Una novedosa propuesta de implementación de Metodologías Agile en el proceso de enseñanza-aprendizaje de una asignatura de Ingeniería en Sistemas Informáticos

Fabián Tesei[1*], Matías Cabrera[1*], Daniel Tedini[1], Guillermo Leale[1]
{FabianElio.Tesei, MatiasAlejandro.Cabrera}@Alumnos.uai.edu.ar,
{Daniel.Tedini, Guillermo.Leale}@uai.edu.ar
[1]Centro de Altos Estudios en Tecnología Informática - Universidad Abierta Interamericana
*Los autores contribuyeron en igual medida en el presente trabajo.

Resumen. Durante los últimos años las metodologías Agile aparecieron como reacción a los métodos tradicionales de desarrollo de software, reconociendo además la necesidad de documentación alternativa en el desarrollo. En la actualidad, en la carrera de Ingeniería en Sistemas, se siguen utilizando metodologías tradicionales para la enseñanza de desarrollo de software, creando una brecha entre el campo universitario y campo profesional. Este trabajo presenta una novedosa propuesta de aplicación de metodologías ágiles (Agile) en la enseñanza enfocada en el desarrollo de software. La aplicación de este enfoque es potencialmente viable y provechoso para las asignaturas de desarrollo de software de la carrera de Ingeniería en Sistemas Informáticos. El presente artículo describe la propuesta de implementación de metodologías Agile dentro de una asignatura de este tipo. Asimismo, se realiza un relevamiento inicial en docentes y alumnos acerca de la viabilidad de la aplicación de este tipo de metodologías en otras asignaturas de la carrera. Por último, se lleva a cabo una evaluación de los resultados preliminares de la primera experiencia de la aplicación de la propuesta, con la participación de un docente y un grupo de estudiantes, en el proceso de enseñanza-aprendizaje de una asignatura real.

1 Introducción

En la evaluación del proceso de aprendizaje dentro del marco de la educación formal superior de carreras de Ingeniería en Sistemas de Información en Argentina, se aplica una óptica de análisis crítico persiguiendo la generación de nuevos conocimientos. En dicha área de trabajo es necesario contemplar, además de los contenidos académicos, el campo profesional en el que se aplica. Con lo cual, luego de transitar los años del ciclo básico de la carrera, se dictan asignaturas que incluyen contenidos relativos al desarrollo de software [1]. En estas asignaturas, se propone la aplicación de metodologías para obtener un producto que sea el resultado práctico de la teoría de desarrollo de software vista en la asignatura. Este trabajo se desarrolla en el ámbito de la producción de Trabajos de Cátedra. En estas currículas, se profundiza en la aplicación de la metodología tradicional en cascada conocida como Waterfall [2]. Por su parte, un importante paradigma utilizado en los últimos años es el conjunto de metodologías denominadas ágiles o *Agile*[1]. Las metodologías Agile introducen prácticas que permiten ejecutar un proyecto basándose principalmente en la satisfacción del cliente junto con la habilidad de

[1] Usaremos en este trabajo la denominación en inglés “Agile” por ser la más utilizada en la industria.

responder a los constantes cambios de requerimientos [3]. Dada la amplia aceptación por parte de la comunidad académica y de la industria hacia las metodologías Agile, así como la rápida respuesta frente a los cambios, es importante capacitar a los alumnos en el uso de estas metodologías [4]. Con ello es posible generar un cambio intra-cátedra, al adaptar el desarrollo de las clases a un esquema de trabajo dinámico, a la manera de los requerimientos de la industria; junto con ello, se puede originar un cambio extra-cátedra, preparando a los alumnos desde su formación académica para trabajar en un contexto cambiante, similar al ámbito laboral real. En este trabajo proponemos la aplicación de la metodología Agile a la enseñanza del desarrollo de software en la carrera de Ingeniería en Sistemas Informáticos en una universidad de Argentina. Dicha metodología está siendo aplicada actualmente en un contexto académico real, en dos asignaturas de campo del ámbito académico que iniciaron su año lectivo en abril de 2019. A partir de esta propuesta, presentamos una evaluación sobre su factibilidad de aplicación y su grado de aceptación por la comunidad académica. Además, presentamos una ponderación de herramientas de software acordes a los requerimientos simultáneos de la metodología y de la currícula correspondiente. Finalmente, especificamos las condiciones de aplicación de la metodología y concluimos acerca de la viabilidad y conveniencia de la propuesta.

Este trabajo está organizado de la siguiente manera. En la Sección 2 damos una visión general de la metodología Agile. En la Sección 3 mostramos la propuesta de implementación de la misma. En la Sección 4 presentamos un relevamiento sobre el grado de aceptación y la factibilidad de la propuesta. En la Sección 5 incluimos una evaluación sobre las herramientas software para documentar nuestra propuesta. Finalmente, en la Sección 6 ofrecemos nuestras conclusiones sobre la propuesta.

2 Metodología Agile

La búsqueda de modelos de mejora de desarrollo software tiene una historia de más de 40 años [5]. En al año 2001, Kent Beck se reunió con un equipo de críticos y estudiosos del software para debatir sobre las técnicas y procesos de desarrollo aplicados hasta el momento, intentando reducir la rigidez y crearon el *manifiesto Agile*, un marco teórico de trabajo que permite acortar los tiempos de desarrollo, eliminar la incertidumbre, mejorar la eficiencia en la producción y la calidad de los productos finales, tener capacidad de respuesta al cambio y brindar la mayor satisfacción posible al cliente a través de la entrega temprana y la retroalimentación continua durante la construcción del producto [3]. Agile no marca las pautas acerca de cómo deben hacerse las cosas, sino que sienta las bases de una filosofía sustentada sobre 4 postulados fundamentales.

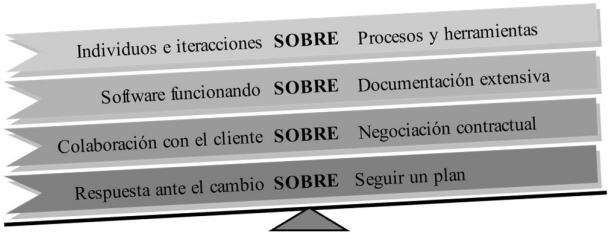

Figura 1. Los 4 postulados fundamentales presentados en el manifiesto Agile.

En la Figura 1 podemos observar los postulados fundamentales del manifiesto, que están presentados en un esquema que asemeja una balanza de cruz. Los conceptos más importantes tienen más peso, o están sobre, los conceptos menos importantes. Cabe destacar que el manifiesto no descarta los términos de la derecha, sino que establece un énfasis en los elementos de la izquierda sin descartar la importancia de los en principio mencionados. Los postulados son los siguientes:

- *Individuos e interacciones sobre procesos y herramientas.* Las personas, sus habilidades y las interacciones entre las personas son fundamentales para el éxito de un proyecto de desarrollo de software, y esos aspectos son comparativamente más importantes que las herramientas y los procesos.
- *Software funcionando sobre documentación extensiva.* Un proyecto existe con el único propósito de entregar software funcionando que satisfaga las necesidades del cliente, no para marcar los entregables de una lista de entregables por el bien de seguir un proceso. Los requisitos deben ser documentados en detalle y la documentación innecesaria debe ser eliminada.
- *Colaboración con el cliente sobre negociación de contratos.* Cuanto más participe un cliente en la entrega, más cerca estará el producto final de las expectativas del cliente y las necesidades de la organización. Además, la colaboración del cliente lo ayuda a hacerle sentir la solución como propia.
- *Responder al cambio sobre seguir el plan.* Responder al cambio es el concepto central y el motor principal de los métodos Agile. Durante un proyecto típico, los requisitos y el grado en que se entiendan esos requisitos cambiarán con el tiempo. Esto es normal. La pregunta clave para un proyecto es cómo manejar esos cambios de manera controlada. La planificación sigue siendo muy importante en los métodos Agile, pero el equipo del proyecto debe estar listo para que el plan cambie. De los 4 valores fundamentales se derivan 12 principios que se pueden ver en el Anexo I. El Anexo I está disponible en https://www.guillermoleale.com/est-02-anexo-i

Existen diversas formas de aplicar metodologías Agile. En oportunidad de acercar la metodología al ámbito académico, el foco estará centrado en *Scrum.* Scrum es un marco de trabajo en el que se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto [6]. Por ello, Scrum está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener prontos resultados, donde los requisitos pueden cambiar o estar poco definidos, y donde la innovación, la competitividad, la flexibilidad y la productividad son de fundamental importancia. Scrum establece los roles, las ceremonias y los artefactos necesarios para una correcta utilización y seguimiento de la metodología. Estos elementos se detallan a continuación.

Roles:

- *Product Owner* (Cliente) representante de todas las personas interesadas (stakeholders) para conseguir una buena definición de los objetivos del producto o proyecto.
- *Scrum Master* (Facilitador) Actúa como facilitador, eliminando impedimentos. Vela por la implementación y cumplimiento de la metodología.
- *Team* (Equipo) conjunto de personas más "técnicas" que de manera conjunta desarrollan el producto del proyecto.

Ceremonias:

- *Sprint Planning* (Planificación de la iteración) es la primer ceremonia de Scrum en dónde se planifican las tareas a realizar en el Sprint en curso.
- *Sprint* (Ejecución de la iteración) es un intervalo de tiempo corto, en donde se desarrolla el incremento de un producto, potencialmente entregable.
- *Scrum Daily Meeting* (Reunión diaria de sincronización del equipo) es una reunión de no más de 15 minutos de duración, donde participa todo el equipo, en el que se sincronizan las actividades que están ocurriendo en el sprint, y la planificación de las actividades de las próximas 24 horas.
- *Sprint Review* (Demostración de los requisitos completados) son sesiones de demostración en los que se busca validar lo que se desarrolló durante el Sprint.
- *Sprint Retrospective* (Retrospectiva) es la última ceremonia en un Sprint. Oportunidad para el equipo de inspeccionarse a sí mismo, y crear un plan de mejora que se pondrá en marcha inmediatamente, en el siguiente Sprint.

Artefactos:

- *Product Backlog Item o PBI* (Requisito) es el elemento (especificaciones, requisitos, historias de usuario) que conforman lo que en Scrum se define como Product Backlog.
- *Product Backlog* (Lista de requisitos priorizados) es la lista priorizada de requisitos, representando la visión y expectativas del cliente respecto a objetivos y entregas del producto o proyecto. La lista parcial correspondiente a un Sprint se denomina *Sprint Backlog.*
- *Burndown Chart* (Gráfico de trabajo) es una forma gráfica de mostrar avance, desvíos y retrasos en cada sprint de las historias comprometidas.

Los precedentes elementos serán utilizados para la implementación de nuestra propuesta.

3 Propuesta de implementación

Este trabajo presenta la propuesta de implementación de un modelo de desarrollo específico dentro del ámbito académico. Este modelo se introduce en el proceso de enseñanza-aprendizaje de una asignatura que incluye el desarrollo de software. Para esta propuesta nos basamos en la difícil tarea que implica llevar adelante el abordaje del ciclo de vida de un proyecto dentro del aula.

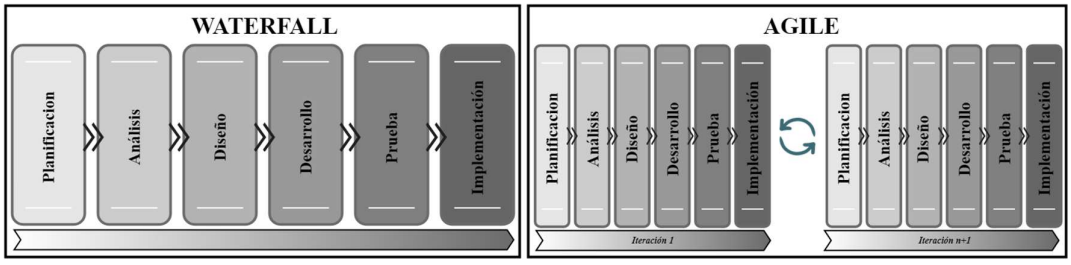

Figura 1. Esquema comparativo de las metodologías Waterfall (izquierda) y Agile (derecha).

Es importante en el planteo de nuestra propuesta tener en cuenta tanto los propios principios de la metodología Agile, así como otro importante modelo utilizado tanto en la industria como en la enseñanza-aprendizaje, y que pretendemos mejorar mediante

esta implementación: el modelo en cascada o *Waterfall* [7], [8]. El modelo Waterfall se basa en un enfoque menos iterativo y flexible, ya que su progreso sigue una dirección lineal ("hacia abajo" como en una cascada) a través de las distintas fases del proceso de desarrollo. El desarrollo de software Agile, en cambio, brinda un enfoque enérgico y expeditivo sobre la solución evolucionando a través del esfuerzo y colaboración de los equipos autoorganizados y multifuncionales. Aboga por la planificación adaptativa, el desarrollo evolutivo, el conocimiento empírico y la mejora continua alentando una respuesta rápida y flexible al cambio. Un esquema comparativo de ambas metodologías puede verse en la Figura 1.

En la actualidad las organizaciones están inmersas en una evolución de desarrollo de software que va de manera tradicional hacia un mundo Agile. Se puede evidenciar este cambio, dado a que la metodología Agile brinda beneficios radicales frente a metodologías tradicionales como Waterfall [9]–[11]. Motivado por este cambio, el presente trabajo busca presentar un nuevo enfoque que aborda una problemática experimentada dentro de un ámbito académico: cómo mejorar el seguimiento del avance del proyecto de desarrollo hasta pasado un tiempo prolongado. Esta mejora está basada en un cambio de metodología en el que se pueda utilizar una forma de trabajo superadora que evite la aplicación de etapas secuenciales no repetitivas en el ciclo de vida del proyecto. En este sentido el modelo Agile propone desde el comienzo del proyecto trabajar en etapas de desarrollo de software de manera simultánea, iterativa e incremental, permitiendo ver avances en forma temprana, adaptándose a los cambios que se presenten de una forma menos disruptiva.

Nuestra propuesta está siendo implementada en un ámbito de clase real, conformado por un docente y 9 alumnos, en la asignatura de Trabajo de Campo, en tercer año de la carrera de Ingeniería en Sistemas Informáticos de la Universidad Abierta Interamericana en la ciudad de Rosario. Proponemos que cada clase de la asignatura esté conformada por dos partes: a) una primera parte consistente en el dictado de un contenido teórico sobre metodología Agile con aplicación de Scrum. Este contenido estará validado por el docente y será dictado en conformidad con el programa de la asignatura; b) una segunda parte compuesta de una práctica que implementa la aplicación de la metodología Agile con Scrum en el contexto educativo sobre un caso de estudio real.

Para esta implementación, el docente propuso como objeto de estudio el desarrollo de un módulo de software dedicado a la seguridad. Dicho módulo tiene como fin gestionar los roles, permisos y usuarios de la solución. De acuerdo con nuestra propuesta, el docente asumirá el rol de Product Owner. Por su parte, los alumnos formarán grupos de 2 personas. Dentro de cada grupo, los alumnos asumirán los roles de Team, en particular en las facetas de Desarrollador y Tester. A su vez, los dos primeros autores asumirán los roles de Scrum Master.

El cursado de la asignatura se encuentra estructurado en Sprints. Dado que la asignatura tiene una duración total de 16 clases con una clase por semana, establecemos que la duración de un sprint es de 4 semanas. Por lo tanto, etiquetaremos, de acuerdo con el uso común en la industria, con el número 0 al primer Sprint (Sprint 0). En este Sprint los Scrum Masters se encargarán de preparar el ambiente y definir conceptos básicos, sin solicitar un entregable en concreto para el Product Owner. De aquí en adelante se comenzará a trabajar con los Sprints etiquetados con los números 1, 2 y 3. En cada uno

de ellos se irán incrementando los requisitos solicitados.

Dado entonces el proyecto general a trabajar en la asignatura, se generará una vinculación entre los temas que el docente transmite en clase y los principios de la metodología Agile. Los requisitos de módulo de seguridad introducidos por el docente son entonces relacionados dentro de nuestra propuesta con los artefactos PBI. Al comienzo de cada Sprint, los equipos deben acordar junto al PO la lista de PBI comprometidos a entregar al concluir el mismo. Los PBI se van a introducir entonces en el Sprint Backlog, de tal forma de contar con una documentación efectiva para cada Sprint. Entonces, los alumnos tendrán 4 clases semanales para realizar la entrega y cerrar un Sprint. Para ello semanalmente se deberá presentar un avance que deberá quedar registrado en una herramienta software de seguimiento del proyecto. Con esto se mostrará el grado de avance de desarrollo de cada PBI.

En cada clase se realizarán simulaciones con los alumnos de las ceremonias de Agile como la Daily meeting donde los alumnos se expresarán dando respuesta a tres interrogantes: a) ¿Cuál es la PBI que se estuvo trabajando dentro del Sprint?; b) ¿Con qué PBI se va a continuar trabajando?; c) ¿Existe algún inconveniente o bloqueo que impida seguir con el desarrollo de los PBI del Sprint?. Esta reunión no debe tomar más de 15 minutos. En cada Daily meeting, se debe completar el progreso de cada tarea, dejando de forma visible el gráfico de Burndown, el cual permitirá mostrar a simple vista si el trabajo del equipo se comporta de manera acorde a la planificación inicial. Al final de cada Sprint se realizarán las ceremonias de Review meeting y Retrospective meeting, donde se realizarán revisiones generales sobre el desempeño del equipo en el mismo.

Los alumnos trabajan en las historias de usuario de cada Sprint según se haya acordado con el Product Owner. Esto quiere decir cada equipo avanzará en el desarrollo de las historias de usuarios según el cronograma planificado. Al final del Sprint se evaluarán los ítems pendientes en cada historia, llamados *puntos de historia.* Si en esta evaluación no queda ningún punto de historia pendiente, el equipo se encontrará al día. El proyecto finaliza cuando el Product Owner no tiene más requerimientos para trabajar con el equipo. Análogamente a la realidad de la industria, en la que existen fechas acordadas con el cliente, en nuestra propuesta los Sprints e historias de usuarios están calendarizados de tal manera de que los equipos puedan cerrarlos adecuadamente al final del cursado.

4 Evaluación de aceptación y factibilidad de la propuesta

Realizamos para este trabajo una evaluación acerca del conocimiento de las metodologías Agile y el grado de aceptación de nuestra propuesta, con el objetivo de obtener indicadores acerca de su factibilidad de aplicación. Para esto, se procedió a hacer un relevamiento inicial dentro del ámbito universitario. Se invitó a los alumnos de las asignaturas Trabajo de Campo y Trabajo de Diploma turno mañana de tercer año junto con el plantel docente de la carrera de Ingeniería en Sistemas Informáticos de la Universidad Abierta Interamericana, Sede Regional Rosario, a completar una encuesta sobre el conocimiento general de metodologías Agile con aplicación de Scrum. De ese modo se generaron dos encuestas, una enfocada a la población docente con la recolección de 10 respuestas y otra enfocada a los alumnos recabando 11 respuestas. Las encuestas están

disponibles en el Anexo II, en https://www.guillermoleale.com/est-02-anexo-ii. El público de la población docente relevado es mixto de género, se encuentra en un rango etario de 35 a 44 años y es en su mayoría oriundo de la ciudad de Rosario. En el ámbito laboral se encuentran mayoritariamente trabajando con un empleo de tiempo completo bajo relación de dependencia y la minoría trabajando por cuenta propia. Los mismos se encuentran ejerciendo la docencia en la Universidad Abierta Interamericana, así como en la Universidad Tecnológica Nacional y la Universidad Nacional de Rosario, dictando materias del ciclo básico y otras enfocadas al desarrollo de software y gestión. Por otra parte, el público de la población de alumnos relevado es mixto de género siendo en su mayoría masculina, se encuentran en un rango etario de 18 a 24 años, oriundos de la ciudad de Rosario. En el ámbito laboral se encuentran en su mayoría como estudiantes y un 30% de los alumnos realizan trabajos de medio tiempo. Manifiestan no tener conocimiento sobre la metodología, habiéndola escuchado por primera vez en el ámbito académico. Dicho esto, se refleja en los resultados que los encuestados nunca utilizaron la metodología, con lo cual desestimamos las preguntas referidas al conocimiento de las herramientas de desarrollo de proyectos Agile. A continuación, se analizan en profundidad los aspectos más representativos obtenidos en la encuesta.

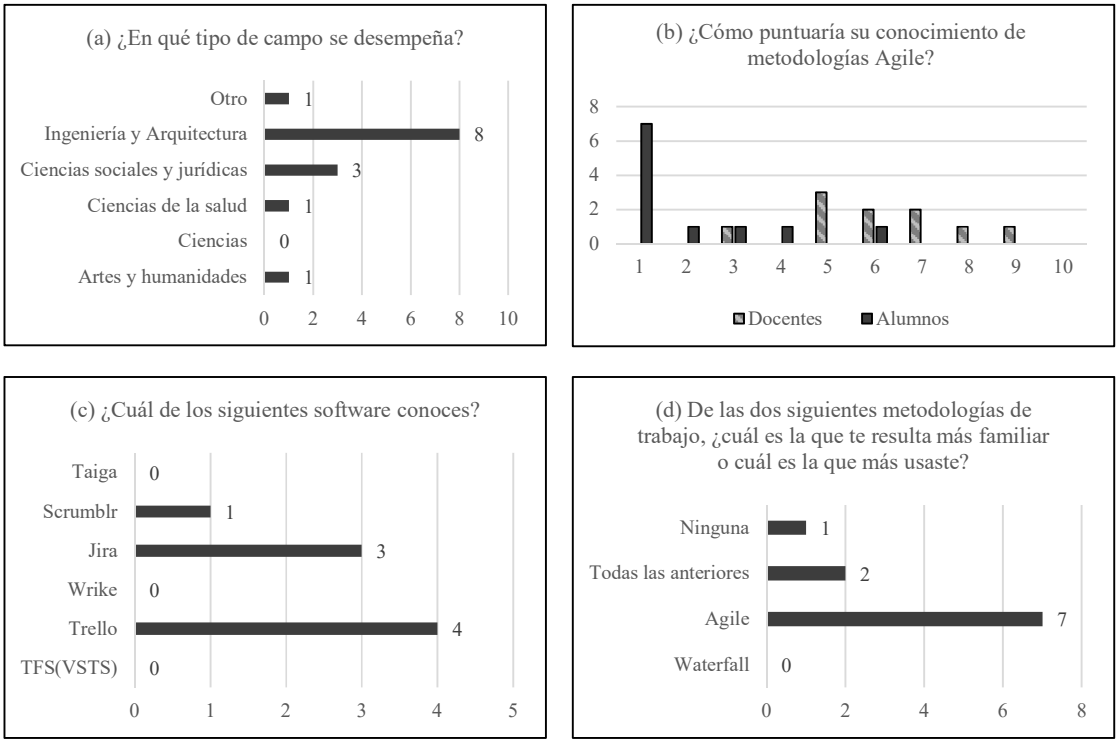

Figura 2. (a) Tipo de campo que se desempeña el plantel docente ejerciendo la docencia. (b) Puntuación general del conocimiento Agile. (c) Conocimiento de software de desarrollo Agile (encuesta Docentes) (d) Conocimiento en metodología de trabajo.

En la figura 2(a), podemos observar que la población docente se encuentra afectada en las ramas de la Ingeniería y Arquitectura en su gran mayoría, sólo el 20% se encuentra en las Ciencias Sociales, Humanidades y Salud. En la Figura 2(b), podemos observar

la puntuación general sobre la población docente y de alumnos sobre el conocimiento que consideran tener sobre metodología Agile, presentados en un esquema de gráfico de barra. En un 30% los docentes se identificaron con un conocimiento medio de la metodología mientras que la mayor parte del plantel docente considera tener un conocimiento alcanzado apenas mayor de la media. En contraposición a los docentes, los alumnos manifiestan tener un conocimiento muy bajo de metodologías, encontrándose en un área de desconocimiento para ellos. En la Figura 2(c), se puede observar que el 50% del plantel docente conoce Trello como herramienta de gestión del desarrollo de software Agile, y en menor porcentaje se encuentra Jira y Scrumblr. Cabe destacar que para el grupo de alumnos, esta pregunta carece de importancia en la respuesta, ya que al no tener conocimientos de la metodología, es de esperar que no conozcan herramientas de software para gestionarla. En la Figura 2(d) se puede observar que el 70% de los docentes indica tener un conocimiento familiar de Agile por sobre Waterfall. Cabe mencionar que los alumnos alegan no reconocer ninguna de las metodologías. Como punto relevante de las encuestas realizadas, se evidenció la predisposición a recibir una capacitación de manera unánime en ambas poblaciones acorde a metodologías Agile aplicando Scrum como marco de trabajo para la implementación de la misma.

5 Estudio de Software

Dada la necesidad de contar con una herramienta software para llevar registro de todas las actividades relacionadas con nuestra propuesta, incorporamos en este trabajo un relevamiento sobre las principales herramientas utilizadas por la metodología. De dicho relevamiento realizamos una comparación entre seis de ellas, a fin de establecer una ponderación acerca de cuál es la más conveniente para acompañar la aplicación de la propuesta. Cabe aclarar que para cada herramienta se tienen en cuenta únicamente las funcionalidades que pueden utilizarse en la aplicación de metodologías ágiles, dejando fuera del análisis el resto de las características de cada una. A continuación, se detallan las principales ventajas y desventajas de las herramientas. Cabe destacar que todas las seleccionadas, tienen la modalidad de trabajo on-line (en línea) es decir, que se pueden administrar de manera remota a través de internet.

Herramienta	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
Trello	Si	No	No	Si	No	No	No	No	No	No
Taiga	Si	Si	Si	Si	Si	Si	Si	Si	Si	Si
TFS	Si	Si	Si	Si	Si	Si	Si	Si	Si	No
Wrike	Si	Si	Si	Si	No	Si	No	No	No	No
Jira	Si	Si	Si	Si	Si	Si	Si	Si	Si	No
Scrumblr	Si	No	No	Si	No	No	No	No	No	No

Tabla 1. Evaluación de herramientas software para desarrollar proyectos Agile.

En la Tabla 1 se muestran los resultados de la evaluación. Cada una de las columnas corresponde a una característica evaluada. Las características correspondientes a cada número son detalladas a continuación.

- (1) Manejo de Backlog: se corresponde a poder incorporar una lista general de requisitos en la herramienta, disponible para en un futuro incluirla dentro de un Sprint.
- (2) Manejo de Sprint: es la capacidad de trabajar con una lista de requerimientos priorizados exclusiva por sprints, cuyo origen es el Backlog.
- (3) Duración de un Sprint: es la posibilidad de definir el periodo de tiempo que se define que dura una iteración de desarrollo de software

- (4) Manejo de historias de usuario: es la capacidad de poder incorporar los requerimientos dentro de la herramienta
- (5) Estimación de historias de usuario: se trata de dimensionar los requerimientos en puntos de usuario, para entender la complejidad, esfuerzo y tamaño del mismo.
- (6) Manejo de Tareas: corresponde a poder incorporar pequeñas actividades a realizar dentro de un requerimiento.
- (7) Estimación de Tareas: estimar la complejidad, tamaño y esfuerzo de una tarea
- (8) Manejo Bugs: incorpora la posibilidad agregar *Bugs* (errores o fallos) detectados en el producto final, para su futura corrección.
- (9) Estimación Bugs: Estimar la complejidad, tamaño y esfuerzo de un fallo
- (10) Código Abierto: Se refiere a la posibilidad de tener acceso al código fuente de la solución.

De acuerdo con los criterios a analizar para cada una de las herramientas obtuvimos las siguientes conclusiones. Dado que se implementará Scrum como marco de trabajo, debemos considerar algunas características como críticas para una aplicación efectiva de la metodología. En primer lugar, es de vital importancia que la herramienta a seleccionar permita la gestión y uso de los artefactos y roles. Por otro lado, la herramienta debe permitir gestionar un Backlog general. Es necesario poder crear nuevos Sprints y a su vez tener un Backlog priorizado, organizado por Sprint. Deben poder medirse las historias de usuario y dar la posibilidad de ver gráficamente el avance del Sprint en un Burndown Chart. Por lo anteriormente expuesto procedimos a seleccionar *Taiga*[2] como herramienta software dado que cumple con los requisitos críticos planteados, además de posibilitar el seguimiento detallado del proyecto de manera colaborativa. Finalmente, es importante destacar que Taiga es de código abierto y provee una opción gratuita perfectamente adecuada a la implementación de nuestra propuesta.

6 Conclusiones

Para concluir luego de plantear la propuesta y relevamiento realizado, el principal objetivo es introducir metodologías Agile dentro del ámbito educativo en las Asignaturas de desarrollo de software de la carrera de Ingeniería en Sistemas Informáticos de la Universidad Abierta Interamericana Sede Regional Rosario. Se ofrece un escenario adecuado y propicio para la aplicación de la propuesta práctica a llevar a cabo, teniendo en cuenta que la población tanto docente como estudiantil no conoce de las metodologías Agile. Adicionalmente, esta propuesta se plantea como alternativa a la metodología Waterfall, que venía siendo utilizada en años anteriores de acuerdo con la planificación curricular. Los resultados de esta metodología quedan fuera del análisis de nuestra propuesta, dada la imposibilidad de recopilar datos comparativos de desempeño específico de la misma aplicada a la asignatura. Con esto, creemos conveniente remarcar que nuestra propuesta implica la primera experiencia de aplicación de metodologías agiles en la asignatura, y por lo tanto pretende sentar un precedente en la recolección de datos de desempeño de acuerdo con la metodología planteada. De esta forma, buscamos construir una cultura de recolección de datos y evaluación de metodologías de tal forma de contribuir a la mejora continua en el proceso de enseñanza-aprendizaje de la asignatura.

Al momento de la presentación de este trabajo, se comenzó con la implementación

[2] Taiga: https://taiga.io/. Último acceso mayo 2019.

de nuestra propuesta a fines de poder presentar sus respectivos resultados en otro trabajo separado. Se persigue como objetivo principal que el alumno adquiera los conocimientos y nociones básicas de las metodologías Agile como así también incorpore un nuevo paradigma en el aprendizaje en el desarrollo de software aplicando principios y estándares de la misma. La aplicación de la propuesta favorecerá a la formación docente y estudiantil en el conocimiento de una temática que se encuentra como requisito en un mercado laboral cambiante como el de la actualidad. Al momento de la presentación del presente documento, la implementación se encuentra en curso. La misma se está llevando a cabo durante las clases de la asignatura con el cronograma detallado en la propuesta.

Referencias

[1] I. G. Pérez, L. E. Guaycochea, and M. A. Wersocky, "Enseñanza de géneros profesionales y construcción de la identidad disciplinar en informática," in *Simposio Argentino de Enseñanza Superior en Informática (SAESI)-JAIIO 47 (CABA, 2018)*, 2018.

[2] K. Petersen, C. Wohlin, and D. Baca, "The waterfall model in large-scale development," in *International Conference on Product-Focused Software Process Improvement*, 2009, pp. 386–400.

[3] K. Beck *et al.*, "Manifesto for Agile Software Development," *The Agile Alliance*, 2001. [Online]. Available: http://agilemanifesto.org/.

[4] M. Huo, J. Verner, L. Zhu, and M. A. Babar, "Software quality and agile methods," in *Proceedings of the 28th Annual International Computer Software and Applications Conference, 2004. COMPSAC 2004.*, 2004, pp. 520–525.

[5] O.-J. Dahl, E. W. Dijkstra, and C. A. R. Hoare, *Structured programming*. Academic Press Ltd., 1972.

[6] K. Schwaber and M. Beedle, *Agile software development with Scrum*, vol. 1. Prentice Hall Upper Saddle River, 2002.

[7] A. Rusu and M. Swenson, "An industry-academia team-teaching case study for software engineering capstone courses," in *2008 38th Annual Frontiers in Education Conference*, 2008, pp. F4C-18.

[8] A. Baker, E. O. Navarro, and A. Van Der Hoek, "An experimental card game for teaching software engineering processes," *J. Syst. Softw.*, vol. 75, no. 1–2, pp. 3–16, 2005.

[9] K. Sureshchandra and J. Shrinivasavadhani, "Moving from waterfall to agile," in *Agile 2008 conference*, 2008, pp. 97–101.

[10] M. Sumrell, "From waterfall to agile-how does a QA team transition," in *Proceedings of the Agile Conference (AGILE)*, 2007, pp. 291–295.

[11] S. Balaji and M. S. Murugaiyan, "Waterfall vs. V-Model vs. Agile: A comparative study on SDLC," *Int. J. Inf. Technol. Bus. Manag.*, vol. 2, no. 1, pp. 26–30, 2012.