Profundización sobre metodologías Ágiles: SCRUM

- Jonathan Orna Ponce
- Fermín Rodríguez Rico
- Juan Camilo Castro Cendales
- Alejandro Rodrigo Ndong Ovono

ÍNDICE

Declaración de no plagio **17**

1.- Concepto de metodología ágil y principales diferencias con los modelos tradicionales…… pag 3

1.1 Concepto de metodología ágil…… pag 3

1.2 Principales diferencias con los modelos tradicionales…… pag 4

2.- Ciclo de vida de Scrum…… pag 5

3.- Roles…… pag 7

4.- Artefactos…… pag 8

5.- Actividades…… pag 10

5.1 First sprint…….. pag 10

5.2 Sprint..... pag 10

5.3 Sprint planning..... pag 10

5.4 Release. ..... pag 10

5.5 Daily meeting. ..... pag 11

5.6 Sprint review..... pag 11

6.- Estimaciones..... pag 11

6.1 Planning póker..... pag 11

6.2 Affinity estimation..... pag 12

7. Las Historias de Usuario..... pag 12

Bibliografía y webgrafía…… … pag 16

Declaración de no plagio…… pag 17

Índice de figuras

Figura 1…… ……..…pag 6

Figura 2…… ……..pag 13

Figura 3…… ……..pag 14

1.- Concepto de metodología ágil y principales diferencias con los modelos tradicionales.

1.1 Concepto de metodología ágil

Para entender el concepto de metodología ágil y el porqué se le da ese nombre hay que partir del siguiente concepto:

Ágil es la capacidad de crear y responder al cambio. Es una forma de lidiar con un entorno incierto y por último tener éxito.

Centrándonos en la metodología ágil, el desarrollo ágil de software es más que prácticas, como la programación de pares, el desarrollo basado en pruebas, stand-ups(reuniones diarias de progreso), sesiones de planificación y sprints. Es un término general para un conjunto de entornos de trabajos y prácticas basadas en los principios expresado en el Manifiesto para el desarrollo de software ágil y los 12 principios que lo respaldan.

Los métodos ágiles fomentan respuestas rápidas y flexibles al cambio mediante la planificación adaptativa, la identificación de requisitos colaborativos y la racionalización entre el equipo interfuncional autoorganizado así como el desarrollo gradual de soluciones.

1.2 Principales diferencias con los modelos tradicionales

Metodologías Ágiles	Metodología 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
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(menos de 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. *Recoge esquemáticamente las principales diferencias de las metodologías ágiles con respecto a las tradicionales.*

Los valores siguientes son características que diferencian un proceso ágil de uno tradicional:

-Al individuo y las interacciones del equipo de desarrollo sobre el proceso y las herramientas. La gente es el principal factor de éxito de un proyecto de software. Es más importante construir un buen equipo que construir el entorno. Es mejor crear el equipo y que éste configure su propio entorno de desarrollo en base a sus necesidades.

-Desarrollar software que funciona más que conseguir una buena documentación. La regla a seguir es "no producir documentos a menos que sean necesarios de forma inmediata para tomar una decisión importante". Estos documentos deben ser cortos y centrarse en lo fundamental

-La colaboración con el cliente es más importante que la negociación de un contrato. Se propone que exista un interacción constante entre el cliente y el equipo de desarrollo. Esta colaboración entre ambos será la que marque la marcha del proyecto y asegure su éxito.

-Responder a los cambios más que seguir estrictamente un plan. La habilidad de responder a los cambios que puedan surgir a lo largo del proyecto (cambios en los requisitos, en la tecnología, en el equipo. etc.) determina también el éxito o fracaso del mismo. Por lo tanto, la planificación no debe ser estricta sino flexible y abierta.

2.- Ciclo de vida de Scrum

Scrum es un entorno de trabajo ideal para proyectos que requieren un cambio rápido de requisitos. El ciclo de vida del Scrum comienza con un registro de prioridad, pero este no es una guía para saber cómo se desarrolla el trabajo.

El desarrollo se realiza mediante iteraciones a las que denominamos sprints y tienen una duración de 30 días. Con cada sprint obtenemos un incremento del software, el cual se muestra al cliente. También podemos destacar las reuniones que se llevan a cabo a lo largo del proyecto. Los pasos claves en el ciclo de vida del Scrum son los siguientes:

1. El Product Owner redacta las User Stories y las sitúa en el Product Backlog.
2. El Product Owner prioriza estas User Stories y ordena el Product Backlog en consecuencia.

3. El equipo Scrum se junta en la reunión de planificación del Sprint, con el objetivo de establecer la lista de las User Stories que se tratarán durante el Sprint. Esto forma el Sprint Backlog y a continuación se descomponen en tareas por el equipo de desarrollo.
4. En este punto el Sprint puede comenzar con una iteración de 2, 3 o 4 semanas.
5. El equipo se reúne diariamente para realizar la Melé diaria(Daily meeting)
6. Como consecuencia del Sprint, obtenemos un producto potencialmente entregable que forma de una demostración durante la revisión del Sprint.
7. El ciclo termina con la retrospectiva del Sprint.

Y a continuación, solo hay que repetir todo de nuevo.

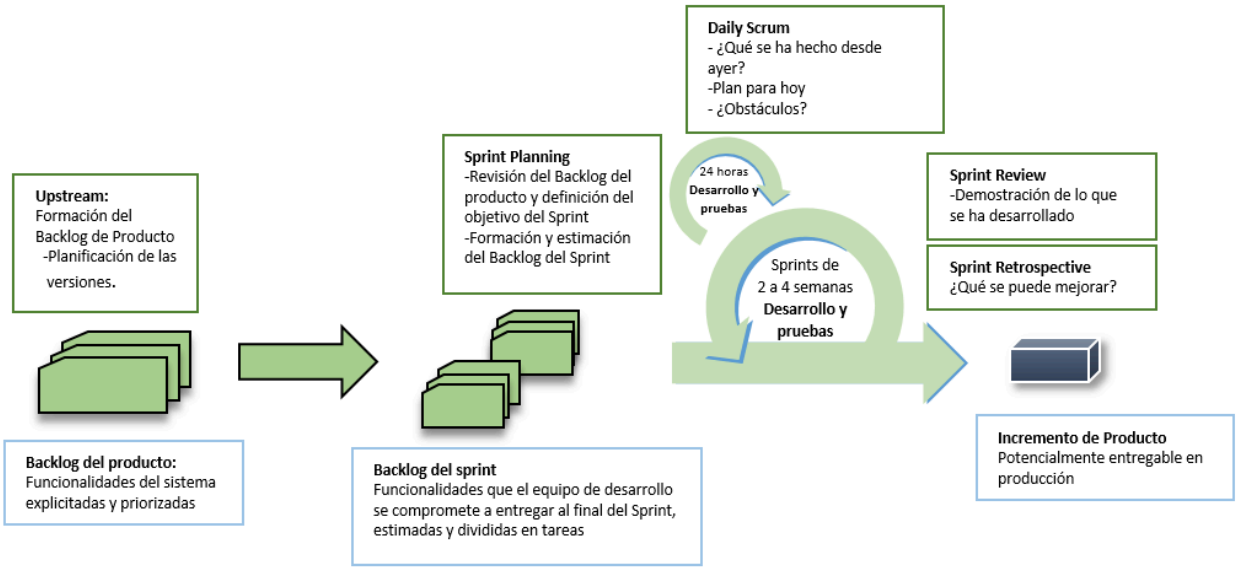

Figura 1.

3.- Roles (Product Owner, Scrum Master, Development Team, Stakeholder)

- **Product Owner:** Se encarga de organizar al equipo a través de los conocimientos que posee acerca de los usuarios, la competencia y las tendencias a futuro para el tipo de sistema que se esté desarrollando. Fomenta un ambiente de trabajo óptimo y una visión clara del objetivo a construir por el equipo.
- **Scrum Master:** Es el responsable de promover y apoyar al Scrum, asesorando y formando a los diferentes miembros para trabajar de forma autoorganizada. Además, ayuda a los miembros del equipo a entender las reglas, los valores y las prácticas de la **teoría Scrum** para que puedan proporcionar estrategias comprensibles con gran eficacia. De esta forma asegura la configuración, el diseño y la mejora continua de las prácticas de scrum en la organización.
- **Development Team:** Conjunto de personas (entre 5 y 9 profesionales) de **gran capacidad técnica** que se encarga de desarrollar el producto y entregar un incremento de software al final del ciclo de desarrollo.
  Uno de los aspectos más destacados del Development Team es que se auto-organiza y se auto-gestiona , para ello:
  - Seleccionan los requisitos que prevén completar en una iteración.
  - Estiman la complejidad de cada requisito.
  - Durante la iteración colaboran para poder completar un requisito.
  - Al finalizar la iteración, realizan una retrospectiva de cada iteración para mejorar el rendimiento del equipo.
- **Stakeholder**: hace referencia a todas aquellas personas o colectivos que tienen algún tipo de interés en torno a una empresa o sus actividades comerciales. Pueden generar beneficios para las empresas tales como un crecimiento sostenido, obtención de información estratégica necesaria para la toma de decisiones y mayor innovación de productos y servicios.

  Existen dos tipos clásicos de stakeholders:

- **Grupo primario:** aquellos considerados como fundamentales para la marcha de cualquier empresa, mantienen una relación directa y estrecha con los negocios.
- **Grupo secundario:** aquellos que no participan de forma directa en las acciones ni en los procesos comerciales.

4.- Artefactos (Product backlog, Sprint backlog, Scrum board)

- **Product backlog:** es una **lista de todas las tareas** que se pretenden realizar durante el desarrollo de un proyecto y que otorga al equipo una visión panorámica del proyecto. Cada tarea se expresa a través de user stories (descripciones de la funcionalidad deseada).

  Cualquier miembro del equipo puede agregar items en el Product Backlog. Sin embargo, es el Product Owner quién se encarga de gestionarlo.

  En función de las necesidades del cliente y la complejidad de cada tarea se establecen ciertas prioridades, los de mayor prioridad suelen ser más claros y estar definidos con mayor detalle. En base a lo mencionado anterior, se debe comprender que el Product backlog es una parte que evoluciona continuamente pues debe adaptarse en cada momento a posibles mejoras o nuevos enfoques de mercado.

  El Product backlog se organiza de la siguiente forma:

  1. Features.
  2. Bugs.
  3. Technical work.
  4. Knowledge acquisition.
- **Sprint backlog:** es un conjunto de tareas que se tienen que completar durante un sprint con el fin de lograr el Sprint Goal y avanzar hacia el resultado deseado. El Development Team y el Product Owner determinan continuamente qué se puede hacer para lograr un objetivo claro para el actual Sprint y en caso de tener que realizarse cambios en el Sprint backlog se debería realizar una nueva reunión de Planificación de Sprint (Sprint Planning meeting) .

La práctica habitual es gestionar el Sprint backlog mediante un tablero con post its. Al lado de cada objetivo se colocan las tareas necesarias para completarlo y se van moviendo para cambiarlas de estado. **Pendiente, en progreso, hechas.**

Es útil porque **descompone el proyecto en tareas de tamaño adecuado** para determinar el avance a diario; **e identificar riesgos y problemas sin necesidad de procesos complejos de gestión.** Es también una herramienta de soporte para la comunicación directa del equipo.

- **Scrum board:** es un tablero físico o virtual que sirve como **punto de unión** entre todos los integrantes del **Scrum Team** y el **Product Owner**. Es empleado por el equipo durante cada Sprint para planificar y realizar un seguimiento del progreso, los miembros del equipo indican la situación actual de la tarea que desarrollan.

  Generalmente suele estar formado por cuatro o cinco columnas:

  - **Columna de Historias:** indica las tareas
  - **Columna de "para hacer":** visualiza las tareas que deberán iniciarse.
  - **Columna de "en curso":** muestra aquellas tareas están siendo desarrolladas.
  - **Columna de pruebas:** contiene las tareas que están en período de pruebas tras haber sido completadas.
  - **Columna de "realizado":** para las tareas que se han completado y probado con éxito.

5.- Actividades.

5.1 First sprint:

Tiene como objetivo los preparativos previos a comenzar el desarrollo. Normalmente, se realizan las siguientes tareas: se dejan listos los entornos de desarrollo, se trabaja en el product backlog, principalmente en dejar listas las historias de usuario, priorizadas y estimadas, se hace una previsión del reparto de historias de usuario por iteración y se hace un estudio de la arquitectura.

5.2 Sprint:

El Sprint es el período en el cual se lleva a cabo el trabajo en sí. Es recomendado que la duración de los sprints sea constante y definida por el equipo teniendo en cuenta la experiencia. Se puede comenzar con una duración de sprint en particular e ir ajustandolo con base en el ritmo del equipo, aunque sin relajarlo demasiado. Al final de cada sprint, el equipo deberá presentar los avances logrados, y el resultado obtenido es un producto que, potencialmente, se puede entregar al cliente.

5.3 Sprint planning:

Planificación del Sprint.

Al comienzo de un sprint, el equipo de scrum tiene un evento de planificación de sprint. Participa el equipo Scrum al completo, pero no Stakeholders. Uno de los objetivos de la reunión es identificar y comunicar cuánto del trabajo es probable que se realice durante el actual Sprint.

5.4 Release:

El Sprint de release es aquel que implementa aquellas tareas necesarias para poner el sistema en producción. Muchos proyectos requieren al final de un ciclo de release, es decir después de un número de Sprint previos a un paso a producción. Un último Sprint dedicado a preparar el paso a producción.

En este Sprint de release se realizan tareas como el despliegue, generación de “scripts” de recuperación del sistema en caso de problemas al pasar a producción, tareas

relacionadas con las bases de datos en producción, documentación, pruebas de carga, etc.

5.5 Daily meeting:

Durante el desarrollo de un sprint, el equipo mantiene una reunión diaria llamada el "daily meeting scrum". Usualmente estas reuniones se realizan en el mismo lugar y a la misma hora, en cada uno de los días.

Idealmente, el daily meeting se realiza por la mañana para planificar la jornada de trabajo. Estas reuniones son estrictamente desarrolladas con un tiempo límite de 15 minutos. Esto hace que la reunión sea breve y se traten puntos importantes.

Esta reunión diaria no se realiza con el fin de resolver problemas específicos. Si existen problemas específicos, estos son tratados de forma externa al daily meeting, con el subgrupo correspondiente, justo después de esta reunión.

5.6 Sprint review:

El Sprint Review es la reunión de negocio por excelencia en Scrum. A la reunión acuden el equipo Scrum y Stakeholders. El Product Owner lo organiza y lidera; durante el mismo se inspecciona el Incremento y se adapta el Product Backlog, no por el Scrum Master ni el Development Team.

6.- Estimaciones.

6.1 Planning póker:

Planning Póker es una técnica para calcular una estimación basada en el consenso, en su mayoría utilizada para estimar el esfuerzo o el tamaño relativo de las tareas de desarrollo de software.

El Planning Poker, según su creador, nació para minimizar el siguiente problema: estimaciones que llevaban mucho tiempo y que no participe todo el equipo en la estimación.

6.2 Affinity estimation:

La estimación por afinidad es una técnica de estimación que permite estimar un gran número de elementos en un tiempo ridículo.

Esta técnica de estimación funciona cuando tienes muchos elementos con los que comparar y también cuando llevas un rato para tomar decisiones muy rápidas basadas en decisiones que has tomado hace segundos o minutos

7. Las Historias de Usuario (User stories):

El primer paso en la estimación y planificación del proyecto es la creación del ***Product backlog,*** o definición del proyecto a realizar. El backlog nos permite seguir el recorrido, incidencias, y vivencias de un futuro proyecto. Al definir nuestro proyecto podemos reconocer las metas y nuestros objetivos. Esos objetivos son los que se expresan a través de historias de usuario. Una historia de usuario es un requisito u oportunidad de negocio visto desde el punto de vista del usuario o consumidor final. Es ponerte en la piel de un consumidor o usuario final y descubrir las satisfacciones o/y dificultades que puede dar nuestro producto. En ese caso como ingenieros de software reducir las dificultades.

- Planificación con Visual Story Mapping

La planificación con Story mapping consiste en workshops donde participan usuarios finales, stakeholders, Product Owner y equipo de desarrollo cuyo principal objetivo es que todos los participantes compartan un mismo objetivo de proyecto, la misma visión de su alcance, riesgos/dificultades y acciones de mitigación desde el inicio del proyecto, de manera que se generen sinergias y sentimiento de equipo para conseguir un mismo objetivo.

Story mapping nos ayuda a organizar y dar prioridad a cada historia del proyecto. A diferencia del método backlog, Story mapping hace visible y te ayuda a confirmar la

completitud de tu backlog o, en otro caso, genera dudas, que te ayuda a confeccionar mejor cada historia o solucionar los errores.

Además es útil para mostrar la relación entre las historias a gran escala, ofrece la opción de priorizar cada historia según el contexto y es útil para reflejar todos los planes y valores de funcionalidad al completo.

El pilar de un Story map son las tareas del usuario o ***user task.*** Bajo el lema de que se encuentran soluciones a través de las interacciones, para la planificación de un story map debemos tener en cuenta estos tres niveles: El objetivo (*goa*l), las tareas (*task*) y las herramientas (*too*l).

Figura 2.

El proyecto de Software contiene o es resultado de las características de las tareas y los objetivos proyectados en su realización.
Cada nivel está compuesto en subniveles de acción. Caso principal el de tareas, que está subdividido en otras tareas.

- Cada tarea tiene un objetivo que debe conseguirse.

- Cada tarea puede subdividirse en tareas más pequeñas.
- Al realizar una tarea se debe analizar la intención por el que hacemos uso de tal o cual herramienta.
- A menudo es normal que las tareas se junten con actividades de otras tareas relacionadas.

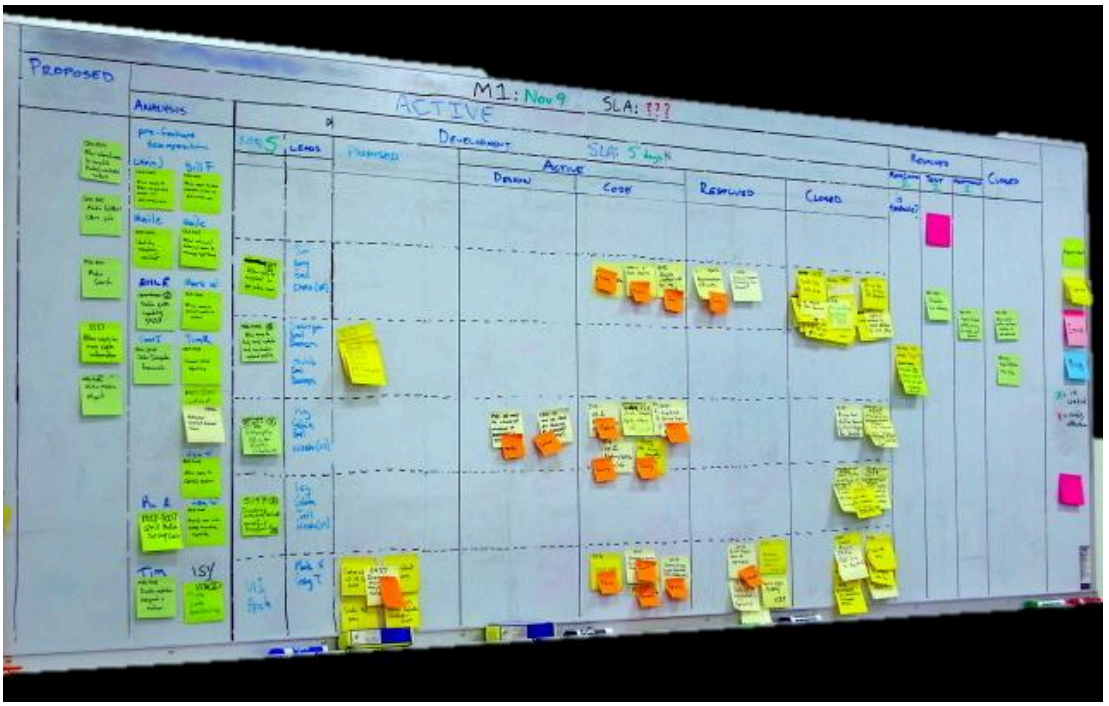

Figura 3.

- Pruebas de aceptación

Las condiciones de satisfacción o afirmación de los objetivos suelen ponerse en forma de **criterios de aceptación**, pruebas que se realizan para verificar si el sistema se comporta de la manera esperada. Son los requisitos de rendimiento y las condiciones esenciales y mínimas, que deben cumplirse antes de que demos por finalizado.

Para ello se puede utilizar la sintaxis de **BDD (Behaviour Driven Development**). El BDD es un proceso de desarrollo de software que busca un lenguaje común para unir la

parte técnica y la de negocio, y que sea desde ese lenguaje común desde donde arranque el Testing y, desde ahí, el desarrollo.

Esta técnica permite evitar la aparición de errores por malos entendidos y evitar re-trabajar. Por ello es recomendable no empezar a desarrollar en una iteración sin antes haber escrito los casos de prueba, especialmente por que es más barato escribir texto y pensar en cómo desambiguar los requisitos que arreglar errores importantes debido a su mal entendimiento.

Bibliografía y webgrafía.

- http://roa.ult.edu.cu/bitstream/123456789/476/1/TodoAgil.pdf
- https://proyectosagiles.org/2010/12/15/planificacion-agil-vs-planificacion-tradicional/
- https://www.heflo.com/es/definiciones/metodologia-agil/
- https://programacionymas.com/blog/scrum-product-owner
- http://noticias.universia.edu.uy/educacion/noticia/2018/03/15/1158527/capacidades-funciones-product-owner.html
- https://proyectosagiles.org/cliente-product-owner/
- http://www.ceolevel.com/scrum-master-que-es-y-que-no-es
- https://www.scrum.org/resources/what-is-a-scrum-master
- https://www.scrum.org/resources/what-is-a-scrum-development-team
- https://jeronimopalacios.com/scrum/
- https://jeronimopalacios.com/2016/05/la-receta-conseguir-buen-sprint-review/
- https://www.javiergarzas.com/2013/07/el-sprint-cero-y-el-sprint-de-release.html
- https://programacionymas.com/blog/daily-scrum-meeting
- https://samuelcasanova.com/2016/07/estimacion-por-afinidad-estimar-a-lo-bestia/
- https://www.ediciones-eni.com/open/mediabook.aspx?idR=715e049f952b9edab35455b751df4451

- https://programacionymas.com/blog/scrum-product-backlog
- https://www.scrum-institute.org/The_Scrum_Product_Backlog.php
- https://www.agilealliance.org/glossary/sprint-backlog/#q=~
- https://www.obs-edu.com/es/blog-investigacion/marketing-y-comunicacion/stakeholders-ejemplos-para-entender-el-concepto
- https://www.webyempresas.com/que-son-los-stakeholders/
- https://www.scruminc.com/sprint-backlog/

Principios y Fundamentos de la Ingeniería del Software

Declaración de no plagio

Los componentes del grupo de trabajo, Juan Camilo Castro Cendales, Jonathan Orna Ponce, Alejandro Rodrigo Ndong Ovono y Fermín Rodríguez Rico.

declaramos que nuestro trabajo es original, no habiéndose utilizado fuentes sin ser citadas debidamente. De no cumplir con este compromiso, somos conscientes de que, de acuerdo con el Artículo 25 *Incidencias en la elaboración de trabajos u otras actividades de evaluación*, de la Normativa de Evaluación para las Titulaciones de Grado y Máster Oficial de la Universidad de Huelva, […] *en la realización de trabajos, el plagio y la utilización de material no original, incluido aquel obtenido a través de Internet, sin indicación expresa de su procedencia […], podrá ser considerada causa de calificación de suspenso […], todo ello con independencia de otras responsabilidades que puedan recaer tras la conclusión del correspondiente procedimiento […].*

Y para que así conste firmamos el presente documento.

En Huelva a 19 de Marzo de 2019