SCRUM como Herramienta Metodológica para el Aprendizaje de la Programación

SCRUM as a Methodological Tool for Programming Learning

Nicolás Tymkiw[1], Juan Manuel Bournissen[1], Marisa Cecilia Tumino[1]

[1]Universidad Adventista del Plata, Libertador San Martín, Entre Ríos, Argentina

nicolas.tymkiw@uap.edu.ar, juan.bournissen@uap.edu.ar, marisa.tumino@uap.edu.ar

Recibido: 05/12/2019 | **Aceptado**: 20/04/2020

Cita sugerida: N. Tymkiw, J. M. Bournissen, M. C. Tumino, "SCRUM como Herramienta Metodológica para el Aprendizaje de la Programación," *Revista Iberoamericana de Tecnología en Educación y Educación en Tecnología*, no. 26, pp. 79-87, 2020. doi: 10.24215/18509959.26.e9

Resumen

El objetivo de este estudio fue conocer el impacto de la utilización de la metodología de desarrollo de software SCRUM, como una técnica que permite reforzar las estrategias de enseñanza y aprendizaje utilizadas en la educación superior en la asignatura de Programación II, en el aprendizaje percibido de los estudiantes en una universidad privada. A partir de una intervención, consistente en la implementación de la metodología SCRUM como estrategia de enseñanza y aprendizaje de la programación, y de la recolección de datos referidos al aprendizaje percibido de los estudiantes mediante una escala construida y validada por Tumino y Bournissen, el estudio ha proporcionado datos que permiten estimar que la aplicación de la metodología SCRUM utilizada tiene un impacto positivo en el proceso de aprendizaje de los estudiantes, ofreciendo una alternativa innovadora a las estrategias tradicionales utilizadas por la cátedra donde se desarrolló la experiencia.

Palabras clave: Metodología SCRUM; Aprendizaje de programación; Impacto en aprendizaje percibido.

Abstract

The objective of this study was to know the impact of the use of the SCRUM software development methodology, as a technique that allows reinforcing the teaching and learning strategies used in higher education in the subject of Programming II, in the perceived learning of students at a private university. Based on an intervention, consisting of the implementation of the SCRUM methodology as a programming teaching and learning strategy, and the collection of data referring to the perceived learning of students through a scale constructed and validated by Tumino and Bournissen, the study has provided data that allow estimating that the application of the SCRUM methodology has a positive impact on the students' learning process, offering an innovative alternative to the traditional strategies used in the courses where the experience was developed.

Keywords: SCRUM methodology; Learning programming; Impact on perceived learning.

1. Introducción

En la actualidad, dentro de las carreras informáticas, la programación es una asignatura fundamental, debido a que es una de las materias base que sustenta el progreso del alumno en la carrera y luego, una vez recibido, orienta su desempeño profesional. Una de las mayores preocupaciones del docente se centra en que el estudiante logre la comprensión completa de los contenidos de programación.

Investigadores como Costelloe [1] muestran que los estudiantes no logran conceptualizar todos los temas de programación enseñados, siendo este uno de los puntos principales por el cual abandonan la carrera. Otro punto que enfatizan los autores es que, si logran aprobar la materia más adelante, el rendimiento en asignaturas correlativas es muy bajo.

Esta metodología requiere de un proceso evaluativo, en el que se revean las practicas docentes, analizando los obstáculos y dificultades que se presentan en los estudiantes.

Barberis, y Del Moral Sachetti [2], señalan que desde que implementaron esta metodología en el año 2014, la tasa de deserción bajó un 23% con respecto a los años anteriores, mientras que el rendimiento académico aumentó un 27%.

Perazo [3] afirma que el porcentaje de deserción en las carreras informáticas se aproxima al 80%, teniendo en cuenta los datos suministrados por la Secretaría de Políticas Universitarias del Ministerio de Educación. Esta problemática podría deberse a que, en su mayoría, es una decisión tomada por adolescentes entre 17 y 20 años que desconocen las incumbencias de la profesión elegida.

En este contexto se espera describir el potencial rol de SCRUM en el proceso de enseñanza y aprendizaje de la programación.

1.1. Beneficios de SCRUM

Albaladejo [4] menciona que SCRUM cuenta con beneficios, fundamentos y requisitos, entre los que cabe mencionar los que se detallan a continuación.

- Entrega de resultados dentro de un plazo corto, mensual o quincenal, con los requisitos prioritarios atendidos.
- Aporte a la gestión de las expectativas del cliente, quien establece sus expectativas y asigna tanto el valor de cada uno de los requisitos como el tiempo en el que espera que se encuentren completados.
- Resultados anticipados, con los que el cliente puede utilizar los más importantes antes de que el proyecto termine, lo que le permite recuperar su inversión con anterioridad y comenzar a utilizar un producto al que solo le faltan características poco relevantes. Esto lo consigue mediante la priorización de requisitos por coste y valor, en el que el cliente señala la prioridad de los requisitos en base al valor que le aportan.
- Flexibilidad y adaptación, donde el cliente va dirigiendo el proyecto en función de sus nuevas prioridades; lo que se consigue replanificando los requisitos y prioridades en el inicio de cada iteración.
- Retorno de Inversión (ROI) mediante el que el cliente maximiza el beneficio del proyecto. En el caso en que el beneficio sea menor al coste del desarrollo, el cliente puede finalizar el proyecto. El ROI se consigue mediante la priorización de los requisitos medidos por valor. En cada una de las iteraciones el cliente dispone de los requisitos completados y replanifica en función del valor que le aportan.
- Definición de los riesgos desde la primera iteración para poder mitigarlos. Si hay que equivocarse, es mejor hacerlo lo antes posible. Para ello es necesario implementar un desarrollo iterativo e incremental de manera de no dejar para el final alguna actividad riesgosa vinculada con la entrega de requisitos.
- Mayor productividad y calidad puesto que el equipo mejora y simplifica su manera de trabajar en cada iteración, al analizar en retrospectiva su método de trabajo, mediante una comunicación continua.
- Trabajo conjunto entre cliente y equipo desde el planteamiento de los requisitos, los detalles y el análisis de los resultados obtenidos.
- Equipo autogestionado y motivado, donde las personas pueden usar su creatividad para resolver problemas y decidir cómo organizar su trabajo a fin de completar todos los requisitos determinados en una iteración.

1.2. Componentes de SCRUM

La metodología SCRUM se encuentra dividida en fases y roles. Las fases se pueden identificar como reuniones también conocidas como Sprint.

Reuniones. Gallego [5] divide las reuniones en tres fases: (a) en la primera fase se desarrolla una planificación del Backlog, dentro del cual se definen las prioridades de los requisitos y la planificación del primer sprint 0, con los objetivos y el trabajo que se necesita completar en esa iteración. Durante esta fase se obtiene la lista de tareas a realizar; (b) para la segunda fase se genera un seguimiento del sprint. Es una fase de reuniones diarias donde se evalúa el lugar del proyecto en el que se encuentra el equipo, los avances desde la última reunión, los trabajos que se realizarán hasta la siguiente reunión y se analiza la forma de solucionar los inconvenientes que surgieron durante el desarrollo y (c) en la tercera fase, cuando concluye el Sprint, se aplica una revisión del incremento que se ha generado, se presentan los resultados finales y una versión de ayuda para mejorar la realimentación del cliente.

Roles. Los roles se dividen en dos grupos: (a) los que están comprometidos con el proyecto y proceso de SCRUM y (b) los que no son parte del proceso pero que se necesitan para la realimentación de la salida de los procesos y el planeamiento de los sprint. En el primer grupo se pueden identificar a los integrantes en tres áreas:

Product Owner: es el encargado de tomar las decisiones del proyecto y el conocedor del negocio del cliente y su visión sobre el producto. Es responsable de recolectar los requisitos del cliente y ordenarlos por prioridad.

SCRUM Master**:** comprueba que la metodología funciona, para lo que debe sobrepasar los inconvenientes que obstaculicen la fluidez del proceso e interactuar con el cliente y los gestores.

Equipo de Desarrollo: conformado por un grupo pequeño de personas y tienen la autoridad para organizar y tomar decisiones para conseguir su objetivo.

Dentro del segundo grupo se pueden encontrar los siguientes roles:

Usuarios: son los destinatarios finales del producto.

Stakeholders**:** son los que participan de las revisiones del sprint.

Managers**:** encargados de la toma de decisiones finales, participan en la selección de los objetivos y los requisitos del proyecto.

1.3. SCRUM en el aula

Kuz, Falco y Giandini [6] explican que, dentro del ámbito educativo, los estudiantes necesitan desarrollar capacidades y aptitudes que posteriormente le servirán en su carrera profesional. Las competencias generales requieren de métodos de aprendizaje activos que posibiliten el desarrollo de la capacidad de organización, planificación, liderazgo, evaluación, autoevaluación y trabajo en equipo, entre otros.

Teniendo en cuenta lo mencionado anteriormente, SCRUM favorece la creación de un ambiente propicio para que los estudiantes puedan experimentar un aprendizaje creativo, enriquecedor y confiable.

Miller [7] explica que un "aula ágil" precisa de la integración de cinco elementos fundamentales para alcanzar objetivos específicos: (a) clase visible, (b) ritmo de aprendizaje, (c) colaboración, (d) capacitación y (e) el camino para evolucionar un aula hacia la autoorganización.

1.4. SCRUM en programación

Varios autores muestran que la utilización de metodologías ágiles, como SCRUM, pueden ser altamente eficientes al desarrollar un software.

Chávez Andrade [8] indagó sobre la estandarización de los procesos de desarrollo utilizando buenas prácticas de programación y SCRUM en departamentos de TI. Tuvo como objetivo incrementar el grado de funcionalidad de un producto de software. Afirma que la carente aplicación metodológica conlleva a obtener un producto de baja calidad. Explica que la aplicación de SCRUM logra la integración del usuario al proceso de desarrollo, así como genera un ambiente de compromiso en el equipo de desarrollo con el fin de obtener un producto de software de calidad, que cumpla con las aspiraciones del usuario. Se identificó la mejora de funcionalidad como atributo de la calidad por parte de los implicados en el proyecto.

Hervás Lucas [9] sostiene que, para alcanzar un aprendizaje eficiente en un lenguaje de programación, se emplean técnicas de integración continua. La investigación desplegada con alumnos del tercer curso de ingeniería informática consistió en el aprendizaje con el método basado en el desarrollo guiado y supervisado de prototipos de complejidad incremental. La conclusión a la que llega el autor es que el método prototipado ágil fomenta el aprendizaje y el desarrollo de competencias clave como el trabajo en equipo, la autodidactica y el análisis crítico.

1.5. Evaluación de Impacto

A partir de las afirmaciones precedentes, se considera relevante proponer una manera de evaluar los resultados derivados de la implementación de la metodología SCRUM como estrategia de enseñanza y aprendizaje de la programación en los estudiantes. El impacto resulta ser un concepto adecuado para identificarlos.

Para definir el concepto de impacto de un proceso educativo, se toma como modelo la definición de Aguilar, [10].

El impacto de un proceso docente- educativo se traduce en sus efectos sobre una población amplia: comunidad, claustro, entorno, estudiante, administración, identificando efectos científico - tecnológicos y económico - social - cultural - institucional, centrados en el mejoramiento profesional y humano del hombre y su superación social.

Tejada Ferández [11] señala que la evaluación del impacto intenta responder la siguiente pregunta. ¿Cuáles son los efectos que la formación produce en un determinado ámbito? Se intenta averiguar en qué medida los aprendizajes adquiridos en las acciones formativas son útiles para mejorar el desempeño.

El proceso de evaluación de acciones formativas cuenta con tres finalidades esenciales: (a) diagnóstica, que permite conocer el desarrollo del proceso de enseñanza y aprendizaje, (b) formativa, que permite valorar una acción educativa durante su desarrollo y (c) sumativa o de productos, que permite conocer si los objetivos se consiguieron o no, los cambios que se produjeron y la toma de decisiones con respecto a las actividades planteadas.

Dado que el impacto de la implementación de la metodología SCRUM debe evaluarse, se considera sustancial partir del concepto de evaluación de impacto, entendido como "el proceso evaluativo orientado a medir los resultados de las intervenciones, en cantidad, calidad y extensión según las reglas preestablecidas" Abdala, [12]. No obstante, para comprender el grado de impacto que las herramientas pueden ejercer en la educación, primeramente, se debe conocer su potencial académico.

2. Metodología

El presente estudio es de tipo explicativo y cuasiexperimental, puesto que busca identificar el nivel de impacto en el aprendizaje percibido en dos grupos definidos por la estrategia de enseñanza y aprendizaje implementada en una clase de programación. El diseño metodológico estuvo basado en un estudio desarrollado por Barberis y otros, [2].

La variable independiente en esta investigación fue la metodología utilizada y las variables dependientes: (a) el nivel de impacto en el aprendizaje percibido de los estudiantes y (b) las calificaciones promedio cuatrimestrales.

2.1. Implementación

Durante los últimos años se ha identificado que en la asignatura de Programación II existía un alto porcentaje de alumnos que perdían la cursada y otro porcentaje, menor pero significativo, de alumnos que decidían abandonarla.

A partir de esta problemática, y con el fin de que los estudiantes pudieran adquirir los conocimientos propuestos por la asignatura y mejorar sus calificaciones, se implementó una nueva estrategia de enseñanza que tuvo el objetivo de potenciar el trabajo de cada uno de los estudiantes.

El plan de trabajo consistió en aplicar la metodología SCRUM, como estrategia de enseñanza y aprendizaje de la programación, en un curso de Programación II de la carrera de Ingeniería en Sistemas de Información de una universidad de Argentina. Se dividió al curso en equipos de tres estudiantes, a quienes se les explico la metodología a implementar.

Durante el cursado de la asignatura a cada equipo se le asignaron trabajos prácticos que iban incrementando el nivel de complejidad a medida que se avanzaba en la complejidad de los conceptos abordados.

Los trabajos prácticos consistieron en desarrollos incrementales de aplicaciones, haciendo uso de los conceptos que se iban introduciendo.

Los estudiantes debían cumplimentarlos en un plazo de tres semanas, siguiendo una serie de requisitos:

a. La composición del grupo durante un trabajo practico debía contar con un líder de SCRUM, mientras que los demás estudiantes asumían el rol de equipo de desarrollo.
b. Cada uno de los estudiantes tenía que pasar por el rol de líder de SCRUM.
c. El grupo debía realizar al menos un sprint (reunión de equipo) por semana, donde a través de un brainstorming podan elegir la forma de desarrollar los requisitos propuestos o solucionar los problemas que fueron apareciendo durante el transcurso del desarrollo.
d. Cada semana el grupo se reunía con el docente de la catedra, quien asumía el rol de cliente, verificaba el cumplimiento de los requisitos planteados e incrementaba los requisitos de acuerdo con los nuevos conceptos tratados en clase.
e. En la tercera semana los equipos debían exponer, frente a la clase, la resolución del trabajo práctico que se les había asignado. En esta exposición debían relatar los procedimientos seguidos para desarrollarlo, los problemas que habían surgido y las soluciones encontradas para seguir adelante. Por su parte el líder SCRUM debía describir la forma en que se habían realizado los sprints y como había sido el trabajo del equipo con el cliente. Por último, cada uno de los estudiantes debía mostrar la parte de la aplicación que había desarrollado, explicando las razones por las que había procedido de ese modo. Esto último serviría para verificar el nivel de comprensión de los conceptos aprendidos en clase.

Además de los trabajos prácticos, las calificaciones se tomaron en base a dos evaluaciones integrales. La primera consistió en un examen individual en el que los estudiantes desarrollaron un sistema con los conceptos aprendidos hasta el momento. La segunda evaluación tuvo una temática distinta a la primera, donde el docente le asignó a cada equipo una problemática distinta y les asignó tres horas para que elaboraran un sistema que solucionara dicha problemática.

La cursada finalizo con un trabajo final, de un grado de dificultad mayor al resto de los trabajos prácticos realizados, en el que se integraban todos los conceptos vistos durante el cuatrimestre. Para calificarlo se les solicitó a cada equipo que, además de cumplir con cada uno de los requerimientos, añadan funcionalidades extras y dediquen tiempo al diseño del software.

2.2. Sujetos

La muestra estuvo compuesta por estudiantes de la carrera de sistemas que cursaban Programación II (grupo experimental) y estudiantes que habían cursado los dos años anteriores (grupo de control). Cabe señalar que la cantidad de cursantes de la asignatura no es superior generalmente a los diez estudiantes, por lo que el diseño fue cuasi experimental al tomar la muestra tal como estaba conformada.

El primer grupo estuvo conformado por nueve estudiantes de ambos géneros, entre 18 y 25 años, que implementaron la metodología SCRUM en la asignatura de Programación II de la carrera de Ingeniería en Sistemas de la Información, durante el segundo cuatrimestre del año 2018.

El segundo grupo se conformó por siete estudiantes, de género masculino, entre 22 y 26 años, que habían cursado la asignatura de Programación II en años anteriores al 2018, mediante la implementación de metodologías tradicionales de enseñanza, a cargo de los mismos docentes.

El profesor titular y el profesor adjunto de la catedra de programación II, además de enseñar los conceptos

estipulados, asumieron el rol de clientes para cada uno de los trabajos prácticos asignados.

2.3. Instrumentos

El impacto fue medido mediante la encuesta de Nivel de Impacto de la Implementación de las estrategias de enseñanza y aprendizaje en el aprendizaje percibido de los estudiantes, construida, validada y adaptada por Tumino y Bournissen, [13], compuesta por ítems vinculados a la percepción del aprendizaje (con 8 ítems) y por ítems relacionados con estrategias de aprendizaje (con 10 ítems). La escala utilizada en las dos dimensiones es de tipo Likert desde 1= muy en desacuerdo, 2= en desacuerdo, 3= neutro, 4= de acuerdo y 5= muy de acuerdo.

Para su operacionalización, se trabaja con las medias de las puntuaciones correspondientes a cada una de las dos dimensiones de la escala.

Inicialmente la escala propuesta se confeccionó con 17 ítems, teniendo en cuenta las ideas de Abdala [11], juntamente con modelos propuestos por Riascos-Erazo [14] y Balas-Nakash, Rodríguez-Cano, Muñoz-Manrique, Vásquez-Peña y Perichart-Perera [15]. El borrador de la encuesta así obtenida en su primera versión se envió a doce expertos, quienes fueron invitados a evaluar en primera instancia los ítems y sugerir las modificaciones que consideraran oportunas (modificación, eliminación o inclusión de ítems). Una vez obtenida la nueva versión de la escala, con 22 ítems, en una segunda instancia se solicitó la valoración de la claridad y pertinencia de cada ítem a fin de obtener evidencias de la validación de contenido. Como etapa final se verificó el grado de acuerdo entre los doce expertos que participaron, mediante la V de Aiken como una de las técnicas que permite cuantificar la claridad y pertinencia de cada ítem respecto de un dominio de contenido formulado por jueces. Como resultado se obtuvieron puntuaciones superiores a .80, excepto en dos ítems que fueron eliminados del cuestionario, quedando 20 ítems en la nueva versión (ver Anexo).

Como siguiente etapa del proceso de validación de constructo de la escala de evaluación del impacto en el aprendizaje percibido de los estudiantes, se procedió a aplicarla a una muestra piloto de 122 estudiantes. Con los datos obtenidos se aplicó el Análisis factorial exploratorio, utilizando la rotación Varimax. La medida de adecuación muestral de Kaiser-Meyer-Olkin, fue de .946 y la Prueba de esfericidad de Bartlett estuvo asociado a $p< .05$, lo que evidenció la adecuación de la muestra para el análisis.

El análisis de componentes principales generó un modelo de dos dimensiones que reflejaba una estructura compuesta por 8 ítems vinculados al aprendizaje y por 10 ítems relacionados con estrategias de aprendizaje. Se encontró que dos de los ítems de la escala presentaron complejidad factorial puesto que cargaban con la misma fuerza en los dos factores representantes del modelo, por lo que fueron eliminados de la escala.

El factor de aprendizaje quedó integrado por los ítems 1, 2, 4, 8, 10, 12, 13 y 14. Mientras que el factor estrategias de aprendizaje representa a los ítems 3, 5, 6, 7, 9, 11, 17, 18, 19 y 20.

Se analizó la confiabilidad de los dos factores identificados, obteniendo un coeficiente Alpha de Cronbach superior a .9 en ambos casos, lo que prueba la consistencia interna de las dos subescalas. Se midió el impacto mediante el test de Nivel de Impacto de Tumino y Bournissen [12]. El test presenta una estructura compuesta por 8 variables vinculadas al aprendizaje y 10 variables relacionadas con estrategias de aprendizaje. La escala utilizada en las dos dimensiones es de intervalo desde 1 = muy en desacuerdo, 2 = en desacuerdo, 3 = neutro, 4 = de acuerdo y 5 = muy de acuerdo.

Esta encuesta fue aplicada a los estudiantes que cursaron la asignatura de programación II con la metodología SCRUM como estrategia de aprendizaje y a estudiantes que cursaron con metodologías tradicionales.

Por su parte, la calificación promedio de los estudiantes encuestados de los dos grupos, fue provista por el docente titular de la cátedra desde sistema académico de la universidad.

2.4. Procedimientos para recolección de datos

En primer lugar, se programó una reunión con la dirección de la carrera y el docente titular de la cátedra, a fin de acordar los términos de la intervención a partir del propósito y las características de la asignatura y de la investigación. En dicha reunión se concertó un modelo a seguir que consta de un trabajo experimental en el que se evaluaría el aprendizaje de los estudiantes, con la finalidad de disminuir la tasa de deserción o pérdida de la materia respecto de años anteriores.

El docente sugirió el modelo de armado de equipos, a partir de una evaluación inicial niveladora, a fin de asegurar el beneficio de todos los estudiantes.

Se propuso aplicar la metodología SCRUM, como estrategia de enseñanza y aprendizaje de la programación.

A fin de afianzar el protagonismo del estudiante en el proceso de aprendizaje, se asignarían trabajos prácticos de desarrollo incremental de aplicaciones, haciendo uso de los conceptos que se iban introduciendo.

Durante los meses de cursado se asistió constantemente a las clases prácticas a fin de observar la forma de trabajo utilizado por los estudiantes y registrándolo en una bitácora de observaciones.

Finalizada la cursada se les administró a los estudiantes el test para recolectar los datos de nivel de impacto en su aprendizaje percibido. El mismo test se aplicó también a estudiantes que años anteriores habían cursado la asignatura con otras metodologías, pero con los mismos docentes.

Por último, se programó una reunión con los docentes a fin de conocer sus percepciones en cuanto a la metodología aplicada y se obtuvieron las calificaciones de todos los estudiantes que respondieron el test de nivel de

impacto. Asimismo, se obtuvieron los datos de cursado de las cohortes de los últimos diez años a fin de analizar comparativamente los resultados.

2.5. Procedimientos para el análisis estadístico de los datos

Una vez recolectados los datos, se utilizó el programa IBM SPSS Statistics versión 23.0 para el análisis descriptivo e inferencial de los datos. Dado el tamaño de la muestra y consecuentemente el incumplimiento de los supuestos de normalidad. se procedió a aplicar pruebas no paramétricas mediante las que se compararon los rangos promedio del nivel de impacto en el aprendizaje percibido y de la calificación promedio entre los estudiantes que utilizaron la metodología SRUM y los que usaron metodologías tradicionales anteriormente.

3. Resultados

A continuación, se muestran los resultados obtenidos a partir de las variables presentadas en la investigación.

El objetivo principal fue conocer si los estudiantes que trabajaron con la metodología SCRUM, podrían mostrar un mejor nivel de aprendizaje de la asignatura, lo que se reflejaría tanto en las calificaciones promedio, como en el impacto de la implementación de la metodología en su aprendizaje percibido.

3.1. Datos de cursado

Mediante el análisis realizado con los datos de cursado de la asignatura obtenidos de las últimas diez cohortes, se pudo observar que en el año en que se realizó la experiencia, el 66,67% de los estudiantes promociono la asignatura, mientras que el 33,3% restante la regularizó. De esta manera ningún estudiante perdió o abandonó la asignatura.

De los últimos diez años de cursado de la asignatura, los nueve años anteriores se utilizó una misma metodología y en el último año se implementó la metodología SCRUM. Se observa que tan solo en este último año ningún alumno perdió o abandonó la cursada, tal como se muestra en la Tabla 1.

Tabla 1. Datos de cursado correspondientes a los últimos diez años.

Año	Promoción Directa	Promoción Indirecta	Pierde la Materia	Abandona
2009	30,0%	30,0%	30,0%	10,0%
2010	11,8%	41,2%	35,3%	11,8%
2011	33,3%	40,0%	26,7%	0%
2012	20,8%	37,5%	33,3%	8,3%
2013	10,0%	50,0%	40,0%	0%
2014	29,4%	23,5%	41,2%	5,9%
2015	28,6%	28,6%	42,9%	0,0%
2016	9%	18,2%	54,6%	18,2%
2017	28,5%	52,4%	19,1%	0%
2018	66,7%	33,3%	0%	0%

3.2. Desempeño de los equipos

Los resultados acerca del desempeño de los equipos se pudieron registrar mediante la bitácora de observaciones. En esta bitácora implementada se plasmó el trabajo de cada uno de los equipos durante la cursada de la asignatura de programación II con la metodología SCRUM. Se pudo observar que los equipos mantuvieron un alto desempeño, buena organización, y concluyeron con los proyectos en el tiempo estipulado, en la medida en que cumplieron con las pautas de trabajo de la metodología. También se observó que desarrollar proyectos, dentro de un ámbito de equipo como exige la metodología SCRUM, generó un ambiente de compromiso y potenció la competitividad de los estudiantes en cuanto a los requisitos que debían cumplimentar en cada uno de los proyectos. Otro resultado positivo que arrojó el trabajo con la metodología SCRUM fue que les permitió a los estudiantes, de segundo año de la carrera de Ingeniería en Sistemas, defender los proyectos oralmente y adquirir un lenguaje técnico apropiado.

3.3. Pruebas de hipótesis

A continuación, se presentan los resultados de las pruebas de hipótesis planteadas durante la investigación.

3.3.1 Primera hipótesis

H_1: *Existe diferencia estadística significativa de rangos promedios de rendimiento académico, en términos de calificación promedio obtenido en programación II, entre los estudiantes que utilizan la estrategia de enseñanza con*

metodología SCRUM y los que cursaron con otras metodologías.

Esta hipótesis se probó mediante la prueba no paramétrica U de Mann-Whitney para dos muestras independientes, tomando a la calificación final como variable de comparación y al tipo de metodología como variable de agrupación. En base a esos datos se observó que existe una diferencia estadísticamente significativa de rangos promedios de calificación promedio entre los dos grupos ($z > 1,96$ y $p < 0,05$). Se encontró un mayor rango promedio de calificación promedio en los estudiantes que trabajaron con la metodología SCRUM (RP = 9.93). Mientras que los estudiantes que trabajaron con otras metodologías obtuvieron un menor rango promedio (RP = 5.07).

3.3.2. Segunda hipótesis

H_2: *Existe diferencia estadística significativa en el nivel de impacto en el aprendizaje percibido en programación II entre los estudiantes que utilizan la metodología SCRUM como estrategia de enseñanza y aprendizaje y los que cursaron con otras metodologías.*

Esta hipótesis se probó también mediante la prueba no paramétrica U de Mann-Whitney, tomando el impacto percibido en sus dos dimensiones como variable de comparación y al tipo de metodología como variable de agrupación. Se encontró que existe una diferencia estadísticamente significativa de rangos promedios en la dimensión aprendizaje percibido entre los dos grupos ($z > 1,96$ y $p < 0,05$) (ver Tabla 2). El mayor nivel de impacto estuvo asociado a los estudiantes del grupo experimental (RP = 10.07) quienes utilizaron la metodología SCRUM, mientras que los estudiantes que usaron otras metodologías obtuvieron un menor rango promedio (RP = 4,93).

En cuanto a la dimensión estrategia de aprendizaje del Nivel del Impacto, no se obtuvo una diferencia estadísticamente significativa con un nivel de confianza del 95%, aunque sí con un 90% de confianza ($z > 1,64$ y $p < 0,1$). Con la metodología SCRUM el rango promedio fue mayor (RP = 9.5) que con las otras metodologías (RP = 5.5) como se observa en la Tabla 2.

*Tabla 2.*Rango promedio de aprendizaje y estrategia-aprendizaje.

Categoría	Metodología	Rango promedio
Aprendizaje	SCRUM	10,07
	Otras	4,93
Estrategia-Aprendizaje	SCRUM	9,5
	Otras	5,5

Conclusiones

Mediante el análisis de los datos obtenidos, se pudo observar que existe una diferencia estadísticamente significativa de rango promedio en el aprendizaje entre los dos grupos estudiados. También existió una diferencia significativa de rango promedio de la calificación final de los alumnos que utilizaron la metodología SCRUM, respecto de quienes trabajaron con otras metodologías.

Otro aspecto que se observó fue que en el transcurso de los años 2009 al 2017 existía un predominio de la promoción indirecta, e incluso de la pérdida de la asignatura, mientras que, en el año 2018, con la implementación de la metodología SCRUM, se denotó el predominio de la promoción directa de los estudiantes.

Los estudiantes que tuvieron la posibilidad de trabajar con SCRUM percibieron un mayor impacto en el aprendizaje. Los docentes que implementaron la nueva metodología expresaron entusiasmo con los resultados obtenidos y decidieron mantenerla durante los próximos años.

La valoración del impacto en el aprendizaje percibido, en los estudiantes que experimentaron con las metodologías anteriores, podría deberse a distintos factores, incluidos la falta de comprensión de los temas dados en clase o la falta de compromiso con la asignatura o con los compañeros.

La utilización de la metodología SCRUM en el ámbito académico, y específicamente dentro de las asignaturas de programación, resulta una práctica muy satisfactoria debido a su impacto positivo tanto en el aprendizaje percibido de los estudiantes como en las calificaciones promedio. Los estudiantes lograron asimilar los contenidos, trabajar en forma colaborativa y autoorganizada, administrar los tiempos de entregas, asumir los diferentes roles y exponer satisfactoriamente el trabajo realizado, lo que cultiva el trabajo en equipo, competencia indispensable en el desarrollo de software. La bitácora de observaciones permitió advertir que los estudiantes lograron trabajar apropiadamente, conforme a lo que se espera en cada uno de los sprint, sobreponiéndose a las dificultades que se presentaban a medida que desarrollaban el producto y obteniendo el resultado esperado.

La implementación de la metodología SCRUM facilitó a los docentes el mantenimiento de la dinámica de la clase, consiguiendo que todos los estudiantes pudiesen realizar un seguimiento constante de los contenidos presentados en la asignatura.

La implementación de la metodología SCRUM permite generar un ambiente de trabajo ideal dentro del aula donde los estudiantes pueden aprovechar los aportes proporcionados por los docentes y por sus pares.

Además de los beneficios de la aplicación de SCRUM en la asignatura de programación tanto en el aprendizaje como las calificaciones obtenidas, existe un beneficio aún mayor y es que los estudiantes aprenden a aplicar una metodología de trabajo muy difundida que utilizan en su profesión, puesto que es implementada por una gran cantidad de software factory en todo el mundo como por ejemplo Google y en forma local la empresa Globant. Esto

permite a los estudiantes egresar con la experiencia de trabajos, siguiendo una metodología.

Si bien los resultados dan cuenta de los beneficios de la implementación de SRUM como refuerzo de las estrategias de enseñanza y aprendizaje, se estima relevante profundizar el estudio con muestras de mayor tamaño a fin de poder generalizar los resultados obtenidos. Se recomienda complementar los datos cuantitativos con los cualitativos, a fin de ampliar la valoración de los resultados producidos por la utilización de metodologías de desarrollo ágil como estrategias de enseñanza.

Referencias

[1] E. Costelloe, "Teaching Programming The State of the Art," Department of Computing, Institute of Technology Tallaght, Dublin, Ireland, 2004.

[2] Á. R. Barberis and L. E. Del Moral Sachetti, "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)*, 2016. [Online]. Available: http://sedici.unlp.edu.ar/handle/10915/54603

[3] Perazo C. "El 80% de los estudiantes de carreras informáticas abandona sus estudios." Diario La Nación. http://www.lanacion.com.ar/1632045-el-80-de-los-estudiantes-de-carreras-informatica (accessed Oct. 25, 2013).

[4] X. Albaladejo. "Fundamentos de SCRUM." Proyectos agiles.org. https://proyectosagiles.org/fundamentos-de-SCRUM/ (accessed 2008).

[5] M. Trigás Gallego, *Metodologia SCRUM*. Barcelona, España: Universitat Oberta de Catalunya, 2012.

[6] A. Kuz, M. Falco and 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 e07, 2018.

[7] J. Miller. "5 Elements of Agile Classrooms." Agile Classrooms. http://blog.agileclassrooms.com/2016/11/5-elements-of-agile-classrooms.html (accessed Nov. 1, 2016).

[8] J. V. Chávez Andrade, "Estandarización de los procesos de desarrollo de software utilizando buenas prácticas de programación y SCRUM como marco de trabajo ágil en departamentos de TI," Master's thesis, Facultad de Ingeniería en Sistemas, Electrónica e Industrial, Universidad Técnica de Ambato, Ecuador, 2019.

[9] R. Hervás Lucas, "Método para el aprendizaje de entornos y lenguajes de programación basado en prototipado ágil" in *Jornadas de Enseñanza de la Informática*, Ciudad Real, 2012.

[10] M. A. Aguilar, *El impacto de la carrera de economía de la BUAP en el mercado laboral: la visión de los egresados de la generación 1995-2000*. México: Benemérita Universidad Autónoma de Puebla, 2009.

[11] J. Tejada Ferández and E. Ferrández Lafuente, "La evaluación del impacto de la formación como estrategia de mejora en las organizaciones," *Revista electrónica de investigación educativa*, vol. 9, no. 2, pp. 1-15, 2007.

[12] E. Abdala, *Manual para la evaluación de impacto en programas de formación para jóvenes*. Montevideo: CINTERFOR, 2004.

[13] M. C. Tumino and J. M. Bournissen, "Integration of Information and Communication Technologies (ICT) in the classroom and its impact on students: construction and validation of measurement scales," *Journal of Educational Research and Innovation (IJERI)*, vol.13, 2019 (en prensa).

[14] S. Riascos-Erazo, D. Quintero-Calvache and G. Ávila-Fajardo, "Las TIC en el aula: percepciones de los profesores universitarios," *Educación y Educadores*, vol. 12, no. 3, 2010. [Online]. Available: http://educacionyeducadores.unisabana.edu.co/index.php/eye /article/view/1536/1841

[15] M. Balas-Nakash, A. Rodríguez-Cano, C. Muñoz Manrique, P. Vásquez-Peña and O. Perichart-Perera, "Tres métodos para medir la adherencia a un programa de terapia médica y nutricia en mujeres embarazadas con diabetes y su asociación con el control glucémico," *Revista de Investigación Clínica*, vol. 62, no. 3, pp. 235-243. [Online]. Available: http://www.medigraphic.com/pdfs/revinvcli/nn-2010/nn103g.pdf

Información de contacto de los Autores

Nicolás Timkyw
Universidad Adventista del Plata
25 de Mayo 99
3103 Libertador San Martín,
Entre Ríos, Argentina
nicolas.timkyw@uap.edu.ar

Juan Manuel Bournissen
Universidad Adventista del Plata
25 de Mayo 99
3103 Libertador San Martín,
Entre Ríos, Argentina
juan.bournissen@uap.edu.ar

Marisa Cecilia Tumino
Universidad Adventista del Plata
25 de Mayo 99
3103 Libertador San Martín,
Entre Ríos, Argentina
marisa.tumino@uap.edu.ar

Nicolás Eduardo Tymkiw Vázquez
Lic. en Sistemas de información. Expositor en III congreso sudamericano de investigación de la educación Adventista 2019. En lo siguientes trabajos: Metodología SCRUM: adecuación para la enseñanza y el aprendizaje de la programación; Sistema de documentación de respaldo para equipamiento de centrales de esterilización; Graduado en la Universidad Adventista del Plata (UAP), Entre Ríos, Argentina.

Juan Manuel Bournissen
Doctor en Tecnologías Educativas: E-learning, y Gestión del Conocimiento en la Universidad de Islas Baleares, España. Master en Ingeniería del Software obtenido en la Universidad Politécnica de Madrid. Magíster en Ingeniería del software obtenido en el Instituto Tecnológico de Buenos Aires, ITBA. Especialista en Entornos Virtuales del Aprendizaje. Profesor Universitario en Sistemas de Información. Ingeniero en Sistemas de Información. Analista Universitario en Sistemas. Profesor universitario de grado y posgrado en la modalidad presencial y a distancia. Administrador de plataformas virtuales en instituciones estatales y privadas. Formador de formadores en e-learnig en Argentina y en varios países de Latinoamérica. Director de centro de cómputos. Asesor informático en sistemas de información y en tecnologías educativas y e-lerning.

Marisa Cecilia Tumino
Ingeniera en Recursos Hídricos, Hidrotécnica, Analista en Informática Aplicada, Magister en Ciencias Computacionales, Doctora en Educación con énfasis en Administración educativa. Desempeño profesional en (a) Instituto Francisco Ramos Mejía - Santa Fe (b) Universidad Adventista del Plata, (c) Instituto Juan Bautista Alberdi- Misiones, (d) Universidad de Montemorelos - México, (e) Herbert Fletcher University - Puerto Rico y (f) Universidad Adventista de Chile - Chile. Secretaria de Investigación en FACEA-UAP. Asesora de procesos de autoevaluación y acreditación de carreras en la Secretaría de Evaluación y Calidad Universitaria-UAP.