Artículo **Revista de Tecnologías de la Información**
Diciembre, 2018 Vol.5 No.17 17-26

Desarrollo del sitio web Sisconve con la metodología Scrum

Development of the Sisconve website with the Scrum methodology

MEX-ALVAREZ, Diana Concepción†*, HERNANDEZ-CRUZ, Luz María, CAB-CHAN, José Ramón y ROMERO-HERNÁNDEZ, Oscar Fabián

Universidad Autónoma de Campeche Camus V, Unidad Habitacional Siglo XXIII por Avenida Ing. Humberto Lanz Cárdenas, Kalá, 24085 Campeche, Camp.

ID 1er Autor: *Diana Concepción, Mex-Alvarez* / **ORC ID:** 0000-0001-9419-7868, **Researcher ID Thomson**: I-4164-2018, **CVU CONACYT-ID**: 842039

ID 1er Coautor: *Luz María, Hernández-Cruz* / **ORC ID:** 0000-0002-0469-5298, **Researcher ID Thomson**: H-3153-2018, **CVU CONACYT-ID**: 662220

ID 2do Coautor: *José Ramón, Cab-Chan* / **ORC ID:** 0000-0003-1043-629X, **Researcher ID Thomson**: I-5425-2018, **CVU CONACYT-ID**: 204250

ID 3er Coautor: *Oscar Fabián, Romero-Hernández* / **ORC ID:** 0000-0001-9974-6635, **Researcher ID Thomson**: I-4171-2018, **CVU CONACYT-ID**: 902799

Recibido: 22 de Octubre, 2018; Aceptado 13 de Diciembre, 2018

Resumen

El presente trabajo propone el desarrollo de un sitio web que permite el seguimiento de los productos académicos, derivados de los convenios de vinculación de la Universidad Autónoma de Campeche con otras instituciones. El sitio web, genera indicadores que miden el impacto de los convenios con los diversos programas educativos. Se realizó un análisis sobre la administración del proyecto de software surgiendo como mejor alternativa el desarrollo de la metodología ágil scrum, debido a las necesidades de cambio en todos sus procesos de desarrollo y etapas de la planificación. Nuestro proyecto plantea las tres fases de scrum, definiendo los objetivos generales y el diseño de la arquitectura de software; siguiendo una serie de ciclos sprint para el desarrollo incremental del sistema, concluyendo con la documentación requerida, así como la retroalimentación del equipo de trabajo sobre las lecciones aprendidas. Se emplearon los principales artefactos scrum: historias de usuario, pilas de producto, lista de tareas, realizándose las estimaciones necesarias, calculando el factor de dedicación con base a la velocidad estimada de nuestro equipo de trabajo.

Desarrollo de software, Metodología ágil Scrum, Sprint, Sitio web

Abstract

The hereby work proposes the development of a web site that allows the tracking of the academic products, wich are a result from the bonding agreements of the Universidad Autonoma de Campeche with other institutions. The web site, generates markers that measure the impact of the agreements with the various educational programs. An analysis about the administration of the software project was carried out emerging as a better alternative the development of the scrum agile methodology, due to the needs of change in all its development processes and stages of planning. Our project poses the three scrum phases, defining the general target and the software architecture design; following a series of sprint cycles for the incremental development of the system, concluding with the required information, as well as feedback from the work team on the lessons learned. The main scrum devices were used: user records, product stacks, task list, making the necessary estimates, calculating the dedication factor based on the estimated speed of our work team.

Scrum Agile Methodology, Software development, Sprint, Website

Citación: MEX-ALVAREZ, Diana Concepción, HERNANDEZ-CRUZ, Luz María, CAB-CHAN, José Ramón y ROMERO-HERNÁNDEZ, Oscar Fabián. Desarrollo del sitio web Sisconve con la metodología Scrum. Revista de Tecnologías de la Información. 2018. 5-17: 17-26

*Correspondencia del autor (correo electrónico: diancmex@uacam.mx)
† Investigador contribuyendo como primer autor.

Introducción

En la Universidad Autónoma de Campeche, actualmente no existe un sistema de información que realice el seguimiento a los productos emanados de convenios celebrados con diversas instituciones, ésto reduce que se promuevan mecanismos que impulsen los beneficios de los mismos y a los programas educativos. Al no existir un monitoreo de los convenios, surgen las siguientes problemáticas:

– Al celebrarse convenios con instituciones interesadas en generar trabajos conjuntos o proyectos de investigación sin que se concreten, se convierten sólo en una "carta de intención" que termina no impactando favorablemente a ningún programa educativo.
– Por otro lado, hay convenios que generan productos académicos de gran valía e impacto para indicadores de diversos programas educativos, sin que exista un registro electrónico, que facilite su acceso.
– El seguimiento de los convenios se realiza de manera manual y documental, lo que representa un cúmulo de trabajo y papeles que hacen complejo el monitoreo y cruzamiento de información.

Para solucionar las problemáticas anteriores, se propone el desarrollo de una aplicación web para automatizar y optimizar el proceso de seguimiento de indicadores, facilitará el monitoreo y cruce de información, así como la generación expedita de estadísticas que aporte información para la toma de decisiones. El presente trabajo establece el uso de una metodología de desarrollo de software para la elaboración de la aplicación web llamada SISCONVE proporcionando métodos, herramientas y técnicas a utilizar.

Metodología

Metodologías de desarrollo web

La estructura en la elaboración de gestión de proyectos dentro de distintas organizaciones en las últimas décadas se ha vuelto más compleja, han surgido nuevas herramientas en las tecnologías resultando cada vez más complejo obtener resultados favorables en el acceso a la información.

Al mismo tiempo, la comunicación y tecnología avanzan a una velocidad considerable, ocasionando una aceleración en la gestión de proyectos, exigiendo la optimización en los trabajos y periodos de tiempo. Lo anterior, promueve el empleo de metodologías para organizar, estructurar y estandarizar la manera de trabajo, obteniendo diferentes herramientas para estimar de forma correcta los tiempos de ejecución, mejorar la relación y habilidades del equipo, reduciendo riesgos posibles durante su desarrollo. En la figura 1 se ilustra el proceso de mejora continua de la metodología Scrum (Pabón, 2016)

Figura 1 Ilustración del uso de una metodología para la elaboración de un proyecto
Fuente: Elaboración Propia

Existen dos esquemas metodológicos para el desarrollo del software: Las metodologías ágiles y las tradicionales.

Las metodologías tradicionales, exigen la planificación total del trabajo, con todas las especificaciones y detalles para iniciar el ciclo de desarrollo de software. En distintas empresas enfatizan la implementación de diferentes procedimientos para la entrega de productos de calidad con costes y tiempos pactados, las metodologías tradicionales ya no bastan para cumplir con el cometido, debido a que no son adaptables para las expectativas del usuario.

Por otro lado, las metodologías ágiles tienen como filosofía priorizar la satisfacción al cliente mediante la entrega temprana y continua de software con valor, disponiendo de diversos principios del manifiesto ágil como la rapidez en la gestión de cambios, mejorando la calidad del desarrollo de software, producto de la adaptación aprovechada de los procesos ágiles para proporcionar ventaja competitiva al cliente entregando software funcional de manera frecuente. (Martel, 2014).

ISSN: 2410-4000

En la tabla 1 se aprecia la comparativa entre las metodologías ágiles y las tradicionales.

Metodologías Ágiles	Metodologías Tradicionales
Basadas en heurísticas provenientes de prácticas de producción de código	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo
Especialmente preparados para cambios durante el proyecto	Cierta resistencia a los cambios
Impuestas internamente (por el equipo)	Impuestas externamente
Proceso menos controlado, con pocos principios	Proceso mucho más controlado, con numerosas políticas/normas
No existe contrato tradicional o al menos es bastante flexible	Existe un contrato prefijado
El cliente es parte del equipo de desarrollo	El cliente interactúa con el equipo de desarrollo mediante reuniones
Grupos pequeños (<10 integrantes) y trabajando en el mismo sitio	Grupos grandes y posiblemente distribuidos
Pocos artefactos	Más artefactos
Pocos roles	Más roles
Menos énfasis en la arquitectura del software	La arquitectura del software es esencial y se expresa mediante modelos

Tabla 1 Diferencias entre metodologías ágiles y no ágiles
Fuente: Letelier & Penadés, 2012

Actualmente se emplean cuatro diferentes metodologías de desarrollo ágil para la gestión de proyectos:

– Extreme Programming XP
– SCRUM
– Kanban
– Agile Inception

Se eligió para el desarrollo de nuestro software, aplicar la metodología ágil Scrum, debido a que propicia la adaptación, flexibilidad e inmediatez para ajustar el proyecto a las circunstancias del entorno. (Macías Brambila, López Laguna, González Del Castillo, & Tolosa Carillo, 2017)

Metodología ágil Scrum

La metodología ágil scrum es un proceso para minimizar los riesgos durante la realización colectiva del proyecto.

El presente trabajo, expone como principales ventajas de la metodología: la productividad, la calidad y el seguimiento diario de los avances del proyecto, que genera una relación de comunicación positiva en los integrantes del equipo. Se presenta IceScrum como plataforma tecnológica de manejo y gestión de proyectos. IceScrum ofrece un conjunto de herramientas al equipo, que integran digitalmente los artefactos scrum y la gestión de eventos; su mayor ventaja respecto a otras plataformas son los análisis, indicadores y gráficos que se generan a partir de la información almacenada sobre el desarrollo. (Guerrero Hernádez, Rosas Cabrera, Cañete Satibañes, & Cedillo Baños, 2017).

En la metodología Scrum ***el equipo Scrum (Scrum Team)*** es multifuncional, con diversas competencias que permiten llevar a cabo el desarrollo del software y se organizan de manera autónoma (Schwaber & Sutherland, 2013). El equipo se conforma por el dueño del producto, el equipo de desarrollo y el líder scrum. Para el desarrollo del presente trabajo, se integró el Equipo Scrum con cuatro personas, tres de ellos alumnos del sexto semestre de la licenciatura de Ingeniería en Sistemas Computacionales de la Universidad Autónoma de Campeche y un profesor de la misma licenciatura quién es responsable del seguimiento de los convenios.

El profesor fungió como ***Dueño del Producto (Product Owner),*** siendo responsable de maximizar el valor del producto, definiendo en la Lista de Producto (Product Backlog) que es lo que tiene que hacer el sistema y cuando hacerlas.

El alumno con conocimientos de ingeniería de software fungió como ***Líder Scrum (Scrum Máster)*** siendo el responsable de solucionar los problemas y obstáculos presentados durante el desarrollo del proyecto liderando al equipo de trabajo.

Los dos alumnos con los conocimientos técnicos necesarios para el desarrollo de software en tópicos de programación y bases de datos conformaron el ***Equipo de Desarrollo (Development Team),*** llevando a cabo las historias comprometidas al inicio de cada sprint (Softeng, 2015).

ISSN: 2410-4000

Requerimientos de desarrollo de Software

Comprender la naturaleza de los problemas puede resultar difícil, especialmente si son nuevos, tras tener conocimiento de ésto, se establecieron los requerimientos para el desarrollo del software fomentando el proceso de analizar, descubrir, documentar y verificar estos servicios en un sistema propuesto, esbozando su comportamiento externo, dividiendo sus características en funcionales y no funcionales. (Sommerville, Sawyer, & Viller, 1999). Se determinaron los dos tipos de requerimientos con relación a nuestro proyecto:

Funcionales: El Sitio web SISCONVE es capaz de generar múltiples indicadores a fin de realizar el seguimiento de los productos resultantes de la relación de convenios con otras facultades de forma automatizada y optimizable. Para nuestro sistema se proponen siete atributos de calidad como requerimientos **No funcionales:** (Bass, Clements, & Kazman, 2012).

- ***Confidencialidad,*** protegiendo accesos no autorizados y divulgación de información.
- ***Integridad,*** protegiendo la información de corrupción e inconsistencias.
- ***Disponibilidad***, garantizando el acceso a la información a los usuarios autorizados.
- ***Escalabilidad,*** adaptándose a las necesidades de los usuarios, prestando servicio adecuadamente, empleando un crecimiento continuo y fluido sin perder calidad en los servicios ofrecidos.
- ***Usabilidad,*** siendo intuitivo, utilizando una interfaz sencilla y atractiva para el usuario.
- ***Soportabilidad,*** facilitando el acceso a la información sobre el manejo, mantenimiento y actualización del software
- ***Compatibilidad***, siendo operable en todos los sistemas operativos del mercado, así como en los navegadores existentes.

Historias de usuario

Las historias de usuario funcionan para identificar, definir y planificar las especificaciones de requisitos en el desarrollo de un proyecto

Se consideran instrumentos para el levantamiento de requerimientos de un software que ha surgido con la aparición de los nuevos marcos de trabajo de desarrollo ágil.

Las historias de usuario abarcan una serie de propiedades concretas que deben cumplir para su ejecución, estas propiedades se les conoce con las siglas INVEST según Bill Wake en su escrito Programming Explored and Refactoring. (Cohn, 2004)

- Independent
- Negotiable
- Valuable to users or customers
- Estimatable
- Small
- Testable

Icescrum permite crear historias de usuario con cada una una de estas propiedades, como se muestra en la Figura 2.

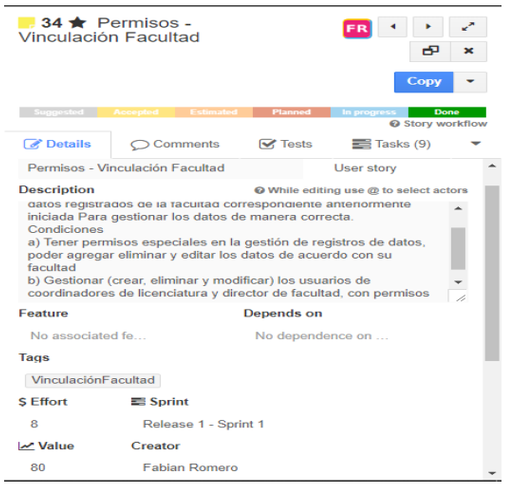

Figura 2 Propiedades de una historia de usuario
Fuente*: Elaboración Propia*

Independientes (Independent)

Es aconsejable evitar las dependencias unas de otras, integra más complejidad en la planificación de la producción de tareas definiendo una priorización en la realización de las mismas, cada una debe corresponder con una única funcionalidad que pueda salir a producción.

Deben establecer su atomicidad, si el alcance es demasiado pequeño, es necesario agrupar con otras historias, o si es grande, las historias deben dividirse.

ISSN: 2410-4000

Negociable (Negotiable)

Las historias de usuarios son una breve descripción de la funcionalidad, consideradas como un recordatorio, permiten cambiar los requisitos establecidos por el cliente, suelen retrasarse los detalles de la historia de usuario debido a evitar cambios de último momento.

Se recomienda apuntar detalles que generan duda o incertidumbres al comprobarlas directamente, éstas notas ayudan en futuras entrevistas con el usuario

La tarjeta manifiesta los elementos más importantes expresando el valor que se desea conseguir desde el punto de vista del usuario.

Valoración (Valuable to users or customers)

Todos aquellos requerimientos obtenidos de un cliente son redactados en un área de ensayo como propuesta de historia de usuario.

Al sugerir cada historia de usuario es importante asignar un valor ponderando la importancia del desempeño en beneficio de implementar una funcionalidad.

Las historias de usuario deben aportar valor, en general definiéndose en base a las peticiones del usuario, no obstante, existe la posibilidad de definir historias de usuario a partir de criterios técnicos por parte del equipo.

IceScrum concede poder determinar el valor en una escala del uno al cien la utilidad del funcionamiento como se muestra en la figura 3. En el proceso de valoración, el dueño del producto estableció una escala tipo Likert, para determinar del grado de importancia del sistema con cinco respuestas como se muestra en la tabla 2:

Ítem	Muy importante	Importante	Moderadamente importante	De poca importancia	Sin importancia
Valoración sistema	76-100	51-75	26-50	1-25	0

Tabla 2 Escala de grado de importancia
Fuente: Elaboración Propia.

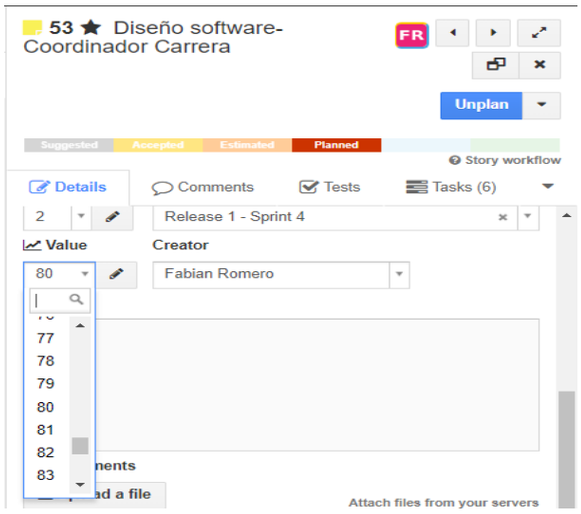

Figura 3 Asignación de valor ponderando el desempeño
Fuente*: Elaboración Propia.*

De 5 grados de importancia establecidos y 69 historias de usuario, se observa de manera gráfica que 81% de se catalogaron como moderadamente importantes, 10% como importantes, 9% muy importantes; se destaca que el dueño del producto no clasificó niguna historia como de poca importancia, ni sin importancia. (Gráfico 1)

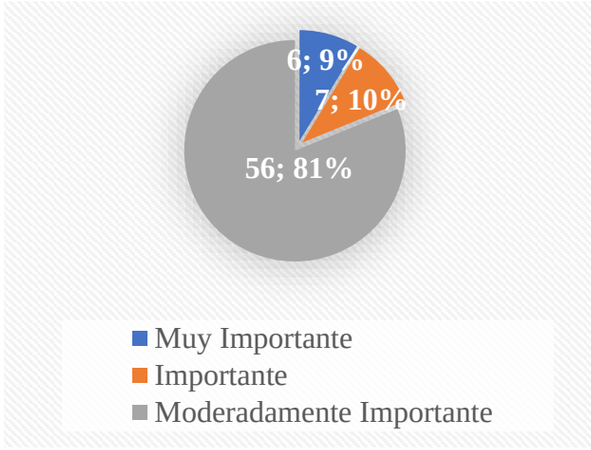

Gráfico 1 Porcentaje total de valores de historias de usuario en gráfica de pastel
Fuente: Elaboración Propia

Después de definir la historia de usuario, procede a ser aceptada si el propietario del producto lo reconoce como una idea valiosa para el proyecto, seguidamente todas las historias aceptadas se trasladan a la pila del producto (Product Backlog).

Las historias son ordenadas y priorizadas por el propietario del producto de acuerdo con sus necesidades y complejidad de cada actividad. (May, Morales, Marrufo, Martín,2013).

ISSN: 2410-4000

Estimable (Estimatable)

El equipo debe estimar el coste aproximado de desarrollo de cada historia, es un requisito fundamental para planificar de manera razonable el trabajo de un sprint generalmente a través de una sesión de planning poker, una dinámica donde se reúne al equipo con una baraja de cartas que ayudarán a estimar el desempeño de desarrollo en una serie de rondas donde el cliente explica un objetivo mientras los miembros de forma individual, ponen sobre la mesa la carta que representa el esfuerzo para dicha historia donde cada número está basado en la secuencia de fibonacci. (Pichler, 2010).

La serie de fibonacci representa la indecisión de efectuar estimaciones sobre hitos de gran duración o esfuerzo.

De acuerdo con el crecimiento de la secuencia, existen grandes diferencias entre los números, es decir, cuanta mayor duda exista sobre un desarrollo, la estimación elegida será más realista.

La plataforma de icescrum ofrece configurar los esfuerzos de tres maneras: con números enteros secuenciales, serie de fibonacci ó con una configuración personalizada como se muestra en la figura 4.

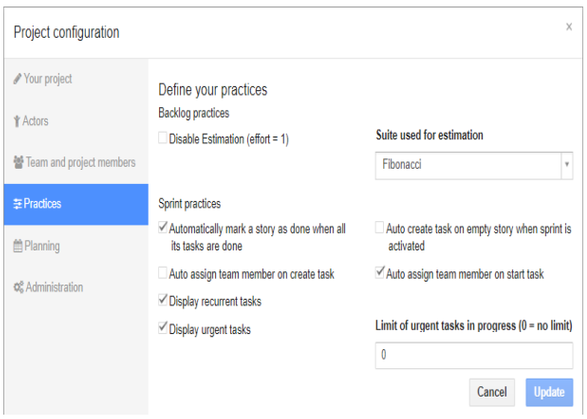

Figura 4 Selección de la serie Fibonacci
Fuente: Elaboración Propia

Se eligió la serie de fibonacci para asegurar que las estimaciones sean más certeras. IceScrum ofrece una configuración de esfuerzos usando barras de escalas de color (Figura 5).

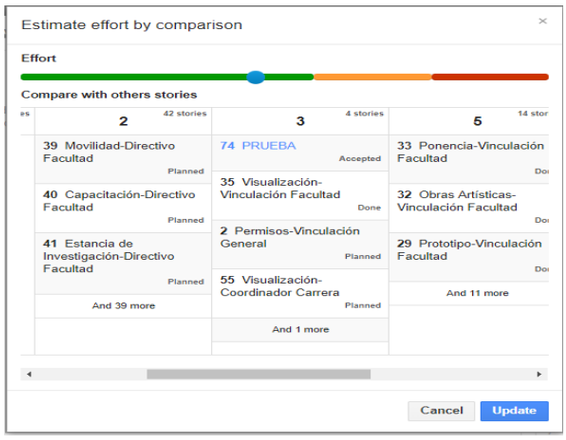

Figura 5 Selección de estimación de esfuerzo
Fuente: Elaboración Propia

Los números de la serie de fibonacci empleados en las historias de usuario de este proyecto son: 2, 3, 5, 8 y 13. Donde los valores 2 y 3 se encuentran en la barra de color verde con un esfuerzo bajo, considerando al 2 como mínimo esfuerzo empleado. Del 5 al 8 se contempla un esfuerzo medio, ubicado en la barra color amarilla, mientras que el valor 13 se juzga como un esfuerzo máximo utilizado, ubicándose en la parte de color rojo de la barra.

De las 69 historias de usuario, 46 se encuentran en el área de la barra color verde conteniendo 42 historias con un esfuerzo de 2, 4 historias con un esfuerzo de 3 en el área de la barra color amarillo, 14 historias con un esfuerzo de 5 y 1 historia con esfuerzo de 8, en el área de color rojo tenemos 8 tareas con un esfuerzo de 13 siendo estas las que mayor esfuerzo implican en nuestro proyecto.

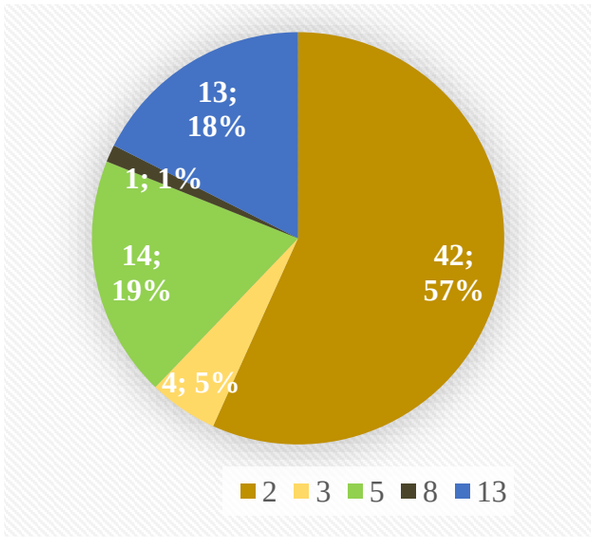

Gráfico 2 Porcentaje del esfuerzo total de las historias de usuario en gráfica de pastel
Fuente: Elaboración Propia

En el gráfico 4 puede observarse la equivalencia de la distribución en porcentajes del esfuerzo en las 69 historias de usuario. Donde las 42 historias con esfuerzo 2 equivalen al 57% de las historias de usuario, 4 historias con esfuerzo 3 al 5%, 14 con esfuerzo 5 al 19%, 1 historia con esfuerzo 8 al 1% y 8 historias con esfuerzo 13 equivalen al 18% del total de historias de usuario (Oyola, 2013).

Pequeña (Small)

Las historias de usuario deben disponer de una duración manejable para todo el equipo, definiendo un esfuerzo razonable para su elaboración, es importante saber que las historias de usuario deben ser estimables y tener un esfuerzo mínimo algo más alto que la duración que se pierde en conversar de ella.

Validable (Testable)

Las historias de usuario deben contener objetivos claros para comprobar el cumplimiento de las expectativas del trabajo para validarlo como realizado.

El equipo elabora en la reunión de planificación de la iteración (Sprint planning) como completar los objetivos/requisitos elegidas para la iteración comprometiéndose a demostrar al cliente el incremento de la preparación del producto para ser entregado al finalizar la iteración.

Esta lista permite ver las tareas donde el equipo está teniendo problemas y no avanza, con lo que le permite tomar decisiones al respecto.

El objetivo es difundir el estado actual de la iteración resultante de la reunión diaria de sincronización (Scrum daily meeting) visualizando los puestos de trabajo del equipo siendo útil en las reuniones futuras.

IceScrum ofrece de manera intuitiva la elaboración del tablero o pizarra de tareas (Scrum Taskboard), identificando aquellas historias de usuario gestionadas en estados de espera, en proceso o terminadas satisfactoriamente en cada iteración ilustrada en la figura 6.

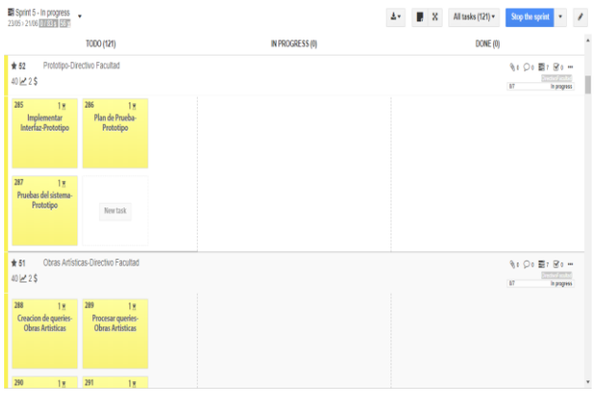

Figura 6 Representación del TaskBoard en IceScrum
Fuente: Elaboración Propia

En la Figura 7 se resume el proceso de evolución de las historias de usuario hasta convertirse en un producto

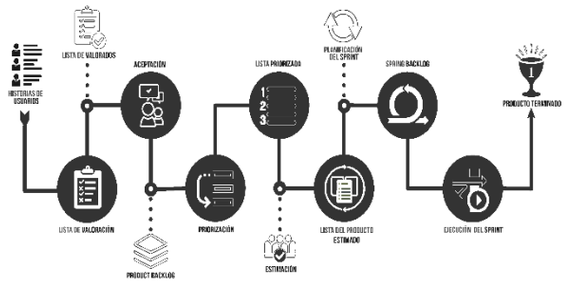

Figura 7 Proceso evolutivo de la historia de usuario
Fuente: Elaboración Propia

Incremento (Sprint)

Los Sprints constituyen la esencia de la metodología scrum, ya que en ellos se ejecutan las tareas de las historias de usuario para convertirse en productos terminados. Los sprints se realizan periódicamente estableciendo una duración para su completa ejecución. En los sprints se eligen las historias de usuario estimadas que serán ejecutadas en la Reunión de Planificación del Sprint (Sprint Planning Meeting), dando seguimiento al trabajo con los Scrums Diarios (Daily Scrums), verificando lo realizado a través de la Revisión del Sprint (Sprint Review) considerando posibles ajustes, y con la Retrospectiva del Sprint (Sprint Retrospective) retroalimentar lo aprendido (Pressman, Troya, 2014).

La plataforma iceScrum proporciona la flexibilidad de planificar los sprint en diversos momentos, incluso declinar una planificación ya realizada por ajustes que puedan surgir. El seguimiento puede realizarse a través de las Sprint notes, llevando un record de los scrums diarios como se muestra en la Figura 8.

ISSN: 2410-4000

La retrospectiva del Sprint también es considerada en iceScrum, donde cada colaborador del equipo puede compartir lo aprendido y dando pautas de mejoras para los próximos sprint, lo anterior mencionado se muestra en la Figura 9 (Satphaty, 2016).

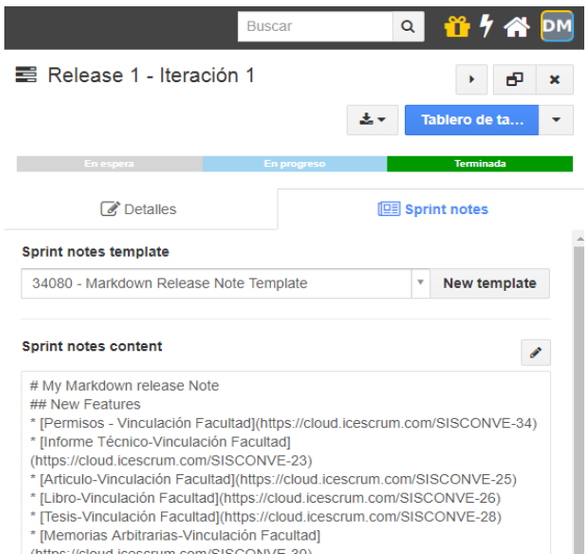

Figura 8 Seguimiento del Sprint
Fuente: Elaboración Propia

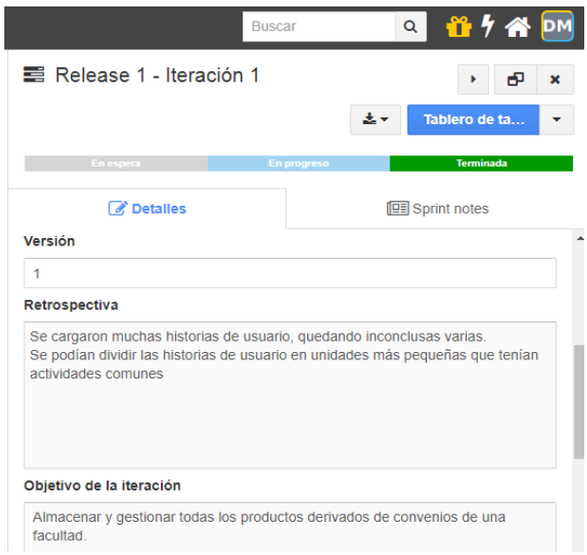

Figura 9 Retrospectiva del Sprint
Fuente: Elaboración Propia

En el proyecto se consideraron 6 sprint con un horizonte no mayor a un mes (Figura 10), como lo sugiere la bibliografía (Schwaber & Sutherland, 2002). Al inicio de nuestro proyecto definimos los objetivos de cada Sprint (Sprint Goal) y una capacidad máxima de esfuerzo por sprint de 57.

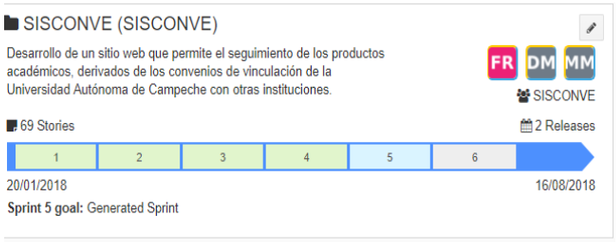

Figura 10 Programación de los Sprints
Fuente*: Elaboración Propia*

Los Scrums Diarios se realizaron de acuerdo con los días laborables del calendario escolar de la Universidad Autónoma de Campeche, debido a que nuestro equipo de desarrollo está conformado por prestadores de servicio social y un docente. (Iglesias, 2011).

En la Tabla 3 presenta las ponderaciones de los sprints:

Iteración	Período	Días hábiles	Historias Usuario	Tareas	Valor	Esfuerzo
1	23/ene-21/feb	22	11	79	440	38
2	22/feb-23/mar	22	5	35	200	43
3	24/mar-22/abril	10	2	14	160	26
4	23/abr-22/may	18	17	119	760	57
5	23/may-21/jun	22	17	119	760	57
6	22/jun-21/jul	21	17	119	800	57

Tabla 3 Ponderación planeada de los Sprints
Fuente: Elaboración Propia

La disponibilidad en los horarios y la dinámica del equipo de desarrollo es muy distinta a una empresa de desarrollo de software, esto nos lleva a planear los sprint considerando cierta holgura debido a la curva de aprendizaje, los tiempos disponibles. (Straccia, Pytel & Cattaneo, 2016). Por otro lado, el dueño del producto, al entrevistarse con los demás stakeholders detectó detalles que repercutían en cambios de las historias de usuario.

Por lo anterior expuesto, al inicio de cada sprint se verificaba si las tareas fueron completadas y en caso de que no se completaran se desplazaban a la siguiente iteración. (Deemer, Benefield, Larman, & Vodde 2009).

La Tabla 4 presenta la relación de historias desplazadas al sprint, así como el número de tareas ejecutadas por iteración.

Iteración	Historias desplazadas	Tareas ejecutadas
1	0	75
2	4	39
3	0	14
4	0	113
5	2	139
6	0	119

Tabla 4 Variación de los Sprint
Fuente: Elaboración Propia

Bajo ese esquema, la ponderación de los sprint terminados se muestra en la Tabla 5

Iteración	Historias de Usuario	Tareas terminadas	Valor	Esfuerzo
1	7	51	280	38
2	9	63	360	43
3	2	14	160	26
4	15	105	620	31
5	19	133	900	83
6	17	119	800	57

Tabla 5 Ponderación terminada de los Sprints
Fuente: Elaboración Propia.

Resultados

La metodología scrum resultó benéfica para el desarrollo de la aplicación web llamada SISCONVE debido a la dinámica del equipo de desarrollo y los requerimientos cambiantes del sistema.

Al comparar los valores de las Tablas 3 y 5, en el Gráfico 3, se puede observar que las historias de usuario fueron subestimadas en el esfuerzo. Debido a que el equipo de desarrollo se conformó de manera eventual, no había un parámetro previo para poder estimar los esfuerzos de manera más precisa, además de ser el primer proyecto de desarrollo para la mayoría de ellos. A causa de la curva de aprendizaje, los primeros tres sprint representaron los de mayor esfuerzo, ocupando 54 días hábiles para ejecutar 18 historias que fueron estimadas con un esfuerzo total de 107; mientras que los últimos 3 sprint requirieron de 61 días hábiles para realizar 51 historias estimadas con un esfuerzo de 171.

Es decir, que a la mitad del proyecto se había realizado apenas el 26% de las 492 tareas planeadas como se muestra en el gráfico 4.

La retrospectiva que ofrece la metodología al final de cada sprint permitió realizar los ajustes necesarios para para alcanzar los productos terminados, además que la dinámica del equipo de trabajo respecto a los tiempos de dedicación pudo elevarse debido al período intersemestral.

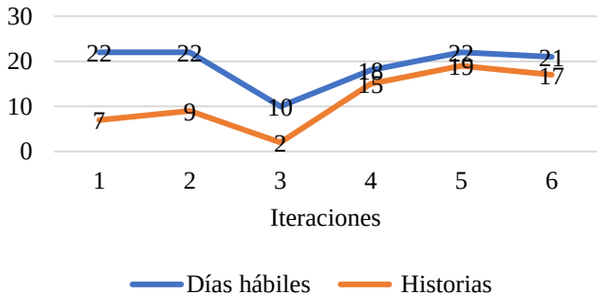

Gráfico 3 Avance del proyecto por iteración
Fuente*: Elaboración Propia*

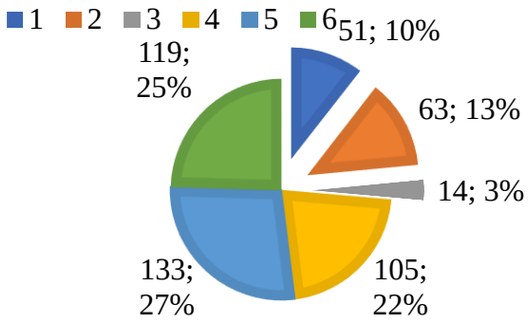

Grafico 4 Tareas ejecutadas por Sprint
Fuente: Elaboración Propia.

Conclusiones

La estimación del esfuerzo en el caso de equipos eventuales requiere de sesiones previas para tipificar los valores de esfuerzos. Es conveniente para los equipos que no tienen un horario fijo de trabajo, realizar una tabla de horas disponibles previo a la Planificación del Sprint. La herramienta iceScrum es potente, sin embargo, puede mejorar en los siguientes aspectos:

– La medición de las tareas terminadas en cada sprint. En iceScrum si una historia de usuario no llega a terminar alguna de sus actividades en el sprint planeado, es asignada al siguiente con todas sus actividades y estimaciones, aunque se hayan ejecutado algunas actividades en el sprint original, impidiendo hacer un cálculo real del esfuerzo efectuado en cada sprint.

ISSN: 2410-4000

– Una vez que ha sido finalizada una iteración, no puede ponderarse nuevamente el esfuerzo de las historias de usuario, por lo tanto, si se valoró inadecuadamente, así permanecerá, aunque las otras historias se ponderen correctamente, será de manera desproporcionada.
– No considera configurar las horas laborables de cada día, ni por cada trabajador, para una mejor planificación.

La metodología scrum es valiosa por su adaptabilidad a los cambios continuos del mercado aportando una ventaja competitiva.

Agradecimiento

Agradecemos a la Universidad Autónoma de Campeche por el apoyo en la realización de esta investigación, en especial al director de la Facultad de Ingeniería M. en C. Guadalupe Manuel Estrada Segovia.

Referencias

Alliance, S. (2016). Scrum Framework.

Bass, L., Clements, P., & Kazman, R. (2012). *Software architecture in practice*. Third Edition. Addison-Wesley Professional.

Deemer, P., Benefield, G., Larman, C., & Vodde, B. (2009*).* Información básica de SCRUM. *California: Scrum Training Institute*.

Cohn, M. (2004). *User stories applied: For agile software development.* Addison-Wesley Professional.

Guerrero Hernádez, O. E., Rosas Cabrera, G., Cañete Satibañes, C. U., & Cedillo Baños, L. (2017). Repositorio móvil para el control de maleza en el cultivo de maíz. *Revista de Cómputo Aplicado*, 1-6.

Letelier, P., & Penadés, M. C. (2012). Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP).

Iglesias-Solano, A. M. (2011). Story Points y su importancia en la estimación de proyectos bajo el enfoque agil. *Revista Investigación y Desarrollo en TIC, 50-58.*

Macías Brambila, H. R., López Laguna, A. B., González Del Castillo, E. E., & Tolosa Carillo, E. (2017). Servidor de aplicaciones como evidencia para sinergia Academia-Empresa MyPyMES de México. *Revista de Tecnología Informática*, Ecorfan-Spain, 39-43.

Martel, A. (2014). *Gestión práctica de proyectos con scrum: desarrollo de software ágil para el scrum máster.* (Vol 1). Antonio Martel.

May, M., Morales, Y., Marrufo, J., & Martín, M. (2013). Implementación de un sistema para el control de activos ISOPTEC, bajo el estándar ITIL y metodología ágil SCRUM. *Ciencias de la Ingeniería y Tecnología Handbook T-II*, 176.

Oyola, J. R. C. (2013). Análisis, propuesta y representación de indicadores en proyectos ágiles con SCRUM. *Cuaderno Activa, (5), 11-21*. Recuperado de http://ojs.tdea.edu.co/index.php/cuadernoactiva/article/view/111

Pabón, J. L. L. (2016). Gestión del Tiempo en Proyectos De Desarrollo de Software. *Revista Teckne, 11(2), 19-28.*

Pichler, R. (2010). *Agile product management with scrum: Creating products that customers love*. Addison-Wesley Professional.

Pressman, R. S., & Troya, J. M. (2014). *Ingeniería del software.* Editorial Mc Graw Hill.

Satphaty, T. (2016). Cuerpo de Conocimiento de SCRUM. Phoenix, Arizona: SCRUMstudy.

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

Schwaber, K., & Sutherland, J. (2013). La guía de scrum: La guía definitiva de scrum, las reglas del juego. *Recuperado de http://www. scrumguides. org/docs/scrumguide/v1/Scrum-Guide-ES. pdf*.

Sommerville, I., Sawyer, P., & Viller, S. (1999). Managing process inconsistency using viewpoints. *IEEE Transactions on Software Engineering*, *25*(6), 784-799.

Straccia, L., Pytel, P., & Pollo Cattaneo, M. F. (2016). Metodología para el desarrollo de software en proyectos de I+ D en el nivel universitario basada en Scrum. In *XXII Congreso Argentino de Ciencias de la Computación (CACIC 2016).*