Extreme Programming*) se inició a comienzos de marzo de 1996 [15] y permitió a sus desarrolladores responder con confianza a las necesidades cambiantes de los clientes, incluso en el ciclo de vida puesto a que hace hincapié en el trabajo en equipo. En XP los gerentes, los clientes y los desarrolladores son socios iguales en un equipo colaborativo, lo que promueve la autoorganización alrededor del problema. Brinda un entorno sencillo con reglas simples basadas en valores y principios, lo que facilita a los equipos ser altamente productivos. Además de la sencillez; mejora un proyecto de software en la comunicación, la retroalimentación, el respeto y el valor.

3.2. Crystal

Alistair Cockburn desarrolló un conjunto de metodologías que se caracterizan por estar centradas en los integrantes del equipo de desarrollo (siendo este un factor clave) como así también en la reducción del número de artefactos producidos. En esta línea, el desarrollo de software en sí es considerado como un juego cooperativo que posibilita la invención y la comunicación, aunque limitado por los recursos a utilizar. El uso de la palabra "*crystal*" hace referencia a las diversas facetas de una piedra preciosa - cada cara diferente en un núcleo subyacente representa valores y principios, mientras que cada faceta representa un conjunto específico de elementos tales como técnicas, roles, herramientas y estándares [16].

[2] Agile Alliance, www.agilealliance.com

3.3. Scrum

Schwaber y Sutherland [17] explican que Scrum es un proceso o una técnica para construir productos, y un marco que ha sido usado para gestionar el desarrollo de productos complejos desde principios de los años 90. Jeff Sutherland describe en [18] que Scrum nació como una forma nueva y diferente de organizar el esfuerzo humano, en vez de una forma de cómo concebir el trabajo. Este marco recibió un nombre que tuvo su origen en el rugby, donde el Scrum contenía la metáfora perfecta para lo que Sutherland comprendía como trabajo en equipo: acoplamiento, unidad de propósito y claridad de metas. El equipo Scrum incluye tres roles: el *product owner* (decide qué trabajo deberá ser realizado), el *scrum master* (actúa como líder servicial, ayudando al equipo y a la organización a hacer el mejor uso de scrum), y los miembros del equipo de desarrollo (construye el producto en forma incremental, en una serie de sprints) [19].

Como plantean Delhij y van Solingen "*el sprint es un conjunto coherente de material de aprendizaje que logra ciertos objetivos de aprendizaje*" [18]. Un sprint es un período fijo de tiempo con preferencia en los intervalos más cortos. En cada sprint, el equipo Scrum construirá y entregará un incremento del producto, donde cada incremento es un subconjunto reconocible, operativo y visiblemente mejorado del producto, que alcanza criterios de aceptación claros y está construido con un nivel de calidad denominado Definición de Hecho (en inglés, *Definition of Done*). En la Figura 1 puede observarse el resumen de la relación entre los artefactos y las actividades o tareas de Scrum.

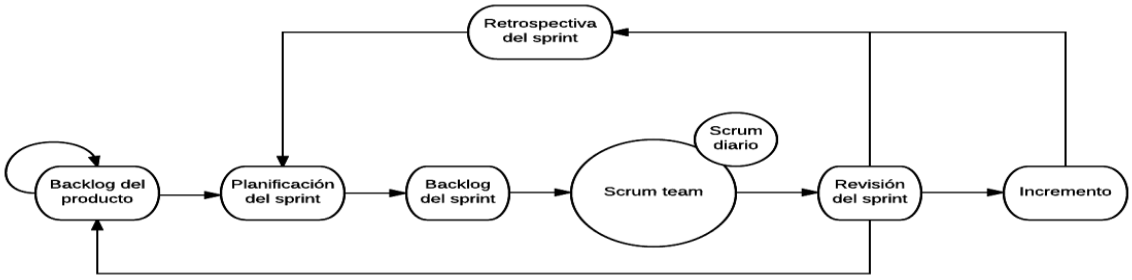

Figura 1. Adaptación de [21]

4. Scrum en el aula

En el ámbito educativo, los estudiantes necesitan desarrollar competencias generales, es decir: capacidades, habilidades y aptitudes que les serán de utilidad en el entorno académico y luego en su carrera profesional. Consecuentemente, las competencias generales requieren de métodos de aprendizaje activos que posibiliten el desarrollo de la capacidad de organización, planificación, liderazgo, evaluación, autoevaluación, trabajo en equipo, entre otros. En esta línea, las metodologías ágiles proporcionan principios, valores y prácticas que pueden encarnar la solución al contexto presentado debido a que la adquisición de las competencias es flexible y sencilla.

Teniendo como base lo anterior, Scrum favorece la creación de un ambiente propicio para que los alumnos sean creativos posibilitando por un lado, que la experiencia áulica sea enriquecedora y confiable, y por el otro, el desarrollo del carácter con una mayor profundidad en el proceso de aprendizaje, vislumbrando el avance del estudio a través de los sprints exitosamente completados. Es decir, que Scrum a través de la inspección, la adaptación y la transparencia, se convierte en un marco de aprendizaje [5].

Es necesario un proceso de análisis para la implantación de Scrum al entorno académico, debido a que es fundamental definir específicamente el contexto donde se aplicará. Dicha implantación presenta dos estadios, el primero concierne a la forma en que se traslada el proceso ágil en el proceso de enseñanza y el segundo, la identificación de cuáles serán los artefactos para el mismo.

John Miller [20] explica que un "aula ágil" (ver Figura 2) está diseñada a través de la integración de cinco (5) elementos, donde cada uno de ellos puede ser combinado en formas diferentes en pos de alcanzar objetivos específicos y el contexto. Precisamente, será más poderosa cuando se la utilice sinérgicamente como un sistema completo. Los elementos son:

1. Clase Visible (en inglés, v*isible classroom*): un sistema de gestión del aprendizaje visual.

2. Ritmo del Aprendizaje (en inglés, *learning rhythm*): un ciclo de aprendizaje iterativo y completo.

3. Colaborar (en inglés, *collaborate*): un modelo de situación para aumentar la capacidad de colaboración y define la relación de aprendizaje entre los estudiantes.

4. Capacitar (en inglés, *empower*): un modelo de situación para aumentar la capacidad de empoderamiento, y define la relación de aprendizaje y los límites de elección entre el profesor y los estudiantes.

5. El viaje (en inglés, *the journey*): un andamio camino para evolucionar cualquier aula hacia la auto-organización. Integra todos los demás elementos.

A continuación, se presentan tres (3) ejemplos de aplicación de Scrum al aula.

4.1. eduScrum

El *Ashram College* en *Alphen aan de Rijn* de los Países Bajos utiliza eduScrum [22], una versión educativa de Scrum para su uso en la Educación Secundaria. eduScrum [18] es un marco que le permite a los alumnos afrontar diversos y complejos problemas, favoreciendo y potenciando el aprendizaje y el crecimiento personal de los alumnos. El aprendizaje se posiciona en el centro del escenario lo que le permite al alumno aprender de forma más inteligente mejorando la colaboración. Esta manera de trabajar, conlleva a que los estudiantes trabajen juntos de manera enérgica, específica y efectiva [6], y fue utilizado en primera medida en las materias Física y Química, pero en la actualidad lo implementan en todas las materias.

4.2. Blueprint

Scrum es utilizado por los estudiantes como una forma atractiva y auto-organizativa de trabajar en colaboración y de forma dinámica en Blueprint High School, con sede en Arizona. Esta Escuela Secundaria es operada por Blueprint Education, una organización sin fines de lucro que se especializa en opciones académicas para estudiantes no tradicionales [6]. John Miller de Agile Schools[1] ha ayudado a Blueprint High School a implementar Scrum como parte de su actividad diaria.

Han puntualizado que una de las razones por las que Scrum fue adaptada favorablemente en el contexto áulico consiste en un cambio de enfoque para con los estudiantes, basado en que en el siglo XXI si bien los alumnos deben obtener el diploma de la Escuela Secundaria, éste no era el final del ciclo. Por ello, ajustaron el objetivo con el fin de prepararlos para la actualidad, con habilidades más allá de la lectura, la escritura y la matemática. Consecuentemente, las habilidades que las metodologías ágiles y Scrum encarnaban eran ideales para satisfacer los objetivos.

En lo que se refiere a los papeles de Scrum es necesario puntualizar que en ciertos contextos, el profesor sirve como propietario principal del producto, como entrenador del Scrum. Otras veces, el equipo de estudiantes actúa como dueño de producto y Scrum Master. Es viable destacar que los sprints duran dos semanas.

4.3. Aprendizaje ágil

En la educación superior, surgió el concepto de aprendizaje ágil (en inglés, *agile learning*) afecta a la forma en la que el alumno aprende y la que el profesor enseña, focalizándose en un modelo de aprendizaje basado en competencias junto con un modelo de evaluación continua [5]. Por ejemplo, ha sido aplicado a ciertas asignaturas de la Escuela Universitaria de Informática en la Universidad Politécnica de Madrid, mediante el uso de Moodle.

En el marco del aprendizaje ágil, se definen el producto final y el *working product* que permiten adoptar Scrum a partir de ellos, siendo el *working product* la entrega de un trabajo o un conjunto de trabajos que el profesor cree que son necesarios para determinar la serie de conocimientos y competencias evaluables de los alumnos. Luego, la acción de entregar el producto final implica el haber aprobado la asignatura es decir, que está constituido por el conjunto de *working products* que el alumno entrega al profesor. Por ello, se obtiene a partir de un proceso iterativo e incremental de aprendizaje.

En lo que respecta a la adopción de nuevas prácticas y metodologías no es viable llevar a cabo la implantación completa de Scrum en un único paso en el sistema educativo. Por esta razón, *Agile Learning* define una hoja de ruta que permite llevar a cabo dicha adopción en el ámbito universitario de forma iterativa, incremental y viable. Las fases de la hoja de ruta son las siguientes:

1) Definición del equipo Scrum (número de alumnos involucrados, trabajos a realizar, carácter del equipo – transversal a diversas asignaturas o independientes)
2) Definición de los sprint (dura entre 2 y 4 semanas)
3) Selección de prácticas ágiles a adoptar
4) Adopción y adaptación de las prácticas seleccionadas al contexto

5. eduScrum

El modelo de equipo en eduScrum está diseñado para la óptima autonomía, la colaboración, la flexibilidad, la creatividad, la motivación y la productividad. Para entender cómo aplicarlo, es fundamental comprender con anterioridad las partes que constituyen un equipo y que pueden ser visualizadas en la Figura 2.

[1] **Agile Schools**: es una organización australiana que se asocia con escuelas y sistemas a nivel global para brindarles ayuda al desarrollar la capacidad de mejora continua y maximizar el aprendizaje. http://www.agileschools.com/

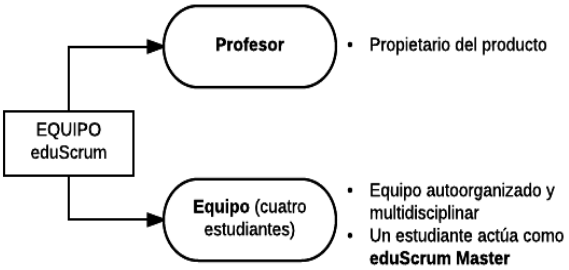

Figura 2. Equipo eduScrum (fuente propia)

El profesor como propietario del producto

Es responsable de determinar qué es lo que debe aprenderse, supervisar y mejorar la calidad de los resultados educativos, y evaluar dichos resultados siempre basándose en la definición de "terminado" y en los criterios de aceptación. El profesor se centra expresamente en el tema y debe fomentar el trabajo cooperativo entre los equipos. Define los criterios de aceptación que le permiten supervisar la calidad de lo que se ha aprendido, por ejemplo: los resultados mínimos de las evaluaciones. Los docentes actúan a veces como súper scrum masters, para el propósito de aprendizaje o enseñanza.

El equipo de estudiantes

Los estudiantes son autónomos y colaboran para alcanzar los objetivos de aprendizaje que son requeridos al final del sprint teniendo en cuenta los criterios de aceptación. Como puede observarse en la Figura 3, uno de los cuatro estudiantes se desempeña como eduScrum Master (elegido por los alumnos). Los equipos son auto-organizados, es decir que ninguna persona puede decirle al equipo cómo deben conseguir los objetivos del aprendizaje. Son multidisciplinares, esto es que cuentan con las habilidades para alcanzar juntos los objetivos de aprendizaje y el desarrollo personal. Vale destacar que los estudiantes pueden tener habilidades específicas, pero la responsabilidad es grupal.

El eduScrum Master es un líder que prepara y atiende [6] al equipo y trabaja con éste en simultáneo; es decir que no dirige al equipo sino que busca lograr el rendimiento óptimo. Le corresponde iniciar y facilitar los eventos de eduScrum, ejecutar los eventos y utilizar los instrumentos correctamente. También, debe atender al propietario del producto creando transparencia sobre el progreso del equipo con "la hoja" (el tablero de Scrum) actualizada y disponible.

5.1. Comprendiendo el proceso

Generalmente, los sprints coinciden con períodos bien establecidos y cada uno de ellos consiste en los puntos que se enumeran a continuación. Es conveniente destacar que la composición del equipo durante el sprint y el alcance no se cambian, mientras que la calidad puede ser renegociada con el profesor. De la misma manera, y a diferencia de Scrum, un sprint no puede ser cancelado en eduScrum pero sí es posible que se proporcionen asignaciones adicionales para lograr los resultados esperados.

5.1.1. La formación del equipo

Son necesarias buenas composiciones de equipos donde las cualidades de los integrantes sean complementarias, que haya una proporción equilibrada entre hombres y mujeres, que varíen entre asignación y asignación, y que no estén basadas en lazos de amistad.

5.1.2. El equipo de estudiantes

Representan la suma de todos los elementos que tienen que ser completados durante un sprint y son ellos los que brindan la flexibilidad necesaria a los estudiantes sobre qué y cómo entregarán el sprint. Son parte de las condiciones finales y pueden ser vistos como hitos en el progreso del equipo.

5.1.3. La planificación del sprint

Hace referencia al trabajo colaborativo del equipo de estudiantes. En primer lugar, el profesor brinda una visión general de la asignación, el número de lecciones por sprint, la gestión en fechas, los momentos centrales, los modelos evaluación, entre otros. Luego, explica los objetivos del aprendizaje y es el equipo de estudiantes el que determina las actividades necesarias, organiza cronológicamente las tareas y las entregas parciales. Los sprints duran dos semanas y la mayoría contiene entre cuatro (4) y seis (6) lecciones de cincuenta (50) minutos.

5.1.4. Las reuniones de pie

Cada una de ellas consta de cinco (5) minutos, donde el equipo realiza un plan hasta la próxima reunión. Durante cada una de ellas, cada miembro del equipo debe expresar qué es lo que ha llevado a cabo con el fin de conseguir el objetivo del sprint desde la clase anterior, qué hará en ese clase y cuáles cree que son los impedimentos que no permiten al equipo o a sí mismo alcanzar el objetivo del sprint.

5.1.5. La revisión del sprint

Los equipos muestran lo que han aprendido en el último sprint, lo que se cotejará con el objetivo de aprendizaje y la definición de terminado.

5.1.6. La retrospectiva del sprint

Permite generar un plan para la mejora del equipo determinando cómo resultó el último sprint con respecto a las personas, relaciones, procesos y herramientas. Los estudiantes llevan a cabo la evaluación de las metodologías y métodos de trabajo, identifican puntos de mejora, evalúan a los miembros de su equipo (y a sí

mismos), y debaten lo que deberían dejar de hacer. A partir de lo anterior, aprenden juntos de forma efectiva y eficiente.

5.2. Describiendo los artefactos

La pila del producto (en inglés, *Product Backlog*) representa una lista ordenada de elementos, que conforman los objetivos de aprendizaje y los métodos de trabajo que cumplan con estos objetivos fundamentales, siendo el Propietario del Producto el responsable del contenido, la disponibilidad y la ordenación de la pila. Considerando que se encuentra ordenada en base al programa de aprendizaje, los objetivos de aprendizaje y las historias deben ajustarse al programa de aprendizaje global, donde los elementos superiores se refieren al próximo sprint, de manera cronológica. Los elementos de la pila son tratados por el equipo de estudiantes de manera que sean descompuestos de forma tal que cualquier elemento puede ser terminado dentro de la duración del Sprint.

La hoja es una representación cronológica del trabajo del sprint que permite alcanzar los objetivos de aprendizaje, y las tareas y asignaciones se van moviendo de acuerdo a cómo van cambiando de estado: pendiente, ocupado y terminado - los alumnos mueven las tareas del panel to-do (pendiente) al done (terminado) pasando por "in progress" (ver Figura 3).

Figura 3. Alumnos trabajando en el tablero [22]

En los tableros pueden encontrarse dos tipos de tarjetas (cada una de ellas es un capítulo de un tema): historias (cuyo dueño es el profesor) y tareas (los dueños son los estudiantes). Ahora bien, en el primer día del sprint los equipos toman cada capítulo y lo descomponen en varias tareas por lo que cuando todas las tareas de la historia estén completas, todos los alumnos han aprendido ese capítulo.

Deben tratarse los impedimentos como por ejemplo diversas dudas que deban ser consultadas al profesor, entre otros; posibilitando la auto-organización de los alumnos a la hora de repartir el trabajo, explicar unos a otros y demás. Es hacedero destacar que la mayor parte del tiempo de clase es realmente tiempo efectivo de estudio y aprendizaje colaborativo, posibilitando también la retrospectiva sobre cómo mejorar el proceso que han seguido y qué partes van a eliminar, mantener o mejorar en el siguiente sprint.

Cuando todas las tareas de un capítulo se encuentren en el panel "*done*" entonces los restantes equipos y el profesor comprenden que los estudiantes de ese grupo ya han comprendido el tema, por lo que el docente puede hacerles preguntas sobre esos capítulos aprendidos y en caso de que no dominen alguno de ellos, recoloca la tarjeta a "*in progress*".

Es factible mencionar que por añadidura a la definición de "done" se presenta la definición de "divertido", puesto que la diversión es un excelente e importante motivador para los alumnos lo que lo convierte en esencial para conseguir mejores resultados de aprendizaje.

Para finalizar, es fundamental focalizar la atención en los aspectos cuantitativos y cualitativos de la calidad educativa. En años anteriores, se efectuó un estudio con 230 estudiantes entre doce (12) y diecisiete (17) años, a partir del cual se encontró por un lado y con respecto al aspecto cuantitativo, que los resultados de los exámenes de las clases que trabajaban con eduScrum eran más altos que aquellas que no lo usaban.

Por el otro y haciendo referencia al aspecto cualitativo, aproximadamente la mitad de los estudiantes notaban una mejora en las materias porque se habían divertido, habían aprendido en una manera más inteligente y habían obtenido mejores notas. También, tres cuartos de ellos mostraron mejoras en la cooperación vislumbrando de la misma manera que un 60% de los alumnos reportaban que conocían sus propias cualidades y limitaciones de mejor manera, con un 40% que se sentían con mayor confianza.

Según observamos también en un artículo escrito por Aubin et al [23] que una experiencia práctica realizada en la cátedra de Programación Avanzada de la Universidad Nacional de la Matanza (UNLaM). En este ariculo ellos revelan las ventajas encontradas de la aplicación en el aula de la programación de a pares, que es uno de los aspectos significativos de las metodologías XP (Extreme Programming), y se muestran las mejoras obtenidas en el rendimiento académico de los alumnos. En este caso las estadísticas de esta aplicación práctica revelan en sus experiencias que van desde el año 2008 al 2013; aportes positivos y cierta su robustez, pero en contra partida revelan que se requiere un cambio cultural.

De la misma manera, Barberis y Del Moral Sachetti exponen en [24] una experiencia metodológica que tiene como fin resaltar no solo la enseñanza sino la práctica cooperativa en pos de adquirir las habilidades del programador en el desarrollo ágil de software. Han realizado experiencias sobre dos asignaturas: Programación Numérica y Cálculo Numérico de la carrera Licenciatura en Análisis de Sistemas de la Universidad Nacional de Salta. Dichas autoras destacan en primer lugar, que la retroalimentación constante entre profesor-profesor y profesor-alumno ha sido en gran parte la clave del éxito; en segundo lugar, que se logró mejorar la calidad cognitiva del alumnado; y en tercer

lugar, se contribuyó con la disminución de la tasa de deserción e incrementar los índices de rendimientos.

Conclusiones

Como se presentó al inicio del artículo, la Ingeniería de Software ha sufrido diversos cambios, donde las metodologías ágiles pueden ajustarse a un amplio rango de proyectos de desarrollo de software donde los equipos de desarrollo son reducidos, con plazos cortos, requisitos volubles, basados en TICs, que necesitan de una solución a medida, simple y con altos niveles de calidad. Estas metodologías se centran en las personas y en el producto software, valorando el trabajo colaborativo [25].

Como explican Orjuela Duarte y Rojas citando a [26], la agilidad es "*la habilidad para balancear flexibilidad y estabilidad*". La metodología que sustenta el desarrollo de software educativo especifica un conjunto de fases que guían el proceso de desarrollo de software que sirvan de apoyo al proceso de enseñanza-aprendizaje.

En el ámbito educativo, la aplicación de metodologías ágiles puede verse en ejemplos como eduScrum, Blueprint [6] y aprendizaje ágil [5]. eduScrum es un *framework* que se fundamenta en la teoría de control de procesos empíricos, la cual afirma que el conocimiento proviene de la experiencia y que se toman decisiones basándose en lo que se conoce. También, utiliza un enfoque iterativo e incremental que permite optimizar la consecución de los objetivos de aprendizaje y el control de riesgo. En lo que respecta a la capa pedagógica, es viable mencionar que permite lograr un aprendizaje eficaz y eficiente debido a que el trabajo colaborativo es un elemento clave de eduScrum.

Las instituciones educativas utilizan Scrum con el fin de que los estudiantes logren aprender de forma eficaz, debido a que los equipos se auto-organizan y trabajan en sprints lo que les permite aprender temas evolucionando el proceso de aprendizaje. A través de este método de trabajo ágil es viable mejorar la calidad de las clases y del aprendizaje, motivando en mayor medida a los estudiantes. Los actores utilizan revisiones y retrospectivas para evaluar los procesos de aprendizaje en el mismo sprint, lo que permitirá introducir mejoras en el próximo sprint, si fuera necesario [24].

Scrum en educación permite llevar al alumno a un papel donde asume su propio aprendizaje y puede determinar el tiempo que prepara y dedica para estudiar. El aprendizaje se convierte en una acción de colaboración, en el que la asimilación del individuo afecta y contribuye al éxito del aprendizaje permitiendo el crecimiento de todo el equipo. La revisión en cada sprint permite monitorear en tiempo real el avance de los estudiantes, cómo trabaja cada equipo y solucionando problemas específicos.

Pero uno de los aspectos problemáticos en lo que concierne a la implementación de la estrategia de aprendizaje basado en proyectos es la exigencia de parte del docente de asumir un rol de facilitador y orientador, con un conjunto de habilidades de gestión de procesos a las que habitualmente no se tiene acceso en la formación académica ni en la experiencia de enseñanza tradicional.

Algunos docentes tal vez encuentren dificultades a la hora de implementar las ceremonias e instrumentos todos juntos al mismo tiempo, por lo que es recomendable realizar la implantación de una vez en el aula. Consideramos que es viable la implementación de Scrum en el ámbito académico, paulatinamente y por materias debido a que prepara y capacita a los alumnos para las demandas de la sociedad tecnológica y en constante movimiento en la que vivimos.

Referencias

[1] O. R. Sanmartín (03/02/2016) *Así son las escuelas más innovadoras del mundo*. [online] El Mundo. Disponible en: http://www.elmundo.es/sociedad/2016/02/03/56b117fa22601d906e8b45e5.html [accedido 28 Feb. 2017]

[2] IEEE Standard Glossary of Software Engineering Terminology," IEEE std 610.12-1990, 1990. ISBN 155937067X.

[3] Centers for Medicare & Medicaid Services, Selecting a development approach. 2008 https://www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/XLC/Downloads/SelectingDevelopmentApproach.pdf

[4] Agile Methodology, http://agilemethodology.org/ [accedido 28 Feb. 2017]

[5] J. Pérez Benedi, A. Yagüe Panadero, J. Díaz Fernández, S. Alonso Villaverde (2011) Un primer paso a la agilidad: retrospectivas para el aprendizaje de la Ingeniería del SW. En: "2 Conferencia Agile-Spain CAS 2011", 20/10/2011 - 21/10/2011, Castellón, España. pp. 11-24.

[6] InfoQ (Agosto, 2013) Scrum for Education – Experiences from eduScrum in Education [online] Disponible en https://www.infoq.com/articles/scrum-education [accedido 28 Feb. 2017]

[7] Z. Cataldi, F. Lage, R. Pessacq, R. García Martínez. Ingeniería de Software Educativo, En: Proceedings del V Congreso Internacional de Ingeniería Informática (pp. 185-199), Agosto 1999.

[8] R. S. Pressman, Ingeniería del Software: Un Enfoque Práctico, Mcgraw-Hill / Interamericana, España, 2001.

[9] R. A. Gómez Castro, A. H. Galvis Panqueva, O. Mariño Drews, "*Ingeniería de Software Educativo con Modelaje Orientado por Objetos: Un Medio Para Desarrollar Micromundos Interactivos*". In: Informática Educativa 11. 1 (1998): 9-30.

[10] A. H. Galvis, Ingeniería de Software Educativo. Santafé de Bogotá: Ediciones Uniandes, 1992.

[11] J. H. Canós, P. Letelier y MC Penadés. *"Metodologías Ágiles en el Desarrollo de Software*". En: Actas Metodologías Ágiles en el Desarrollo de Software VIII Jornadas de Ingeniería del Software y Bases de Datos, JISBD 2003. Alicante, del 12 al 14 de Noviembre de 2003.

[12] IBM, Rational Unified Process: Best Practices for Software Development Teams https://www.ibm.com/developerworks/rational/library/content/03July/1000/1251/1251_bestpractices_TP026B.pdf

[13] Manifiesto por el Desarrollo Ágil de Software. Disponible en http://agilemanifesto.org/iso/es/manifesto.html [accedido 28 Feb. 2017]

[14] Principios del Manifiesto Ágil. Disponible en: http://agilemanifesto.org/iso/es/principles.html [accedido 28 Feb. 2017]

[15] Extreme Programming, Disponible en: http://www.extremeprogramming.org/ [accedido 28 Feb. 2017]

[16] M. Alexandrou, (s.f), Crystal Methods Methodology Disponible en: http://infolific.com/technology/methodologies/crystal-methods/ [accedido 28 Feb. 2017]

[17] A. Delhij, R. van Solingen, W. Wijnands, (Sep. de 2015.), La guía de eduScrum: "las reglas del juego". Disponible en: http://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-es.pdf [accedido 28 Feb. 2017]

[18] J. Sutherland, Scrum: El arte de hacer el doble de trabajo en la mitad de tiempo. Editorial Oceano, Apr 1, 2016 - Business & Economics.

[19] Scrum Alliance "Core Scrum—Values and roles." (2013). Recuperado de https://www.scrumalliance.org/scrum/media/ScrumAllianceMedia/Files%20and%20PDFs/Learn%20About%20Scrum/Core-Scrum.pdf [accedido 28 Feb. 2017]

[20] John Miller, (Noviembre 2016), 5 Elements of Agile Classrooms, Disponible en: http://blog.agileclassrooms.com/2016/11/5-elements-of-agile-classrooms.html [accedido 28 Feb. 2017]

[21] Scrum.org (s.f), What is Scrum? Recuperado de https://www.scrum.org/Resources/What-is-Scrum, [accedido 28 Feb. 2017]

[22] eduScrum, (s.f.) About eduScrum, retrieved from: http://eduscrum.nl/en/about-eduscrum, [accedido 28 Feb. 2017]

[23] V. Aubin, L. Blautzik, G. Dejean. Mejoras en el proceso de Enseñanza-Aprendizaje de programación utilizando metodologías de la industria del software. CoNAIISI 2013, UTN Facultad Regional Córdoba.

[24] Barberis, Á. R., & Moral Sachetti, L. E. D. (2016, August). Scrum como herramienta metodológica en el entrenamiento cooperativo de la programación: de la teoría a la práctica. In XI Congreso de Tecnología en Educación y Educación en Tecnología (TE&ET 2016).

[25] A. Orjuela Duarte, M. Rojas C. Las Metodologías de Desarrollo Ágil como una Oportunidad para la Ingeniería del Software Educativo. In: Revista Avances en Sistemas e Informática, Vol.5 No.2, Junio de 2008, Medellín, ISSN 1657766.

[26] Highsmith, J. Agile Project Management, 2000.

Información de contacto de los autores:

Antonieta Kuz
Universidad Tecnológica Nacional, FRLP
60 esq. 124
La Plata, Buenos Aires
Argentina
akuz@frlp.utn.edu.ar

Mariana Falco
Universidad Tecnológica Nacional, FRLP
60 esq. 124
La Plata, Buenos Aires
Argentina
mfalco@frlp.utn.edu.ar

Roxana Giandini
LIFIA, UNLP
Calle 50 y 120
La Plata
Argentina
rgiandini@frlp.utn.edu.ar

Antonieta Kuz.
Ingeniera en Sistemas de Información. Doctora en Ciencias de la Computación. Docente Universitaria UMET- UTN FRLP

Mariana Falco.
Ingeniera en Sistemas de Información. Docente universitaria. Investigadora en tecnología en educación, aplicaciones del Análisis de Redes Sociales y agentes inteligentes en el aula en PID UTN-FRLP.

Roxana S. Giandini.
Doctora en Ciencias de la Computación, UNLP. Docente Universitaria. Investigadora.