UNIVERSIDAD NACIONAL DE TUCUMÁN

FACULTAD DE CIENCIAS ECONOMICAS
UNIVERSIDAD NACIONAL TUCUMAN

SCRUM COMO MÉTODO ÁGIL IMPLEMENTADO EN UNA EMPRESA DE DESARROLLO DE SOFTWARE

Autores: Burgos, Fernanda Gabriela
Fanari, Ana Carolina
Sández, Verónica Fernanda

Director: Rodríguez, María Fernanda

2019

Trabajo de Seminario: Licenciatura en Administración de Empresas

RESUMEN

La finalidad del presente trabajo es analizar la implementación de Metodologías ágiles en una empresa dedicada al desarrollo de Software, para tal fin estudiamos el marco teórico para poder realizar un paralelismo entre la Metodología Tradicional y la Metodología Ágil.

Los motivos que impulsaron a la utilización de esta nueva metodología son los avances de las tecnologías de información y los continuos cambios que se producen en el mercado. Esta metodología ágil se enfoca en la adaptabilidad a los cambios, con equipos auto-organizados, autonomía a la hora de tomar decisiones e involucramiento del usuario o cliente, el equipo y los interesados.

Actualmente existen una gran variedad de metodologías ágiles, las que se seleccionan dependiendo de las necesidades de los clientes y las particularidades de cada proyecto.

La empresa estudiada, Censys, adopta la metodología ágil Scrum, es un marco de trabajo que define un conjunto de prácticas y roles, realizándose entregas parciales y regulares del producto final que aportan un beneficio al receptor del proyecto.

Si bien la aplicación de este enfoque fue compleja, el cambio fundamental se observó en la cultura de la organización, los resultados obtenidos superaron lo esperado, lo cual se materializó principalmente en el incremento de la rentabilidad, capacidad de productiva, disminución de la tasa de errores y disminución en la rotación de recursos.

INTRODUCCION

Según estudios realizados por el *Project Management Institute* (PMI), establece que el 71% de las organizaciones a nivel mundial utilizan metodologías ágiles y más del 75% coinciden que éstas son cruciales para el éxito en la transformación digital.

Resulta de gran interés indagar dentro de estos métodos, la potencialidad de *Scrum* como herramienta útil y eficiente para ayudar a las organizaciones a incrementar la productividad, acelerar la obtención de ganancias, reducir el *time to market*, y disminuir fallos operativos.

En el presente trabajo se aborda el desarrollo teórico de las metodologías ágiles y la implementación de *Scrum* en la empresa Censys, dedicada al desarrollo de *Software* para entidades bancarias.

El diseño metodológico seleccionado fue la entrevista y la observación directa, ambos de tipo exploratorio con técnicas cualitativas para la recolección de datos.

CAPITULO I

NOCIONES BASICAS DE UN SISTEMA

Sumario: 1.- Definición de sistema. 2.- Sistema de información. 3.- Definición de Software. 4.- Clasificación de Software. 5.- Desarrollo de Sistemas.-

1.- Definición de sistema:

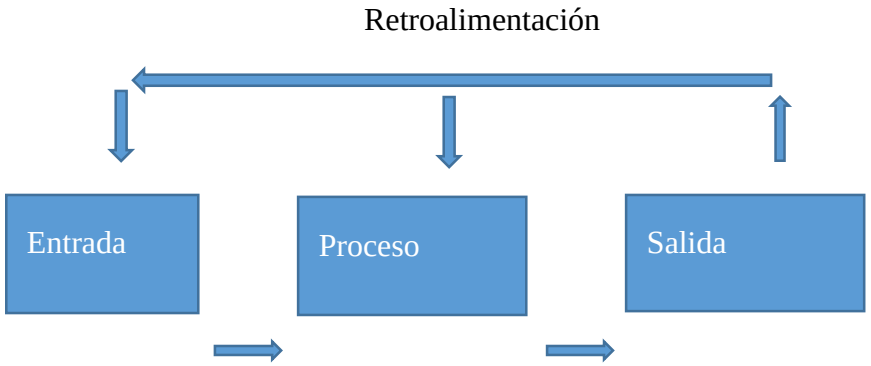
Un sistema es un conjunto de elementos o componentes que interaccionan para alcanzar un objetivo. ([1]) Los elementos por sí mismos y las relaciones entre ellos determinan cómo funciona el sistema. Éste tiene entradas, mecanismos de procesamientos, salidas y retroalimentación.

2.- Sistema de información:

Un sistema de información es un conjunto de elementos o componentes interrelacionados que recaban (entrada), manipulan (proceso), almacenan y distribuyen

([1]) STAIR, Ralph y REYNOLDS, George, Principios de Sistemas de Información, trad. por. Víctor Campos Olguín y Carlos Roberto Codero Pedraza, 9º Edición, Cengage Learning Editores, (México 2010), pág. 8.

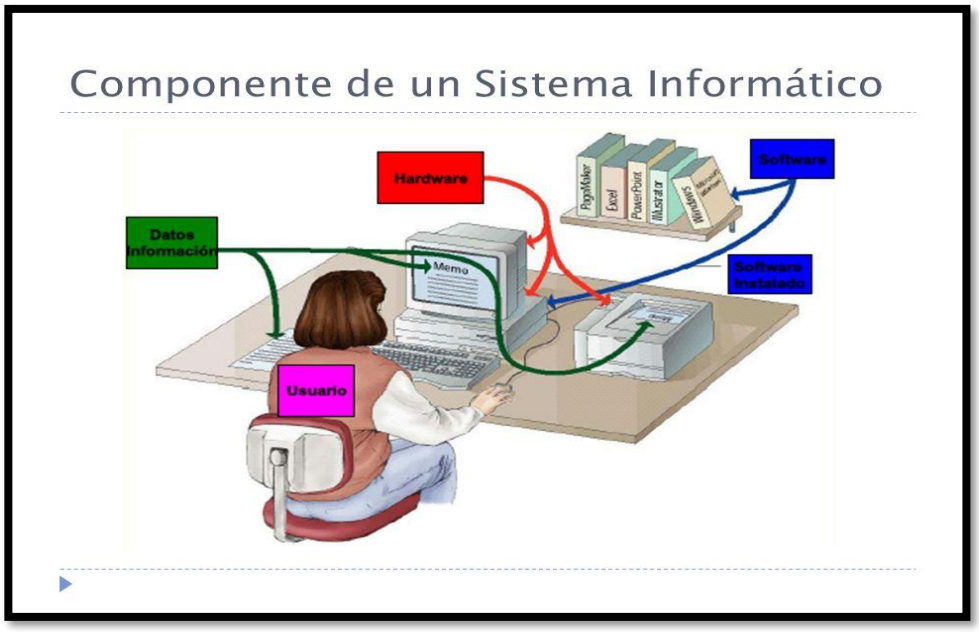
(salida) datos e información y proporcionan una reacción correctiva (mecanismo de retroalimentación) si no se ha logrado cumplir un objetivo[2]. El mecanismo de retroalimentación es el componente que ayuda a las organizaciones a cumplir sus objetivos, tales como incrementar sus ganancias o mejorar sus servicios al cliente.



- **Entrada**: Se define como la actividad consistente en la recopilación y captura de datos.
- **Procesamiento**: Significa la conversión o transformación de datos en salidas útiles.
- **Salida**: Involucra la producción de información útil, por lo general en la forma de documentos y reportes. En algunos casos, la salida de un sistema puede convertirse en la entrada de otro.
- **Retroalimentación**: Es la información proveniente del sistema que se utiliza para realizar cambios en las actividades de entrada y procesamiento.

Un sistema de información basado en computadora es un conjunto único de hardware, software, bases de datos, telecomunicaciones, personas y procedimientos configurados para recolectar, manipular, almacenar y procesar datos para convertirlos en información.

([2]) Ibídem, Pág. 4.



3.- Definición de Software:

Es la suma total de los programas de ordenador, procedimientos, reglas, la documentación asociada y los datos que pertenecen a un sistema de cómputo[3]. Pero el *software* no son sólo los programas sino todos los documentos asociados, la configuración y el conjunto de datos necesarios para lograr que los programas operen correctamente.

4.- Clasificación del software:

- ***Software* de sistemas**: Son los programas que permiten a otros poder operar. Un ejemplo de este tipo de software son los sistemas operativos.
- ***Software* en tiempo real**: Son programas que interactúan de manera natural con el ambiente. Este tipo de *software* generalmente se encuentra en ambientes de producción, ventas y gestión, aunque

([3]) PRIOLO, Sebastián, "Métodos agiles", Primera Edición, Gradi S.A. (Buenos Aires, 2009). Pág. 20.

todos los programas que estamos acostumbrados a utilizar hoy en día podrían caer, en mayor o menor medida, en esta definición.

- ***Software* de ingeniería**: Son herramientas que facilitan la toma de datos, los cálculos, la generación de imágenes, el modelado de objetos y otro tipo de apoyo.
- ***Software* embebido**: Está instalado en otros elementos o productos industriales, comúnmente conocido como *software* empotrado.

5.- Desarrollo de sistemas:

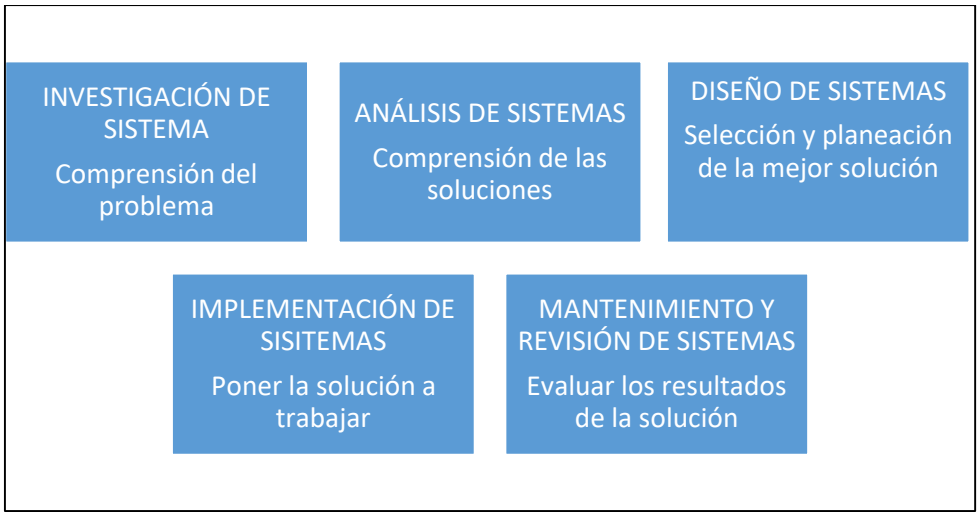
El desarrollo de sistemas se define[4] como la actividad consistente en crear o modificar los sistemas de negocios. Los proyectos de desarrollo de sistemas pueden ser pequeños o muy grandes y abarcar campos del conocimiento tan diversos como el campo bursátil o los videos juegos.

El desarrollo de sistemas de información que cumplan con las necesidades del negocio representa una tarea muy compleja y difícil, tanto que es muy común que los proyectos relacionados con los sistemas de información excedan el presupuesto y las fechas de terminación programadas. Una estrategia para mejorar los resultados de un proyecto de este tipo consiste en dividirlo en varias etapas, cada una de las cuales debe contar con una meta bien definida y un conjunto de tareas a cumplir.

- **Investigación y análisis de sistemas**: el objetivo de la investigación de sistemas es obtener una compresión clara del sistema que se desea resolver o la oportunidad que se enfrenta. Si la decisión es aplicar la solución, el siguiente paso, el análisis de sistema, define los problemas o oportunidades del sistema actual.
- **Diseño, implementación, mantenimiento y revisión de sistemas**: el diseño de sistemas determina la forma en que trabajara el nuevo sistema con el fin de cumplir las necesidades de la empresa definidas durante la etapa de análisis. La implementación de sistemas incluye

(4) STAIR, Ralph y REYNOLDS, George, op. cit., pág. 26

el diseño y adquisición de los diferentes componentes que los conformaran (hardware, software, base de datos, etc.) y que se definieron en la etapa de diseño, su ensamble y la puesta en marcha de la nueva herramienta. El propósito del mantenimiento y revisión de sistemas consiste en verificarlo y modificarlo de tal manera que siga cumpliendo con las necesidades del negocio.



CAPITULO II

METODOLOGIAS TRADICIONALES

Sumario: 1.- Metodologías vs Paradigmas. 2.- Metodologías tradicionales. 3.- Metodología ágil.-

1.- Metodologías vs Paradigmas:

Constantemente se utilizan los términos[5] **metodología, práctica, método** o **paradigma** como sinónimos cuando en realidad, en la construcción de *software*, no lo son.

Existe un ciclo de vida del *software* que puede ser dividido en fases. Cada una de ellas debe ser completada mediante la aplicación de tareas, técnicas y herramientas. Una metodología es el elemento que nos dice cuáles son las herramientas que se van a usar, para quiénes realizarán las tareas, en qué momento y cuáles son consideradas tareas críticas, entre otras cosas. Existen innumerables metodologías y

(5) PRIOLO, Sebastián, op. cit. Pág. 31

cada una posee sus ventajas y desventajas, lo que hace necesario tomar la decisión de cuál utilizar para cada desarrollo en particular.

Al referirnos a **paradigma**, damos como opciones el estructurado o el orientado a objetos (entre otros). Estos no son otra cosa que la forma de desarrollar o ver el problema. No definen el ciclo de vida, ni tareas concretas a realizarse ni el momento en que se efectúan.

2.- Metodologías tradicionales:

Las metodologías tradicionales proponen[6] reglas inclusivas, es decir, proporcionan unas pautas de actuación que indican con detalle que es lo que hay que hacer en cada momento del proyecto y en cada una de las situaciones posibles. Dependiendo del tipo de proyecto que vayamos a llevar a cabo, una metodología de este tipo puede ayudar a organizarnos y aportarnos mucho valor. Esto ocurre cuando el cliente tiene claros todos los requisitos y las necesidades que quiere cubrir al inicio de su proyecto y tiene la certeza de que no va a necesitar que se realice ningún cambio ni en los requisitos ni en el alcance de los mismos. En estos casos se podrá calcular con bastante precisión el coste del proyecto y la duración del mismo. En definitiva, podremos organizarnos sin miedo a correr demasiados riegos.

Un enfoque tradicional, propone fijar los requisitos con un alto nivel de detalle al inicio del proyecto y a partir de ellos, se hace una estimación del coste y de la fecha de entrega del mismo.

Modelo de cascada

Las principales etapas de este modelo se transforman en actividades fundamentales de desarrollo[7]:

([6]) ALVAREZ GARCÍA, Alonso, DE LAS HERAS DEL DEDO, Rafael, y LASA GÓMEZ Carmen. "Métodos Agiles y Scrum". Grupo Anaya S.A. (Madrid 2012). Pág. 32.

([7]) SOMMERVILLE, Ian. Ingeniería del Software, trad. por. Víctor Campos Olguín, 9º Edición, Addison Wesley, (México 2011). Pág. 31.

1- **Análisis y definición de requerimientos**: los servicios, restricciones y metas del sistema se definen a partir de las consultas con los usuarios. Entonces, se definen en detalle y sirven como una especificación del sistema.

2- **Diseño del sistema y del *software***: el proceso del diseño del sistema divide los requerimientos en sistemas *hardware* o *software*. Establece una arquitectura completa del sistema. El diseño del *software* identifica y describe las abstracciones fundamentales del sistema *software* y sus relaciones.

3- **Implementación y prueba de unidades**: durante esta etapa, el diseño del *software* se lleva a cabo como un conjunto o unidades de programas. La prueba de unidades implica verificar que cada una cumple su especificación.

4- **Integración y prueba del sistema**: los programas o las unidades individuales de programas se integran y prueban como un sistema completo para asegurar que se cumplan los requerimientos del *software*. Después de las pruebas, el sistema *software* se entrega al cliente.

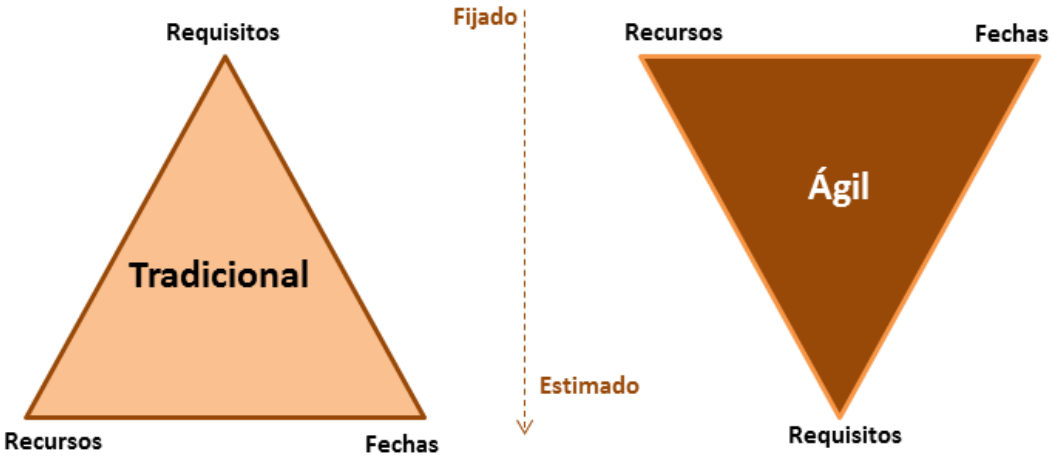
5- **Funcionamiento y mantenimiento**: por lo general (aunque no necesariamente) esta es la fase más larga del ciclo de vida. El sistema se instala y pone en funcionamiento práctico. El mantenimiento implica corregir errores no descubiertos en las etapas anteriores del ciclo de vida, mejorar la implementación de las unidades del sistema y resaltar los servicios del sistema una vez que se descubren nuevos requerimientos.

El resultado de cada fase es uno o más documentos aprobados. La siguiente fase no debe empezar hasta que la fase previa haya finalizado. En la práctica, estas etapas se superponen y proporcionan información a las otras.

3.- Metodología ágil:

Los métodos ágiles proponen[8] un cambio de paradigma ya que se parte de un presupuesto y unas fechas de entrega, y a partir de ahí, se trabaja para implementar la funcionalidad valiosa para el cliente en cada momento.

Trabajando de esta manera, el alcance será flexible.



Para ser competitivos, la mayoría de las veces debemos adaptar el producto a medida que lo vamos construyendo. Los métodos ágiles sugieren formas de construir un producto en los que los cambios impacten lo menos posible en el desarrollo del producto. Lo que proponen estos métodos ágiles son una serie de reglas del juego que se deberían cumplir siempre y en todas las situaciones pero que fomentarán y facilitarán que en cada situación se genere la práctica que realmente sea útil, tanto para el equipo como para el producto.

([8]) ÁLVAREZ GARCÍA, Alonso, DE LAS HERAS DEL DEDO, Rafael, y LASA GÓMEZ Carmen. Op. Cit. Pág. 32.

METODOLOGÍAS TRADICIONALES	**METODOLOGÍAS ÁGILES**
El cliente se relaciona con el equipo, no participa.	El cliente es parte integral del equipo de desarrollo, interactúa constantemente
Se selecciona una arquitectura al comienzo del proyecto.	Se redefine la arquitectura a medida que el proyecto avanza.
El individuo está definido de acuerdo a su posición y rol.	Se valoriza al individuo y sus capacidades.
Se intenta disminuir las posibilidades de cambio.	Se sabe que el cambio ocurrirá y se lo espera.
Destinado a proyectos de todo tipo tamaño.	Generalmente aplicable en proyectos pequeños.
Grandes cantidades de artefactos y elementos de documentación.	Pocos elementos para modelar y documentar.
Grupos grandes y posiblemente distribuidos en distintos lugares.	Grupos pequeños (<10 integrantes) trabajando todos en el mismo espacio físico.
Existe un contrato prefijado.	No hay contrato tradicional, y si existe es flexible.
Proceso mucho más controlado con numerosas políticas y normas.	Proceso mucho menos controlado, con pocos principios.


TRADICIONAL
Planteamiento
Análisis
Diseño
Programación
Pruebas
Puesta en marcha
AGILE
Planteamiento
Requerimientos priorizados
Iteración
Puesta en marcha

CAPITULO III

GESTION AGIL

Sumario: 1.- Gestión de proyectos. 2.- El Manifiesto ágil. 3.- Contratos ágiles.-

1.- Gestión de proyectos:

Proyecto

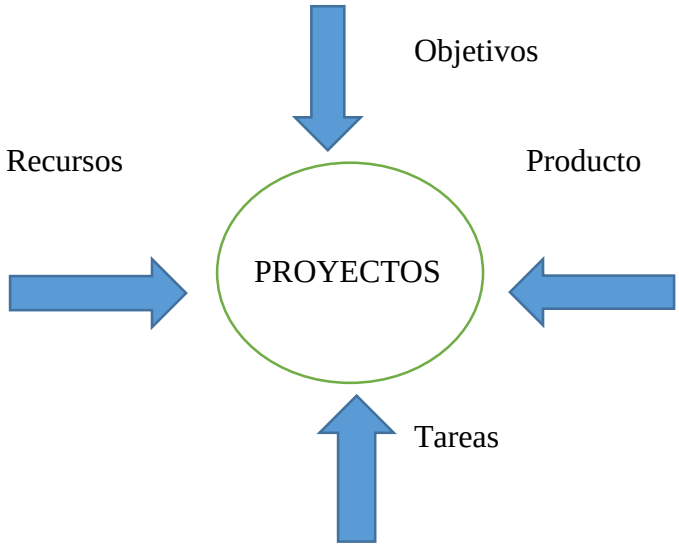
Proyecto[9] es un conjunto de actividades relacionadas que utilizan recursos para cumplir con un objetivo deseado dentro de un plazo de tiempo delimitado. Presenta las siguientes características:

- Fecha de inicio y fin.
- Definición de tareas y calendario.
- Sucesión de actividades.
- Necesidad de recursos.
- Producción de un resultado único.

([9]) PRIOLO, Sebastián, op. cit. Pág. 92.

Un proyecto es exitoso cuando[10]:

- Se termina dentro del tiempo pactado, con el presupuesto comprometido y con la funcionalidad deseada.
- El cliente se siente satisfecho con el producto.
- El producto ofrece las ventajas comerciales esperadas.
- El equipo del proyecto cree que su participación fue valiosa.
- El proyecto eleva el nivel de conocimiento y permite mejorar a futuro.



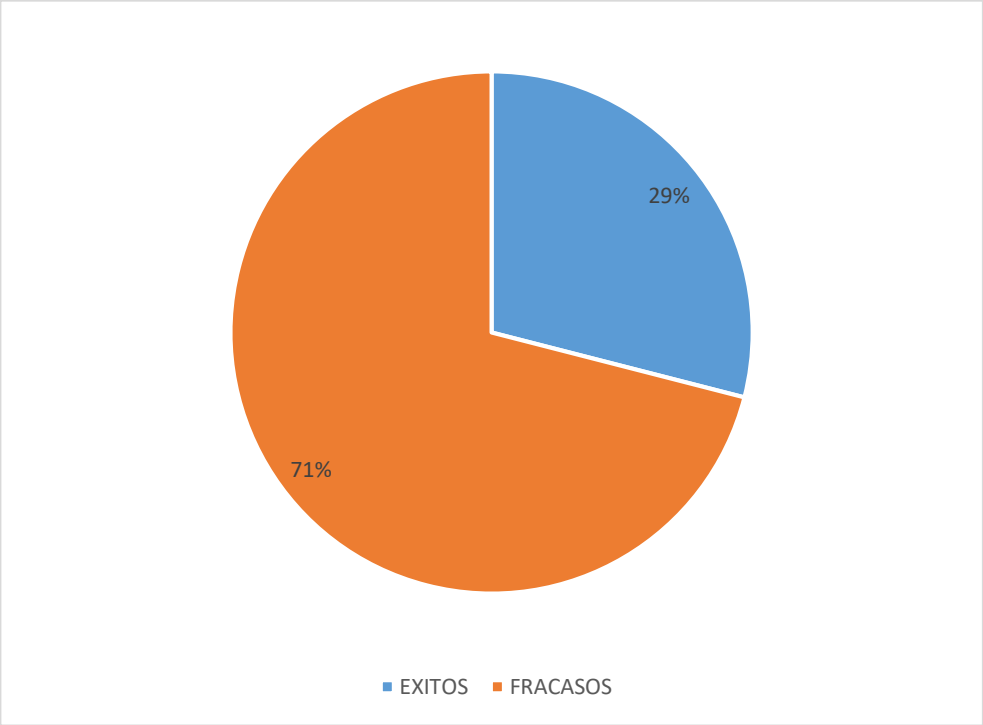
Los factores que influyen al fracaso son múltiples e infinitos, pudiendo resumirse en los siguientes:

- Falta de personal capacitado.
- Error en la selección de recursos.
- Falta de capacidad administrativa.
- Falta de comunicación.
- Incompleto análisis de requisitos.
- Problemas técnicos de diseño.

([10]) Ibídem.

- Fallas en el planeamiento.
- Fallas en la ejecución de tareas

Porcentaje de éxitos y fracasos en los proyectos de desarrollo de software



Gestión de proyectos

A medida que los proyectos crecieron y se profesionalizaron, fue necesario crear un conjunto de disciplinas relacionadas para planificar, controlar y asegurar el éxito.

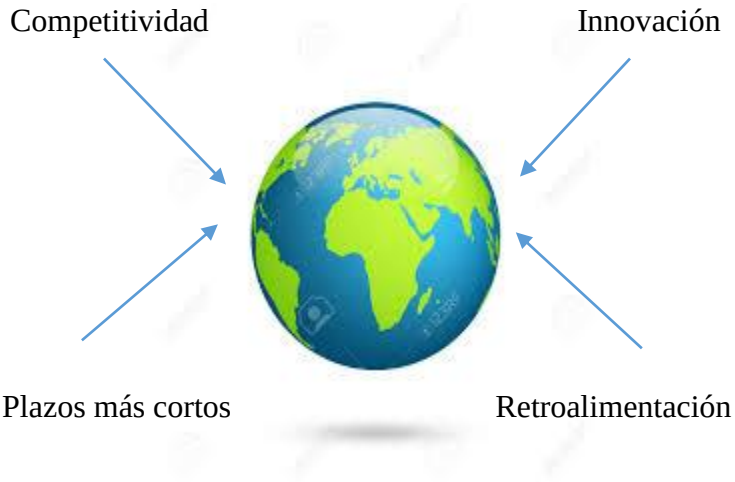
La gestión de proyectos es la disciplina encargada de organizar y administrar recursos para poder llevar a cabo los proyectos cumpliendo con las restricciones pactadas. La administración de proyectos es una tarea compleja con gran cantidad de variables originadas principalmente por el resultado, único producto del proyecto.[11]

([11]) Ibídem. Pág 95.

Nuevo escenario

La posibilidad de nuevos productos y la velocidad del cambio junto con los adelantos tecnológicos hicieron necesario que las metodologías de gestión pasaran a ser más dinámicas. Las diferencias entre las predictivas y las modernas son notorias. En este nuevo escenario competitivo, las variables más importantes son[12]:

- Retroalimentación del producto y el entorno.
- Mayor innovación del producto.
- Reducción de los tiempos.
- Salida inmediata al mercado.
- Los productos deben evolucionar, no están terminados.



La figura representa como las nuevas tecnologías y la velocidad condiciona los tipos de desarrollo.

Gestión ágil de proyectos

La Gestión Ágil de proyecto[13] intenta convivir con la idea, y a la vez fomentarla, de que no existen productos finales. Todos los productos son versiones beta en constante mejora. La anticipación y la adaptación se rigen entonces como los pilares

([12]) Ibídem. Pág. 96.
([13]) Ibídem. Pág. 97.

de este nuevo enfoque. Por lo tanto la gestión ágil intenta responder ante los nuevos valores de la industria: **valor, tiempo, fiabilidad y agilidad.**

2.- El Manifiesto ágil:

El "Manifiesto Ágil"[14], publicado en 2001, dónde un conjunto de autores y personas relevantes del mundo del desarrollo *software* se reunieron para plasmar en unos pocos puntos lo que eran las ideas y el sentir general de la industria. En principio surgido como reacción a la forma de desarrollar proyectos del momento, este manifiesto ha trascendido este propósito y se ha convertido en la piedra fundacional de una nueva forma de trabajar. Esta forma ágil de desarrollar proyectos se fundamenta en cuatro puntos:

- Valorar a individuos y sus iteraciones, frente procesos y herramientas. Aunque todas las ayudas para desarrollar un trabajo son importantes, nada sustituye a las personas, a las que hay que dar toda la importancia y poner en primer plano.
- Valorar más el *software* (producto) que funciona, que una documentación exhaustiva, el foco debe estar siempre en lo que queremos construir y los demás es secundario.
- Valorar más la colaboración con el cliente que la negociación de un contrato. La forma productiva de sacar adelante un trabajo es establecer un marco de confianza y colaboración con quien nos lo encarga.
- Valorar más la respuesta al cambio que el seguimiento de un plan. Se trata de apreciar la incertidumbre como un componente básico del trabajo, por lo que la adaptación y la flexibilidad se convierten en virtudes y no en amenazas.

Estos cuatro puntos están acompañados de los siguientes principios[15], acaban de definir la forma de trabajar de acuerdo con los medios ágiles:

([14]) Manifiesto for Agile Software Development, en Internet: www.agilemanifesto.org
([15]) Tomados literalmente de la traducción española el manifiesto ágil.

1- Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.
2- Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos ágiles aprovechan el cambio para proporcionar una ventaja competitiva al cliente.
3- Entregamos software funcional frecuentemente, entre dos semanas y dos meses, con preferencia al período de tiempo más corto posible.
4- Los responsables del negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.
5- Los proyectos se desarrollan en torno a individuos motivados.
6- El método más eficiente y efectivo es la conversación cara a cara
7- El software funcionando es la medida principal de progreso.
8- Los procesos ágiles promueven el desarrollo sostenible.
9- La atención continua a la excelencia técnica y al buen diseño mejora la agilidad.
10- La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.
11- Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados.
12- A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para a continuación ajustar y perfeccionar su comportamiento en consecuencia.

Todo ello se puede resumir en orientación al cliente y al producto, flexibilidad, dar relevancia a las personas y a la comunicación, colaboración, velocidad, auto-organización, calidad, simplicidad, y mejora continua.

Ágil en la práctica

La aparición de metodologías ágiles es una reacción a la falta de respuesta a los problemas históricos del desarrollo de proyectos. La incertidumbre es uno de los grandes desafíos en el desarrollo de proyectos, según el autor Álvarez García Alonso

se ha tratado de combatirla controlando el proceso, planificando pormenorizadamente, estimando y diseñando cada paso.

El manifiesto ágil recoge una serie de ideas que describen una nueva forma de trabajar, dejando atrás la orientación inflexible y determinista de los proyectos convencionales. Ahora, la incertidumbre no es una amenaza, es una dimensión en la que nos basamos y que aceptamos. Es una corriente que impulsa en lugar de luchar contra ella.

3.- Contratos ágiles:

Si los métodos Ágiles establecen una nueva forma de trabajar y de relación con los clientes, una de las expresiones de esa relación, los contratos, también se verá afectada. Se abandona la orientación tradicional en la que el trabajo está perfectamente definido antes de empezar, por otra en que se orienta hacia el cambio y facilita la introducción continúa de nuevos requisitos.

Las expresión más extrema del contrato ágil se resume en las frase: *"money for nothing, change for free"*[16] (dinero a cambio de nada, modificaciones gratis) que no deja de ser una forma de definir la **confianza mutua,** la que tiene el cliente en el equipo al pagar sin un contrato formal que defina detalladamente lo que va a recibir al final del proceso, y la del equipo, que realiza los cambios que se le soliciten sin exigir una evaluación económica a cada paso.

Es una filosofía completamente distinta de la de un contrato convencional. Los contratos son más una herramienta de protección que la definición de una relación de confianza.

Al final se corre el riesgo de pagar más de lo que se pensaba y recibir menos de lo que se esperaba.

Lo cierto es que un contrato debería ser un acuerdo y potenciar que ambas partes salgan beneficiadas, más que proteger a una frente la otra. Un contrato debería reflejar una colaboración y no ser una entre las partes.

([16]) Acuñada por Jeff Sutherland, uno de los firmantes del manifiesto ágil, y una de las grandes referencia mundiales en métodos ágiles.

El acuerdo se fija ante todo en tres dimensiones[17]: **alcance** (lo que incluye calidad), **costes y plazos;** y como mucho se establecen mecanismos para que la variación de una no afecte a las demás. Sin embargo, en un contrato ágil, un contrato debe afectar a las restantes dimensiones: si cambia el alcance debe variar el tiempo y/o el coste, es inevitable, o la calidad se verá comprometida.

Trabaja en bloques pequeños y ciclos cortos, propio de los proyectos ágiles, ayuda a reducir la complejidad, y permite asumir cambios como algo natural.

Otra premisa importante de los contratos ágiles es asumir que los riesgos son compartidos, para evitar que el contrato se convierta en un arma arrojadiza.

En un contrato convencional, que busca cubrir a las partes, se trata de contar con unos requisitos cerrados de antemano y con una aceptación única final.

Lo que se busca con estos contratos es llegar a trabajar como socios, no en una relación cliente-proveedor.

Hay bastantes modelos[18] de contratos ágiles, que recoge todo el rango de graduación de confianza entre las partes:

- Fijación de precio y alcance, pero dejando abierta el resto de las variables. En realidad el riesgo pasa al proveedor y a su capacidad de estimar.
- Tiempo y materiales: que supone pagar por unos recursos en un tiempo, aunque dejando abierta la funcionalidad. En este caso, es el cliente el que asume todo el riesgo, ya que no hay compromiso de alcance.
- Desarrollo en fases: Una aproximación cooperativa en la que se establecen *releases* que limitan el riesgo mientras el resto de los factores se acuerdan entre las partes. Puede complementarse con un sistema de penalizaciones
- Beneficio fijo: Se estable un beneficio y un coste fijo. Si se obtiene el alcance en fechas, el proveedor recibirá ese beneficio; si no es así,

---

([17]) Contratos ágiles, en Internet, www.proyectosagiles.org
([18]) Ibídem.

el cliente seguirá pagando el esfuerzo, pero esta vez no habrá beneficio posible para el proveedor.

- *Money for nothing, change for free*: La expresión más extrema de los contratos agiles. El cliente paga por el esfuerzo, pero el compromiso del proveedor es total. Aparentemente el cliente paga por un alcance que no está completamente cerrado, peor el proveedor se entrega al proyecto y asume todo el esfuerzo preciso para completar todo el alcance comprometido en cada iteración, así como respaldar la calidad del producto.

CAPITULO IV

TIPOS DE METODOLOGIAS AGILES

Sumario: 1.- Introducción. 2.- Lean Software Development. 3.- Kanban. 4.- Pragmatic Programming. 5.- Feature Driven Development. 6.- Dynamic Systems Development Method. 6.- Programación Extrema o Extreme Programming.-

1.-Introducción:

Todos los métodos ágiles tienen en común muchas de sus características mientras que unos pocos aspectos los hacen diferentes. La idea es que en cada situación se elija el método que mejor se adapte a su producto. Pero, ¿Qué hace que un método sea ágil?, es decir, ¿Qué es lo que tiene en común estos métodos?

Todos ellos consideran la colaboración un elemento clave. Tanto las personas que están construyendo el producto como el cliente deben trabajar en constante comunicación y sentirse miembros de un gran equipo. Con este enfoque, la comunicación constante y a todos los niveles es crucial para crear el producto con una calidad excelente y que cumpla exactamente las necesidades del cliente, evitando sorpresas a todos los implicados. Por otro lado, un método es ágil si permite construir

un producto de forma incremental, es decir crear algo muy sencillo inicialmente y que vaya siendo enriquecido y completado de forma progresiva. No se construirán trozos de productos por separado que luego tendremos que hacer encajar al final como un rompecabezas, sino que se construye contemplando la totalidad desde el principio.

Otro factor común de estos métodos ágiles es su sencillez. Sus reglas son sencillas y de sentido común pero eso sí, es necesaria la experiencia y profesionalidad para obtener el máximo beneficio de ellas.

Para que un método pueda considerarse ágil, debe ser adaptativo, es decir, se debe construir contemplando siempre la posibilidad de introducir modificaciones y cambios en cualquier etapa de su construcción.

2.- Lean Software Development:

Es un método ágil centrado en la estrategia y su origen está en la empresa de la manufacturación y la posterior adaptación al desarrollo de Software.

Este método tiene tres objetivos principales[19] que son: reducir drásticamente el tiempo de entrega de un producto, reducir su precio y reducir también el número de defectos o *bugs*.

Los principios en los que se basa este método para conseguir sus objetivos son los siguientes:

- Eliminar todo lo que no aporte valor: No perder el foco, es crucial no hacer algo que no sea específicamente lo que se espera que se haga, es necesario simplificar toda la burocracia, gestión, optimizar los procesos y evitar la falta de información así como las interrupciones al equipo de trabajo. Todos los implicados en la construcción de un producto deben buscar los factores y procesos que reduzcan el valor del mismo para analizar su origen y eliminarlo. Si no pudieran

([19]) POPPENDIECK, Mary and Tom, Lean Software Development An Agile Toolkit, Series Editors. 2003.

eliminarse el impedimento, al menos debe tratar de reducir su impacto.

- Optimizar el todo: El cliente necesita un todo. No le aporta mucha información ver pequeños trozos de lo que se espera sin saber cómo va a encajar el puzle final. Necesita, desde el principio, ir viendo una foto global de lo que va a recibir.
- Construir con calidad: Deben cubrirse todo tipo de pruebas de forma que los defectos se corrijan lo antes posible con una construcción dirigida por las pruebas constantes.
- Aprender constantemente: La clave está en ir aprendiendo y entendiendo lo que se necesita a medida que se construye, ya que no podemos adivinar el futuro. En Lean se utiliza con frecuencia la expresión "último momento de responsabilidad", que lo que pretende transmitir es que se debe aprender e investigar todo lo posible antes de tomar una decisión que afecte al producto pero esa decisión debe tomarse en el momento justo y no más tarde. En definitiva, huir del análisis-parálisis pero también de la toma de decisiones de forma precipitada.
- Reaccionar rápido: Para reaccionar rápido es necesario trabajar con iteraciones cortas de forma que los comentarios del cliente sean frecuentes y se cubren sus expectativas y necesidades en cada momento.
- La mejora continua: El foco de la mejora debe centrarse en las personas y en los procesos que hacen posible construir un producto y no en mejorar exclusiva y directamente el producto en sí. De esta forma se mejorará el producto actual y el sistema estará listo para poder crear otros productos con éxito en el futuro.
- Cuidar al equipo de trabajo: Un equipo de trabajo debe estar motivado y esto se consigue proporcionándole cierto grado de autonomía para poder tomar decisiones con sentido, ofreciendo a

cada persona la posibilidad de aprender y mejorar de manera permanente y por último, procurar que sientan que su trabajo es valioso en todo momento.

3.- Kanban:

Kanban[20] es una palabra de origen japonés que significa "Tarjetas visuales." Aplicando este método se consigue mostrar permanentemente y de forma muy visual el estado del proyecto a todos los implicados.

Kanban es un método tremendamente útil para gestionar los productos cuyos requisitos cambian constantemente, bien porque aparezcan nuevas necesidades o bien porque su prioridad varíe. Este método también es útil en los casos en los que las planificaciones o estimaciones de un equipo se alarguen demasiado y dejen de ser productivas así como cuando no se pueda comprometer un equipo a trabajar con iteraciones de duración fija y predeterminada por el motivo que sea.

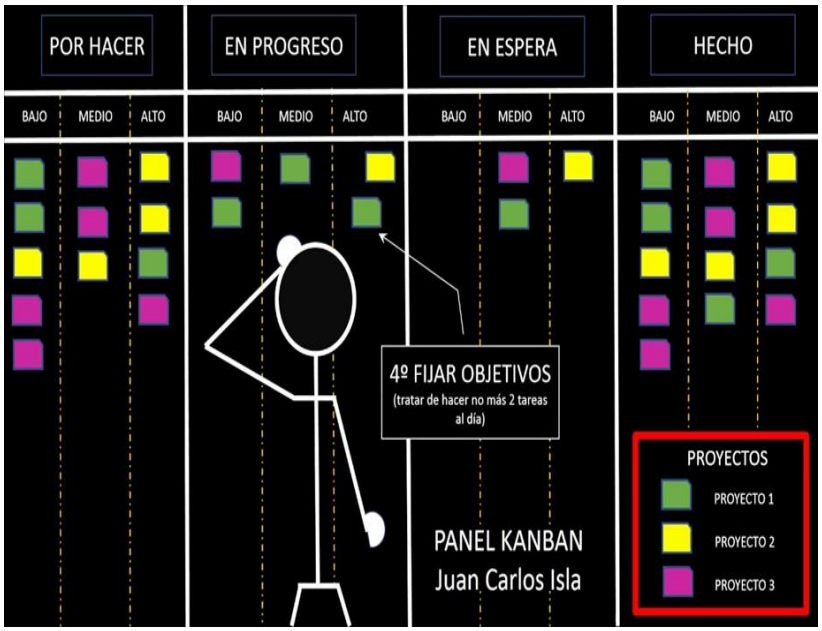
Pasos que se deben seguir para trabajar con *Kanban*

- Visualizar el flujo de todo el trabajo: En un panel debe estar representado todo el flujo del trabajo que hay que realizar en el proyecto, desde el principio hasta el último momento. Para que el panel sea útil, represente las columnas que permitan mostrar en qué estado del flujo está cada ítem en cada momento. La primera columna representa el *Backlog* del producto, es decir, la lista priorizada de las necesidades. Divida el trabajo en ítems pequeños y escriba cada uno en una tarjeta. Priorícelos y colóquelos ordenados en la primera columna del tablero.
- Limite el trabajo en curso: Es imprescindible poner un límite al número de ítems permitidos en cada columna y de esta forma evitar colapsos, cuellos de botella y eliminar cuanto antes los

(20) KNIBERG, Henrik and SKARING Mattias, Kanban and Scrum Making the most of Both, Enterprise Software Development Series (2010)

impedimentos que surjan y que impidan trabajar con un ritmo sostenible.

- Mida el tiempo empleado en completar un ciclo completo: Calcule el tiempo que se emplea desde que se empieza a trabajar con un ítem o tarea hasta que se da por cerrado o terminado y trate de buscar la manera de disminuir ese tiempo. Con *Kanban* los implicados en la creación de un producto tienen acceso a toda la información del mismo y al estado de cada una de sus partes en cada momento. El grado de compromiso aumenta notablemente ya que todos pueden participar en la mejora directa e inmediata del proceso. Las características de *Kanban* hacen que pueda utilizarse para organizar y gestionar el trabajo en cualquier campo y no exclusivamente para proyectos de desarrollo de *software*.



4.- Pragmatic Programming:

Es una serie de mejores prácticas de programación englobadas dentro de la

filosofía de trabajo ágil. Estas prácticas fueron publicadas en el año 2000 por Andrew Hunt y David Thomas [21], ambos firmantes del manifiesto ágil.

La filosofía de estas prácticas podría resumirse muy brevemente en los siguientes puntos:

- Cuando se comprometa a realizar un trabajo, debe asumir la responsabilidad de hacerlo lo mejor que pueda.
- Construya con un buen diseño y cree códigos de calidad.
- Si considera que es necesario realizar algún cambio, fomente que se realicen y forme parte activa de este proceso de cambio.
- Es crucial construir un producto que sea satisfactorio para el cliente pero es importante también saber detectar en qué momento es necesario dejar de construirlo y no seguir durante un tiempo indefinido añadiendo funcionalidad que no se ha solicitado.
- El aprendizaje continuo debe ser una constante para cualquier persona implicada en la construcción del producto.
- Aunque está programado, la comunicación con los demás es clave y por ello debe tratar de mejorar sus habilidades de comunicación constantemente.

5.- Feature Driven Development:

El desarrollo orientado a la funcionalidad es un método ágil concebido para el desarrollo de sistemas informáticos. Este método no pretende cubrir todo el proceso de desarrollo de un producto sino que se centra en las fases de diseño y construcción. Sus puntos claves son el trabajo en iteraciones, control continuo, la calidad de lo creado y entregas frecuentes para poder realizar un seguimiento continuo en la colaboración con el cliente y poder así incorporar sus necesidades en el producto con frecuencia.

([21]) HUNT Andrew y THOMAS David, The Pragmatic Programmer. Addison Wesley. (2000)

FDD[22] propone seguir unos pasos secuenciales:

- Crear un modelo global: Tener un conocimiento del alcance, los requisitos y del contexto del sistema en el que se va a construir el producto. Al finalizar esta etapa, los expertos proporcionarán a los miembros del equipo una descripción general del sistema.
- Crear una lista de funcionalidades: Es necesario plasmar este modelo global, junto con los demás requisitos del conjunto del sistema, en una única lista de necesidades o funcionalidades a cubrir.
- Planear por funcionalidades: A la hora de hacer un plan a alto nivel, debe siempre tenerse en cuenta la prioridad de las funcionalidades y las dependencias entre ellas.
- Diseñar y construir por funcionalidad: Se deben diseñar y construir las funcionalidades de forma iterativa. En cada iteración se seleccionará un conjunto de funcionalidades y estos ciclos deben oscilar entre algunos días y dos semanas.

6.- Dynamic Systems Development Method:

La forma de trabajar que propone este método para el ciclo de vida de un proyecto, está estructurada en 5 fases de las cuales, las dos primeras se realizan una sola vez y las tres últimas, se realizan de forma iterativa e incremental. Estas etapas son: estudio de la viabilidad del proyecto, estudio del negocio, iteraciones del modelo funcional, iteraciones para la creación del diseño y desarrollo del producto y finalmente, iteraciones para la implementación.

Tal y como explica *Dean Leffingwell*[23] la filosofía de DSDM es sencilla:

- El desarrollo de un producto debe entenderse como un trabajo en equipo ya que para que tenga éxito debe combinarse el conocimiento

---

([22]) PALMER Stephen and FELSING John, A Practical Guide to Feature-Driven Development, The Coad Series (2002)

([23]) LEFFINGWELL Dean, Scaling Software Agility Best Practices for Large Enterprises. Series Editors. (2007)

de las necesidades del negocio que tienen los clientes con el perfil técnico de los desarrolladores.

- La calidad debe contemplarse desde dos puntos de vista: la solidez técnica y la facilidad de uso.
- El desarrollo puede y debe hacerse de forma incremental.
- Debe trabajarse inicialmente en las funcionalidades que aporten mayor valor al negocio y por lo tanto, los recursos deben invertirse en el desarrollo de las mismas.

Uno de los principales principios en los que se basa DSDM es la creencia de que un requisito no se puede prefijar completamente al inicio del desarrollo, y cuando así se hace, solo una parte de éste es realmente valioso para el cliente.

Otro tema revolucionario que propuso entonces DSDM fue dar la vuelta al enfoque de los métodos más tradicionales en los que se fijaban unos requisitos y en función de ellos, se estimaban tanto los recursos como la fecha de entrega. Lo que DSDM propone es fijar los recursos destinados a un producto y la fecha de entrega y hacer una estimación de la funcionalidad que se entregará. En definitiva, se sabrá cuándo se va a entregar algo valioso al cliente pero a priori no se sabrá qué es exactamente lo que se va a entregar.

7.- Programación Extrema o Extreme Programming:

Tal y como lo define Kent Beck[24], *Extreme Programming* (XP) es un método ágil para el desarrollo de software muy útil a la hora de abordar proyectos con requisitos vagos o cambiantes. XP es especialmente útil si se aplica a equipos de desarrollos pequeños o medianos.

Es un método adaptativo, es decir, se ajusta muy bien a los cambios. Propone desarrollar el código de forma que su diseño, arquitectura y codificación permitan incorporar modificaciones y añadir una funcionalidad nueva sin demasiado impacto en la calidad del mismo.

([24]) BECK, Kent, Extreme Programming Explained, Addison-Wesley. (1999)

Por otro lado, XP es un método muy orientado hacia las personas, tanto a las que están creando el producto como a los clientes y usuarios finales.

Desarrollando como propone XP, se obtiene rápidamente resultados. Al trabajar con pequeñas iteraciones, se puede obtener con frecuencia comentarios del cliente, lo que tiene como resultado que el producto final cubra ampliamente sus expectativas y necesidades.

Para XP las pruebas son la base de la construcción y propone que sean los desarrolladores los que escriban las pruebas a medida que van construyendo el código y se realice una integración continua, de forma que el *software* creado tenga una gran estabilidad. Las pruebas automáticas se realizan de forma constante para poder detectar los fallos rápidamente. Cuanto antes se detecte un problema, antes podrá resolverse sin que las consecuencias y el impacto sean mayores.

Antes de cada iteración se planifica el trabajo que va a realizarse y a continuación se realizan de forma simultánea el análisis, el desarrollo, el diseño y las pruebas del código.

CAPITULO V

SCRUM

Sumario: 1.- Definición. 2.- Historia. 3.- Metodologías ágiles y Scrum. 4.- Algo más sobre Scrum. 5.- Principios. 6.- Valores. 7.- Ventajas. 8.- Roles.-

1.- Definición:

Scrum es un proceso para desarrollo que apunta al trabajo en equipo y a la aplicación de mejores prácticas para conseguir los resultados esperados de un proyecto.

Las principales ideas de *Scrum* es la realización de entregas periódicas, la confección de equipos altamente capacitados y la relación con el cliente, a diferencia de otras metodologías, que se centran solamente en la parte de *software*, es posible afirmar que ésta permite el desarrollo de productos de toda clase. Sin embargo, por su origen y prácticas, se la ha utilizado mayoritariamente en la industria del *software*, aunque cada vez es mayor la cantidad de proyectos de otras áreas que la están incorporando.

Muchos teóricos definen a *Scrum* como *Framework* o marco de trabajo, en lugar de metodología. Esta diferencia parte del hecho de que no dictamina lo que se

debe realizar, sino que da pautas generales que deben ser comprendidas y aplicadas en mayor o menor medida por los encargados del proceso.

Scrum es poco prescriptivo, pero lógicamente se pueden añadir los roles, artefactos o reuniones que sean necesarios. Eso sí, tal y como recomienda Henrik Kniberg[25] es mejor estar seguro de que se necesita algo nuevo antes de incorporarlo, basándonos siempre en la filosofía general Agile de hacer las cosas de la manera más sencilla posible. En caso de duda, comience por lo mínimo y vaya añadiendo lo que realmente se necesite en cada momento.

2.- Historia:

Entre los años 1985 y 1986, los investigadores de origen japonés Hirotaka Tkeuchi e Ikujijo Nonaka seleccionaron una gran cantidad de empresas en Estados Unidos y en Japón y observaron la cantidad de ingresos que éstas obtenían por nuevos productos e innovación. Sus artículos destacaban que existían organizaciones que, a pesar de moverse en el mismo ambiente cambiante de sus rivales, obtenían mejores productos en tiempos reducidos. Algunos de elementos estudiados presentados como ejemplo era la fotocopiadora Fuji- Xerox, la copiadora personal canon, el automóvil de 1200cc de Honda y la computadora personal NEC PC 8000.

Las empresas observadas tenían, como diferencia principal en el ciclo de desarrollo, que sus fases de construcción se solapaban. Además, en lugar de tener especialistas en diferentes equipos, muchas de las tareas las llevaba a cabo un solo grupo y en el mismo lugar físico. A esto se le denomino campos de *Scrum* (por su similitud con el rugby).

A pesar de estos estudios, pasaron varios años hasta que, en la década de los 90, Ken Schwaber y Jeff Sutherland comenzaron a implementar métodos similares a los propuestos, siendo este último quien la bautizara como *Scrum*. Ambos autores

---

([25]) KNIBERG, Henrik and SKARING Mattias, Kanban and Scrum Making the most of Both, Enterprise Software Development Series (2010)

presentaron, en 1995 durante las conferencias OOSPLA, los primeros artículos que describían esta metodología.

A partir de ese momento, su crecimiento se produjo no solamente por un gran interés teórico sino porque grandes empresas comenzaron a desarrollar productos con equipos pequeños bajo estas ideas.

La metodología tomo impulso final en el año 2001 cuando Schwaber y Mike Beedle presentaron el libro Desarrollo Ágil de *Software* con *Scrum*.

En la actualidad, existen miles de profesionales formados bajo estas prácticas y se agrupan bajo una organización sin fines de lucro que se denomina *Scrum Alliance*. En su sitio web se puede acceder a artículos, recursos, entrenamiento y certificación, noticias, información sobre eventos en todo el mundo y las condiciones para registrarse y pertenecer a la comunidad.

3.- Metodologías ágiles y Scrum:

Los métodos ágiles se han consolidado y se aplican en muchos campos, más allá de su nacimiento en el mundo del desarrollo *software*. Algunas de ellas se centran en las formas productivas de desarrollar aplicaciones informáticas, mientras que otras buscan reflejar específicamente procesos de la empresa. La influencia de los métodos ágiles es tal que estos están afectando a la forma de contratar proyectos, a través de los contratos ágiles, que reflejan la flexibilidad y capacidad de adaptación aplicada a las relaciones formales entre empresas.

De entre todos los métodos ágiles, se destaca *Scrum* por su difusión y aceptación. Nacido antes del manifiesto ágil, se ha consolidado como el método más utilizado, y ha demostrado ser aplicable en toda clase de proyectos.

4.- Algo más sobre Scrum:

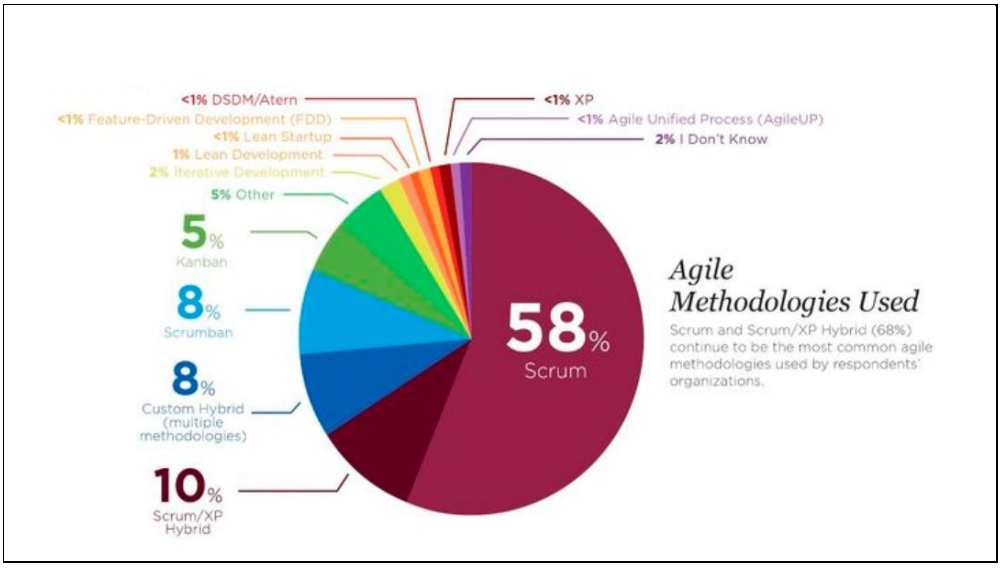
Scrum (o melé en castellano) es la jugada de deporte del rugby que se utiliza para reincorporar al partido una pelota que se había quedado fuera de juego. En esta jugada el equipo actúa como una unidad para desplazar a los jugadores del equipo contrario.

La comparación con el rugby se realiza por la similitud entre la forma de jugar en este deporte y la necesidad de que se modifique la manera de trabajar entre los equipos de desarrollo ya que el deporte del rugby es altamente coordinado, colaborativo y reactivo.

Scrum está siendo aplicado actualmente en diferentes áreas de numerosas empresas como por ejemplo telefónica, Google Xerox, Vodafone, Siemens, IBM, Nokia, Toyota, Microsoft, Adobe, Amazon, Ericson y la lista continúa.

Scrum es tremendamente eficaz aplicado al desarrollo de aplicaciones *software* pero no es exclusivamente en este campo donde ha demostrado su utilidad. *Scrum* se ha incorporado también con éxito en todo tipo de proyectos como por ejemplo para desarrollo de videojuegos, páginas web o teléfonos móviles y ha aportado un enorme valor aplicado para la organización de comités ejecutivos, empresas de ventas y marketing, logística militar y en otros muchos casos.

El siguiente gráfico representa cómo se están aplicando y adoptando los diferentes métodos ágiles en el mercado.



Ken Schwaber lo compara con el ajedrez en el sentido de que en ambos casos las reglas del juego son sencillas pero para ser un maestro es necesario practicar,

aprender y mejorar continuamente. *Scrum* o el ajedrez no fracasan o triunfan. Simplemente marcan reglas que hay que seguir para jugar. En el caso del ajedrez con la práctica se puede llegar a ser un gran maestro. En el caso de *Scrum*, ese maestro será el que consiga que la organización funcione con éxito, que los clientes lo valoren y los usuarios lo aprecien y que sea respetado por la competencia.

5.- Principios:

Scrum se basa en los siguientes principios[26]:

Inspección y Adaptabilidad: En *Scrum* se trabaja en iteraciones llamadas *Sprint*, que tiene una duración de entre 1 y 4 semanas. Cada iteración termina con un producto entregable. Al finalizar cada iteración, este producto se muestra al cliente para que opine sobre él. A continuación el equipo se reunirá para analizar la manera en que está trabajando. Uniendo los dos puntos de vista, "el que" sea hecho y "el cómo" se está construyendo, se aprenderá con la experiencia y podremos mejorar iteración tras iteración.

([26]) ÁLVAREZ GARCÍA, Alonso, DE LAS HERAS DEL DEDO, Rafael, y LASA GÓMEZ Carmen. Op. Cit. Pág. 39.

Auto-organización y Colaboración: El equipo se gestiona y organiza así mismo. Este nivel de libertad implica asumir una responsabilidad y un gran nivel de compromiso por parte de todos. Los líderes y clientes colaborarán igualmente con el equipo de desarrollo en todo momento facilitando su trabajo, resolviendo dudas y eliminando posibles impedimentos.

Priorización: Es crucial no perder tiempo y dinero en algo que no interesa inmediatamente para el producto. Para ello es necesario tener unos requisitos perfectamente priorizados reflejando el valor del negocio.

Mantener un latido: Es tremendamente valioso mantener un ritmo que dirija el desarrollo. Este latido marcará la pauta del trabajo y ayudará a los equipos a optimizar su trabajo. El tener un ritmo fijo de trabajo, tanto a nivel del día a día como a nivel de *Sprint*, permite que el equipo sea predecible ya que éste aprenderá a estimar la cantidad de trabajo a la que puede comprometerse. El mantener un latido ayuda a todos a centrarse en crear el producto y tener muy estables las fechas clave de una iteración.

Una de las principales características de *Scrum* es que en cada iteración todas las etapas de creación de un producto se solapan, es decir, en cada Sprint se realiza la planificación, análisis, creación y comprobación de lo que se va a entregar al final del mismo.

6.- Valores:

Los valores de Scrum son los siguientes[27]**:**

Mejora continua: Los miembros de un equipo de Scrum deben procurar identificar continuamente puntos de mejora y hacer lo posible por aplicarlos para realizar su trabajo de forma más productiva y con mayor calidad.

Calidad: El objetivo último de todos nuestros esfuerzos por mejorar la forma en la que trabajamos y los productos que construimos.

Time-boxing: Significa aprovechar y no perder tiempo.

([27]) Ibidem.

Responsabilidad: Una organización en la que prima la auto-organización sólo funciona con un grado de responsabilidad superior entre los miembros del equipo. En *Scrum* la responsabilidad es compartida y afecta a todos por igual.

Multidisciplinar: El equipo de trabajo debe ser capaz de realizar todas las tareas necesarias del proyecto, en *Scrum* se espera que cada uno pueda ser autónomo y realizar todos los trabajos precisos sin contar con contribuciones externas.

Flexibilidad: *Scrum* es una forma de dejar de lado la idea de que un proyecto parte de una descripción estática de lo que quiere el cliente, en su lugar *Scrum* reconoce la realidad y el cambio, se define en torno a la idea de que los requisitos son flexibles y variables.

Ritmo (latido): *Scrum* favorece que el equipo trabaje a un ritmo determinado. Alcanzar ese ritmo será la base para convertirse en un equipo maduro, capaz de funcionar de forma sincronizada, y de ofrecer estimaciones de alcance y fechas a sus clientes.

Compromiso: La confianza y la autonomía que otorga a todos los participantes requieren que su actitud hacia el proyecto sea activa y comprometida.

Simplicidad: Es un rasgo de calidad y un valor añadido que se da al producto realizado, y que facilitará la labor de los que tengan que trabajar en un futuro con él.

Respeto: *Scrum* se centra prioritariamente en las personas, y las relaciones personales no son fructíferas si no hay un respeto mutuo entre los participantes.

Coraje: Los participantes en un proyecto bajo *Scrum* deben afrontar decisiones comprometidas, tomar iniciativas, actuar en función de un objetivo común.

Foco: *Scrum* no permite distracciones.

Predictibilidad: El equipo debe adoptar un ritmo determinado, trabajar de una forma ordenada y disciplinada, y todo con el objetivo de acabar siendo predecible y que cantidad de trabajo puede realizarse en un período determinado.

Personas: *Scrum* se centra sobre todo en las personas participantes o interesadas, en favorecer el flujo de comunicación entre ellas para lograr unas relaciones ricas y fluidas.

Ésta es la lista de valores de los autores, pero no es la única lista posible. Pueden encontrarse además otras listas más resumidas o más extensas y con algunos otros valores que no se contemplan aquí porque no sean importantes.

7.- Ventajas:

Scrum presenta ciertas ventajas frente a otras metodologías[28]:

Resultados anticipados: Gracias a las entregas periódicas con funcionalidad, el cliente puede conocer mejor el estado del proyecto y afirmar sus requisitos o modificarlos sin impactos significativos.

Gestión del ROI (retorno de la inversión): En cada iteración el cliente dispone de un producto con mayor funcionalidad. En base a esto, planteará el camino a seguir. Cuando el costo de la funcionalidad pendiente sea superior que los beneficios que le aportará, se puede decidir la finalización del proyecto. Se obtiene una mayor gestión y un mejor control sobre el retorno de la inversión.

Simpleza: La metodología es muy simple y puede ser aprendida en minutos. Sin embargo, no debemos engañarnos ya que los profesionales que trabajan con *Scrum* conocen todos sus pormenores y alcanzar este nivel de conocimiento lleva mucho tiempo y experiencia de campo.

Normas claras: Generalmente, al tener pocas referencias técnicas, los equipos que utilizan *Scrum* se familiarizan rápidamente con la metodología y con los límites de sus funciones.

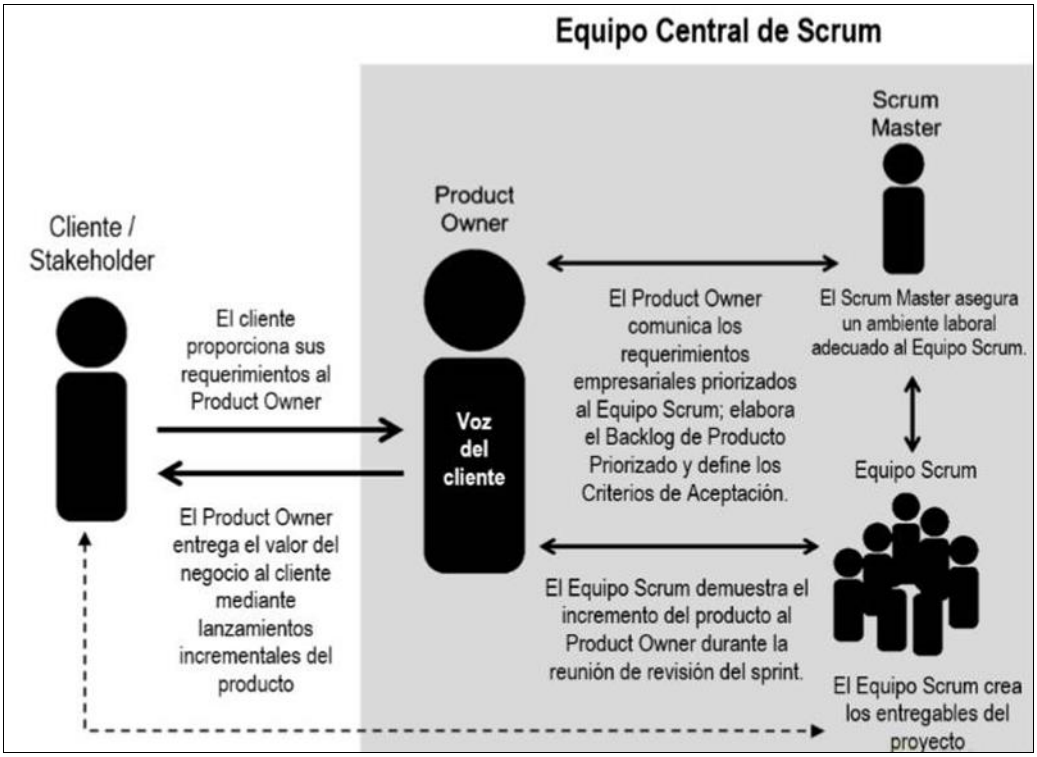
8.- Roles:

Cliente

En realidad, en *Scrum*, más que de cliente se debe hablar de los *Stakeholders*[29], es decir, de todas las personas y organizaciones que tienen algún interés en el trabajo.

([28]) PRIOLO, Sebastián, op. Cit. Pág. 157.

([29]) ÁLVAREZ GARCÍA, Alonso, DE LAS HERAS DEL DEDO, Rafael, y LASA GÓMEZ Carmen. Op. Cit. Pág. 63.



Aunque no parezca que se le dedica mucha atención en *Scrum*, en realidad juega uno de los papeles más importantes, al ser quien tiene una necesidad que plantear al equipo, y cuenta con los recursos (generalmente económicos) para construir la solución. Así que es dueño de los requisitos y de los recursos.

Los *Stakeholders* en su conjunto, no tienen por qué entrar a formar parte del proceso de *Scrum*, pero es conveniente que lo conozca y esté familiarizado con su terminología y forma de operar.

El papel de cliente en *Scrum* implica ante todo dos grandes tareas: proporcionar requisitos y validar resultados. Es decir, definir el producto que se quiere construir, y examinar cuidadosamente los resultados intermedios (y finales) que ofrezca el equipo para dar sus comentarios, correcciones y sugerencias, su *feedback*.

Product Owner

En los productos y proyectos en los que la metodología Agile no está presente, hay un cliente que tiene una necesidad y define unos requisitos que son

tomados por un equipo que quiere convertirlos en realidad, los interpreta en base a su conocimiento y crea lo que entiende que el cliente está esperando. El resultado suele distar de lo que el cliente había querido explicar inicialmente en sus requisitos.

En *Scrum*, para mejorar este proceso de entrega a un cliente y minimizar la diferencia con lo finalmente entregado al cliente, es la creación de un rol específico, que ayude a converger la Visión del cliente con la Visión del equipo de trabajo. Este es el rol del *PRODUCT OWNER*, PO O DUEÑO DEL PRODUCTO[30].

El PO es la voz del cliente, es el visionario que aúna las necesidades de todos los clientes, personas a las que les puede afectar o resultar relevante el producto o proyecto en desarrollo, conocidos colectivamente *STAKEHOLDERS*. Es el responsable de inspeccionar y adaptar estas necesidades, al esquema de trabajo definido en *Scrum*, esto quiere decir que constantemente estará creando nuevos requisitos.

Dichos requisitos se pueden crear continuamente porque el PO mantiene una Visión actualizada del producto o proyecto. La Visión, una vez convertida en el objetivo perseguido, representa todas las características que aportan valor al usuario o cliente final.

El PO es el estratega del producto, se encarga de definir una estrategia a largo y corto plazo para garantizar el éxito del producto, ya que es el responsable final del éxito del proyecto y de su ROI (Retorno de Inversión). Por esta razón el *Product Owner* debe tener a su alcance todos los medios y poder de decisión y asi materializar ese éxito.

Además, cuando se realiza un proyecto, conjuntamente con los intereses del cliente final, se generan muchas expectativas por parte de otras personas (gestores, proveedores, etc.) que deben ser gestionadas correctamente. Es responsabilidad del *Product Owner* representar al producto frente a todas las personas o entidades, incorporando toda la información derivada de ellos como requisitos al producto. Esta responsabilidad requiere unas capacidades especiales de comunicación y negociación para poder conseguir siempre lo mejor para el producto o proyecto en desarrollo.

([30]) Ibidem.

El *Product Owner* debe asumir tareas de gestión, las que implican definir y actualizar el plan de entregas del proyecto o controlar que el presupuesto para la ejecución del producto o proyecto sea el correcto.

El *Product Owner* es líder, pero siempre sin perder de vista que es también un miembro de un equipo que tiene un interés común. Lo ideal sería siempre que el cliente fuera el *Product Owner*. En muchas organizaciones no es posible que el cliente sea el propio PO, por lo que se crea la figura del *Proxy Poduct Owner*, como representante de los clientes aunando sus peticiones.

El *Product Owner* no guía el producto o al equipo diciendo como hacer las cosas, sino que específica qué se tiene que hacer y en qué orden para que el equipo se auto-gestione y encuentre la mejor manera de llevarlo a cabo. El *Product Owner* reta al equipo y este se compromete y responde al desafío.

Precisamente compromiso es una de las palabras claves en *Scrum* y lleva a que se diferencien los roles. El *Product Owner* es un papel comprometido al 100% con el equipo y con el producto. Cualquier otro implicado en el producto o *stakeholder* podrán aportar mucho al producto o proyecto pero no llegaran a estar comprometidos.

Todas estas responsabilidades que un PO debe asumir se traducen en las siguientes tareas operativas dentro de *Scrum*:

- Definición de la visión del producto
- Organización de talleres de obtención de requisitos
- Creación y mantenimiento de *Backlog*
- Resolución de dudas del equipo en cualquier momento
- Preparación de las reuniones de estimación y planificación
- Asistencia a las reuniones de *Scrum*
- Aceptación o rechazo del trabajo realizado durante un *Sprint* por parte del equipo.

Scrum Master

Se trata de un papel difícil, el cual reúne responsabilidades, no está dotado del "poder" para ordenar a otros miembros del equipo, si no que su capacidad de influir

viene dada más bien por el ejemplo o inspiración que pueda inducir. El SM ejerce un liderazgo moral, nunca jerárquico. Pero aunque no tenga la autoridad formal sobre el equipo, si tiene mucho que decir sobre el proceso.

El *Scrum Master*[31] es, por encima de todo, el responsable del proceso y de garantizar que se sigue *Scrum*. Se trata de un rol intermedio entre el *Product Owner*, responsable del éxito del producto, y del equipo que se responsabiliza del desarrollo exitoso del trabajo.

Si hay una palabra que define al *Scrum Master* es facilitador, armoniza el trabajo de los demás, dando protagonismo a unos cuando otros se muestran inseguros, y consiguiendo que todo el equipo realice un trabajo conjunto sin estridencias.

Es cierto que suele convocar reuniones y hacer todo lo necesario para que asistan las personas implicadas, pero no está a las órdenes del PO ni debe ser su subordinado para tareas administrativas.

El cometido principal del *Scrum Master* es encargarse del seguimiento correcto de los principios de *Scrum*, no es una tarea abstracta, se manifiesta en aspectos muy concretos, por ejemplo:

- Velar por la productividad del equipo, lo que se traduce de manera práctica en aislar al equipo en la medida de lo posible de interferencias externas que puedan distraerle, y en resolver los impedimentos que puedan aparecer en el trabajo.
- Debe procurar que fluya la comunicación y la colaboración. Por eso asiste a todas las reuniones, y procura garantizar la asistencia de las personas necesarias para su éxito. También busca este objetivo fuera de las reuniones, en el trabajo día a día.
- Además es responsable de introducir y fomentar las prácticas Agile. Por ello debe conocer profundamente la metodología y sus variantes y hacer un esfuerzo por difundir este conocimiento entre el equipo y el PO.

([31]) Ibidem.

- Supervisa el *Backlog*, asegurándose que todas las historias estén correctamente descritas, priorizadas y estimadas.
- Es también el intermediario entre el mundo exterior y el equipo de trabajo. Esto forma parte de su misión de fomentar la productividad protegiendo al equipo de interferencias externas.
- Tiene un papel destacado en la formación del equipo, el PO e incluso los clientes para que adopten las mejores prácticas de Scrum en su trabajo. El *Scrum Master* es el primer *coach* del equipo.

Lista de atributos y características principales del Scrum Master que selecciona Mike Cohn[32] :

- **Responsable**: el papel del *Scrum Master* no debe restarle protagonismo a los miembros del equipo. No debe destacar ni distinguirse. Convence por medio del ejemplo y la inspiración.
- **Humilde**: el *Scrum Master* se basa en la colaboración y debe ser el abanderado de este principio fomentando la colaboración y buscando la forma de frenar las actitudes contrarias y egoístas. Por ello, ayuda a crear una atmósfera positiva de colaboración, haciendo que los debates sean constructivos y no deriven en disputas con vencedores y vencidos.
- **Comprometido**: formalmente pueda parecer que es tanto el *Product Owner* o el equipo quienes tienen los compromisos más fuertes con el éxito del trabajo. Sin embargo es imposible que el proyecto llegue a buen puerto si el propio *Scrum Master* no adopta una actitud comprometida con el proyecto, sus fines y la forma de llevarlo a cabo. Eso se manifiesta en procurar resolver con rapidez los impedimentos que surjan, y ayudar a mantener permanentemente actualizada la información del trabajo. Como muestra de su compromiso, el *Scrum Master* debería mantenerse ligado al proyecto hasta su conclusión.

([32]) "Leader of the Band" se puede encontrar en: www.scrumalliance.org/articles/36-leader-of-the-band.

- **Influyente**: al no contar con autoridad formal, el *Scrum Master* no tiene más remedio que convencer por medio del ejemplo y de su capacidad para persuadir a los otros. Esto obliga a que un buen *Scrum Master* deba dotarse de unas armas más políticas que metodológicas, técnicas o científicas.
- **Entendido**: Precisamente una forma de influir en otros es desplegando un conocimiento erudito, tanto de *Scrum* y aspectos metodológicos como del campo de aplicación el en que se esté desarrollando el trabajo. El objetivo es entender la naturaleza de los problemas que pueda tener el equipo en su trabajo.

Equipo de trabajo

Ya se tiene claro un primer esbozo de **qué** se tiene que hacer, así que es el momento de organizar el **cómo** se va hacer. Estas cosas que hay que organizar son: el equipo y la logística, el *Product Backlog*, las reglas del juego y la gestión de la incertidumbre.

Los equipos de *Scrum* están formados por el *Scrum Master* y un equipo multidisciplinar con ello se hace referencia a que es un equipo en el que se aglutinan todos los perfiles necesarios para la realización de la visión. En *Scrum* no se cuenta con varios equipos especializados que van realizando sus tareas de forma secuencial. La idea es que todas las operaciones se hagan por un mismo equipo.

Cuando se está creando un equipo en una empresa con cierta cultura agile, lo ideal es crear el equipo y que sea este quien elija a su *Scrum* Master. El *Scrum Master* es el representante del equipo y está para servirlo. El liderazgo que sustenta es basado en el ejemplo y en la confianza que el equipo mantiene en él. Elegirlo antes de la creación del equipo y que este participe en la formación del mismo, crea un sentimiento de jerarquía contrario a las bases de autogestión y de “otorgamiento de poderes” o empowerment al equipo. Además del conocimiento específico necesario para formar

un buen equipo agile, sus miembros deben tener una serie de cualidades[33], que harán que la capacidad conjunta del equipo se maximice:

A) **Trabajo en equipo**: *Scrum* se trata de trabajo en equipo. Establece que el trabajo de dos personas en equipo es más que la suma de sus dos trabajos por separado. Miembros que no sepan trabajar en equipo quedaran aislados rápidamente.

B) **Generosidad**: no se trabaja por objetivos individuales. Los objetivos son los del equipo. Por esta razón es casi más importante ayudar a un compañero, si tiene problemas, antes que terminar una propia tarea. No falla un miembro del equipo, falla el equipo.

C) **Comunicación**: un equipo funcionando es un equipo sincronizado. La base de la sincronización es la comunicación. Miembros aislados y trabajando por su cuenta no aportaran valor al equipo.

D) **Capacidad de aprendizaje**: *Scrum* se basa en el principio de revisar y adaptar. Para hacer esto, es necesario un aprendizaje constante y una adaptabilidad.

Una vez que se forma el equipo, es necesario encontrar un sitio para que trabaje. Lo ideal sería que se compartiera un mismo espacio físico y que estén colocados cerca, ya que es la mejor manera de que se fomente la comunicación que se está buscando para sincronizar el equipo.

Finalmente, es necesario dotar al equipo con todas las herramientas y materiales que vayan a necesitar en la ejecución del proyecto o creación del producto.

La forma tradicional de desarrollar el trabajo se basa en la jerarquía, la autoridad formal y la estructuración. Frente a ella, el equipo en *Scrum* se auto-organiza, tiene la responsabilidad final por el éxito del trabajo y es capaz de asumir cualquier actividad dentro de las necesarias para desarrollar el proyecto.

El equipo debe tener un elevado grado de compromiso. Debe ser capaz de auto-organizarse, frene a un equipo tradicional donde un jefe de proyecto asignaba a

---

([33]) ÁLVAREZ GARCÍA, Alonso, DE LAS HERAS DEL DEDO, Rafael, y LASA GÓMEZ Carmen. Op. Cit. Pág. 106.

cada persona tareas concretas y fijas. El equipo también deber ser capaz de realizar todas las actividades requeridas en el trabajo.

Con la ayuda del *Scrum Master*, el equipo deberá ser capaz de seguir la evolución de su productividad y hacer un proceso continuado de mejora, lo que incluye también sus conocimientos y habilidades, y la aplicación de las mejores prácticas en su trabajo.

CAPITULO VI

PROCESO DE SCRUM

Sumario: 1.- Visión esquemática del ciclo de Scrum. 2.- Sprint 0. 3.- Product Backlog. 4.- Sprint Planning. 5.- Sprint Backlog. 6.- Tablero de tareas. 7.- Planificación detallada. 8.- Desarrollo de Sprint. 9.- Block de impedimentos. 10.- Sprint review. 11.- Retrospectiva.-

1.- Visión esquemática del ciclo de Scrum:

Sprint 0.Todo empieza aquí[34]. Es una etapa previa muy importante para el desarrollo del resto del trabajo en la que tiene un papel relevante el *Product Owner* o PO que es parte del cliente (salvo excepciones) y la persona que actúa como punto de contacto y representante para el equipo. Partiendo de las necesidades del proyecto, el PO se encarga de armar las bases para el trabajo posterior: crear el equipo, contar con los recursos, fijar requisitos y plazos, traducir las necesidades de cliente en unos requisitos y elaborar el diseño preliminar formal. El resultado es un *Backlog* o

([34]) Ibídem. Pág 59.

repositorio del proyecto en el que se han introducido todos los requisitos expresados en forma de menor a mayor detalle.

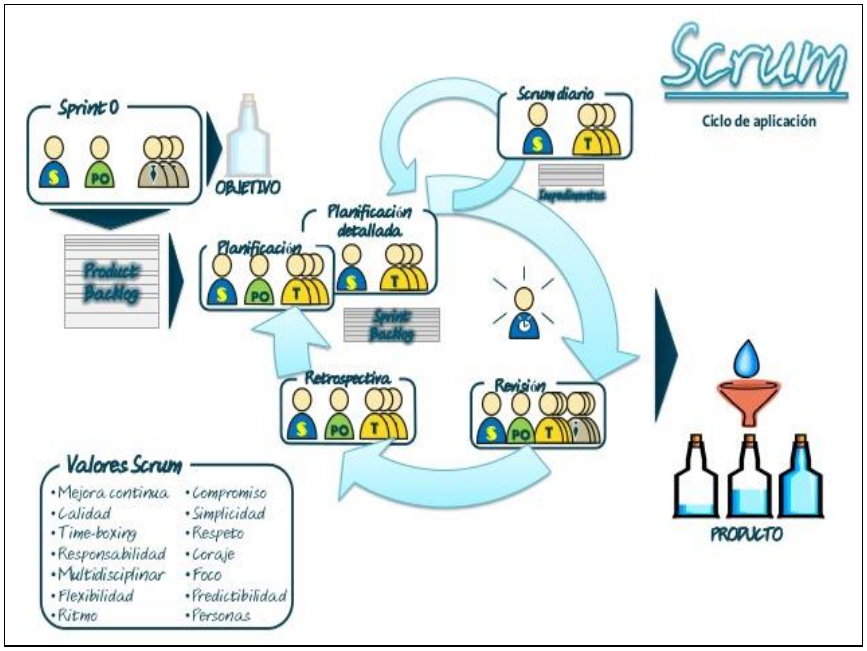
El trabajo posterior se divide en *sprints* o iteraciones que se agrupan en una o varias *Releases* o entregas (al menos deberá haber una al final) de acuerdo con la longitud del proyecto, las necesidades del cliente y la naturaleza del trabajo.

Cada *Sprint* arranca con el *Planning* donde el PO podría contar con la ayuda del *Scrum Master* o SM (facilitador del trabajo, intermediario entre PO y equipo, y vigilante del cumplimiento de los principios de *Scrum*) prioriza todas las historias de usuario (temas y épicas) y añade criterios de aceptación para determinar cuando se han cumplido. El equipo de trabajo valora el esfuerzo necesario para realizarlas y selecciona, siguiendo la prioridad, las que podrán realizarse en el transcurso del *Sprint* de acuerdo con la capacidad de trabajo del equipo. En una segunda etapa de la planificación, el equipo traduce las historias al lenguaje del proyecto, y las subdivide en unidades menores o tareas. Con todo ello se construye el *Sprint Backlog* o repositorio de los trabajos que se realizarán durante la iteración.

El trabajo se va realizando a lo largo del *Sprint* y el equipo va exponiendo sus avances en la *Daily meeting, Scrum* diario o reunión diaria. Se trata normalmente de un foro interno (compuesto por el equipo y SM) donde cada participante comenta los avances realizados, el trabajo futuro y los impedimentos para la actividad. Estos impedimentos se guardan en un *Impidement Backlog* y el SM vigila que se resuelvan.

El final del Sprint lo señala el *Review* meeting o reunión de revisión de resultados, donde el equipo con el SM expone los trabajos realizados y los resultados alcanzados al PO y al resto de las personas que pueden estar interesadas, para que los acepten o no y recoger comentarios y sugerencias que se puedan aplicar en la próxima iteración. Finalmente, en la reunión de Retrospectiva, el equipo y el *Scrum Master* revisan el proceso e identifican por un lado aspectos positivos que merece la pena cuidar y mantener y por otro, puntos de mejora que hay que resolver dentro del proceso de mejora continua que supone *Scrum*.

Todo el proceso queda documentado y se hace un seguimiento de la capacidad del equipo o velocidad como medida útil para el incremento de la productividad.



2.- Sprint 0:

Cuando es el momento de crear algo nuevo, en cualquier ámbito, existen una serie de requisitos y una preparación mínima, que hay cubrir para poder empezar. *Scrum* no es una excepción y necesita una gestación que facilite el resto de las actividades que se realizarán posteriormente.

A este proceso de gestación o preparación inicial, que recoge todas las actividades necesarias para iniciar las iteraciones de trabajo, se le llama *SPRINT 0, SPRINT ZERO O INCEPTION SPRINT*.

Tiene que existir una fase inicial, en la que se prepare toda la logística, mecánica y metodología a seguir durante todo el desarrollo del proceso de creación de producto o proyecto. El encargado de esta gestión es el *PRODUCT OWNER*.

Las reglas del juego

Para poder tener un equipo sincronizado es importante acordar cual será la forma de trabajar, viene definida por dos dimensiones[35], por un lado se debe identificar como trabajará el equipo a nivel procedimental, y por otro lado se debe acordar como trabajará el equipo a nivel ejecutivo.

A *nivel procedimental*, se pueden definir la mecánica de trabajo que se va a seguir y las herramientas que se van a usar. En este periodo, se suelen decidir acciones como la duración de los *Sprints*, qué herramientas de comunicación se van a utilizar, cuál va a ser el procedimiento para realizar las entregas o cualquier proceso que se tenga que definir.

A *nivel ejecutivo*, hablando de la propia ejecución de las tareas para llevar a cabo el proyecto o producto, el equipo tiene que definir o coordinar como va a trabajar para maximizar su sincronización. A este nivel se fijan los estándares, principios, reglas o criterios que el equipo va a compartir. El fijar estas reglas o principios tiene como objetivo minimizar las discusiones internas, o el tiempo perdido en dudas relacionadas con elementos, que no sean de la propia ejecución de las tareas del proyecto o producto.

Definiendo el Plan Maestro. Creando el *Release Plan.*

Es muy importante definir qué y cómo se va hacer un producto o proyecto, pero no hay que olvidar el tener claro **cuándo** se van a hacer las cosas, existe una planificación detallada que se denomina **plan de entrega o *reléase plan*[36].**

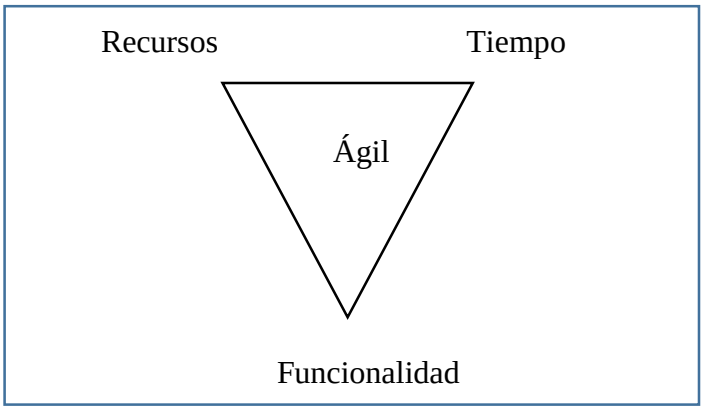
El plan de entregas es algo que se empezara a gestar en el Sprint 0, pero tendrá una gran adaptabilidad durante todo el ciclo de vida del producto o del proyecto. Por norma general se debe revisar el plan después de cada *Sprint.*

Para empezar a hacer un plan de entregas es necesaria una información de base para trabajar con ella.

([35]) Ibídem. Pág. 107.
([36]) Ibídem. Pág. 109.

Lo primero que es necesario, es tener una base de requisitos en el *Product Backlog*, aunque sea a nivel de temas o *themes*. Los temas, son categorías de alto nivel que engloban colecciones de requisitos. Lo segundo que es necesario es precisamente una necesidad de crear una entrega. El tener una necesidad o razón para hacer una entrega es importante para fomentar el compromiso del equipo.

Una vez que se tiene estas dos premisas es el momento de planear una entrega. Para poder hacerlo, se tendrá que jugar con tres variables principales en la gestión de proyectos o procesos: Tiempo, Recursos y funcionalidades-características.



Las tres variables de gestión de proyectos no son independientes, si se modifica una de ellas afectara a las otras dos indirectamente. Si se parte de la base de que se van a tener unos recursos conocidos y constantes para la realización del proyecto, las dos variables con las que se podrá jugar son el tiempo y la funcionalidad.

El plan de entregas es dinámico, porque depende de la velocidad, estimaciones y prioridades del *Backlog* que van cambiando *Sprint* a *Sprint*.

En resumen, el plan de entregas se construye en base a la experiencia del equipo, por medio de sus estimaciones, ya que son quienes realmente conocen cuanto se tarda en hacer las cosas. Además no será un plan fijo, se adaptara como todo elemento en *Scrum* después de cada iteración.

3.- Product Backlog:

Necesitamos identificar las grandes divisiones del trabajo que vamos a realizar, ordenarlas, priorizarlas y documentarlas.

Vamos a aprovechar la potencia de *Scrum* para construir nuestra lista flexible, adaptable y dinámica de requisitos: nuestro *Product Backlog* o Pila de Producto.

La visión del producto o proyecto, es el punto de partida para empezar a concebir la base en la que se va a trabajar, pero es insuficiente para empezar a trabajar de manera práctica.

Para iniciar el proceso de desarrollo del trabajo es necesario definir un nuevo artefacto que permita alimentar al equipo que va a convertir la idea en una realidad. Este artefacto debe contar con definiciones concretas de lo que se quiere crear.

Durante años se han utilizado las definiciones de requisitos para especificar los productos o proyectos que se quieren llevar a cabo. Han funcionado de manera correcta pero han presentado siempre una serie de problemas que deterioran la calidad de los resultados obtenidos. Los problemas más importantes identificados se enumeran en la siguiente lista[37]:

- Desconexión entre las personas que definen los requisitos y las personas que los llevan a cabo.
- Interpretación ambigua de los requisitos.
- Cambio de los requisitos desde que se definen hasta que se implementan por variaciones en las condiciones internas o externas.
- Necesidad de incorporar nuevos requisitos durante el ciclo de vida del desarrollo de un producto.

Con el objetivo de res.olver estos problemas en las especificaciones tradicionales de requisitos surge en *Scrum* el concepto de *Product Backlog*, PB o Pila de Producto. Este artefacto intenta definir un nexo de comunicación entre los clientes y el equipo Scrum. Este artefacto sirve de punto de encuentro para discutir, conversar,

([37]) Ibídem. Pág 116.

definir y aclarar las características que deben cumplir el producto o proyecto que se está llevando a cabo de una forma dinámica y cambiante durante la duración de este.

Los requisitos se descubren y emergen constantemente así que el Product Backlog de ayer posiblemente no sea el de hoy, éstos surgen para aportar valor de negocio a los que se está desarrollando.

Los requisitos pueden tener tipos, esto suele ser a gusto del equipo. Lo mínimo que debe tenerse es una división de requisitos funcionales y no funciones (u operacionales). Los funcionales se conocen como historias de usuario que identifican una situación funcional del producto desde el punto de vista de un usuario que desempeña un papel determinado. Los no funcionales están relacionados con cualidades que son necesarias para el producto y no se pueden definir mediante historias de usuario.

Cada equipo puede tener su propia organización pero en la literatura se suele hablar de una estructuración en Temas (*Themes*), Épicas (*Epics*) e Historias del Usuario (*User Stories*).

La pila de producto debe convertirse en algo real sobre lo que se pueda trabajar y discutir. El cómo se implementa depende del equipo. Hay equipos que utilizan un tablón con los ítems en *post-it*, otros utilizan las hojas de cálculos o pueden utilizarse herramientas informáticas desarrolladas para este propósito. Independientemente del formato del *Product Backlog* que se utilice, lo importante es que sea algo que esté al alcance del equipo en todo momento y con el que todos los miembros se sientan cómodos. Si no ocurre esto, el *Product Backlog* se dejara de usar como base del trabajo.

Un *Backlog* priorizado sirve para organizar el plan del equipo y conocer cuál va a ser la ruta de trabajo a corto plazo. La primera pregunta importante es saber que hay que priorizar o a que nivel hay que hacerlo. Lo que se tiene que priorizar son los elementos que se tienen en el *Product Backlog*. Estos elementos son las historias del usuario y los requisitos no funcionales.

El responsable de priorizar es el *Product Owner* pero no tiene por qué estar solo en esta tarea. El *Product Owner* se puede valer de su equipo de producto,

stakeholders o del propio equipo de *Scrum* para crear una prioridad que refleje una visión más universal del producto.

Para realizar una buena priorización es importante definir un criterio sobre el que priorizar. Si no se define, no se podrá alinear la opinión de todas las personas que están participando en la priorización.

4.- Sprint Planning:

Scrum y las metodologías ágiles, aunque eviten formalismos y burocracia, no fomentan una forma de trabajo sin control. Por eso se organiza la actividad del proyecto en una serie de **iteraciones o *Sprints*** cuya duración se fija inicialmente en el *Sprint* 0. Al principio de cada iteración hay que discutir su contenido, su alcance y como verificar que se han llegado a los objetivos planteados.

A continuación, en una reunión llamada ***Sprint Planning*** en la que el equipo revisa, junto con PO, las tareas de *Product Backlog* valorando su complejidad y selecciona, siguiendo la prioridad, las que podrán realizarse en el transcurso del *Sprint*. En una segundo etapa de la planificación, el equipo traduce las historias a lenguaje de proyecto, y las subdivide en unidades menores o tareas. Con todo ello se construye el *Sprint Backlog* o repositorio de los trabajos que se realizaran durante la iteración.

El trabajo inicialmente definido en el *Sprint* 0 del proyecto se divide en *Sprint* o iteraciones, que a su vez se pueden agrupar en una o varias *Release* o entregas (al menos habrá una al final) de acuerdo con la longitud del proyecto, las necesidades del cliente y la naturaleza del trabajo.

Esta división del trabajo es muy útil por varias razones: permite abordar tareas más reducidas y controlables, reduciendo la incertidumbre; facilita la obtención de resultados intermedios, lo que va a servir para detectar desviaciones o malas interpretaciones de los requisitos mucho antes; y permite definir una velocidad constante de avance del proyecto, siendo predecible.

Cada una de las iteraciones o *Sprints* se dividen fundamentalmente en tres etapas: planificación del contenido, desarrollo del trabajo y revisión del resultado.

Todo *Sprint* se comporta como un proyecto pequeño en sí mismo y por ello debe tener un alcance y objetivos claros más allá de cumplir unas determinadas horas de trabajo. Una de las primeras tareas de la planificación es precisamente identificar ese objetivo, algo que deberá hacer el *Product Owner*, para lo que puede contar con la ayuda del *Scrum Master*.

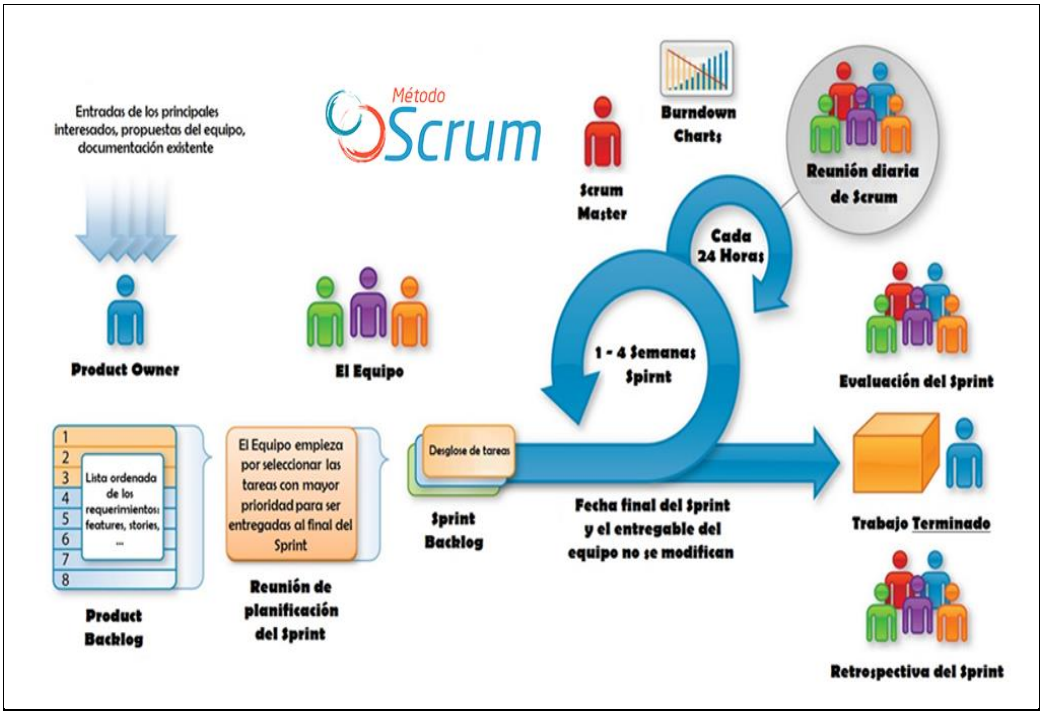
Otra tarea importante previa al proceso de *Planning* es la priorización de las tareas del *Backlog*. El repositorio o *Backlog* del proyecto contiene la lista de trabajos definidos usando el lenguaje de negocio, por lo que reciben el nombre de Historias de Usuario (*User Stories*).

Aunque todas las historias de usuario son importantes para alcanzar el objetivo último de trabajo, se deben ordenar de más a menos prioritarias, ya que no es posible abordar la resolución de todas simultáneamente. Esta priorización define el orden en el que se hará el trabajo en el proyecto, y con él, el contenido aproximado de cada *Sprint*.

Con el *Product Backlog* priorizado podemos continuar adelante. El punto de partida para iniciar un *Sprint* es la planificación. La misma consta de de dos etapas: una de selección de historias, y otra subdivisión en unidades más pequeñas o tareas. En la primera fase se realiza la selección de historias para poblar el *Sprint Backlog*.

5.- Sprint Backlog:

El objetivo del proceso de *Sprint Planning* es contar con *Sprint Backlog* o pila de *Sprint*. Se trata de un repositorio que recoge los trabajos que van a realizarse en una iteración o Sprint determinado. Es decir, que cada *Sprint* tiene un *Sprint Backlog* distinto. Este repositorio contiene las historias de usuario y, sobre todo, las tareas que el equipo, que es quien gestiona este *Backlog*, ha identificado en el momento de la planificación de detalle. El *Sprint Backlog* es propiedad del equipo, que es quien lo gestiona y actualiza. El Sprint *Backlog* se puebla a partir del *Product Backlog*, seleccionando historias en función de la priorización hecha por el *Product Owner*.



Cada historia seleccionada se divide a su vez en tareas, descriptas en el lenguaje del dominio técnico del trabajo, pequeñas, detalladas y que pueden contar con una estimación de tiempo para su realización.

Aunque hay equipos que prefieren usar medidas relativas para comparar las tareas, es bastante habitual que se tome como unidad de referencia el tiempo, ya que las tareas concretas se pueden estimar con más exactitud.

Si lo comparamos con el *Product Backlog*, el contenido del *Sprint Backlog* es mucho más rico y detallado. Para empezar, cada historia seleccionada para el Sprint debe contener un claro y detallado criterio de aceptación del resultado que se espera. Este criterio de aceptación es uno de los elementos fundamentales de la operativa del *Sprint*. Sirve de guía a los miembros del equipo a la hora de desarrollar el trabajo y alcanzar los resultados y permite validar si se han alcanzado o no los objetivos del trabajo.

En paralelo, cada tarea va a tener una “definición de hecho” o *Definition of done* que introduce el equipo para saber cuándo ha alcanzado el resultado esperado para esa tarea concreta.

Cada miembro que trabaje en una historia puede ir añadiendo información relativa a su desarrollo y, si la herramienta usada lo permite, incluir todo tipo de documentos, diagramas, planos, código o fotografía. Toda esa información permite seguir el trabajo mientras se desarrolla, y documentarlo a su finalización, de forma que sea fácil entender por otras personas que se hizo y cómo. El *Sprint Backlog* tiene tres dimensiones principales[38]:

En primer lugar está la **prioridad,** que refleja la del *Product Backlog*. Las historias seleccionadas para el Sprint durante el proceso de planificación se extraen del *Product Backlog* de acuerdo con el orden de prioridad fijado por el PO.

Otra dimensión es la del **detalle**, de forma que dentro de cada historia de usuario tendremos el desglose de las tareas concretas.

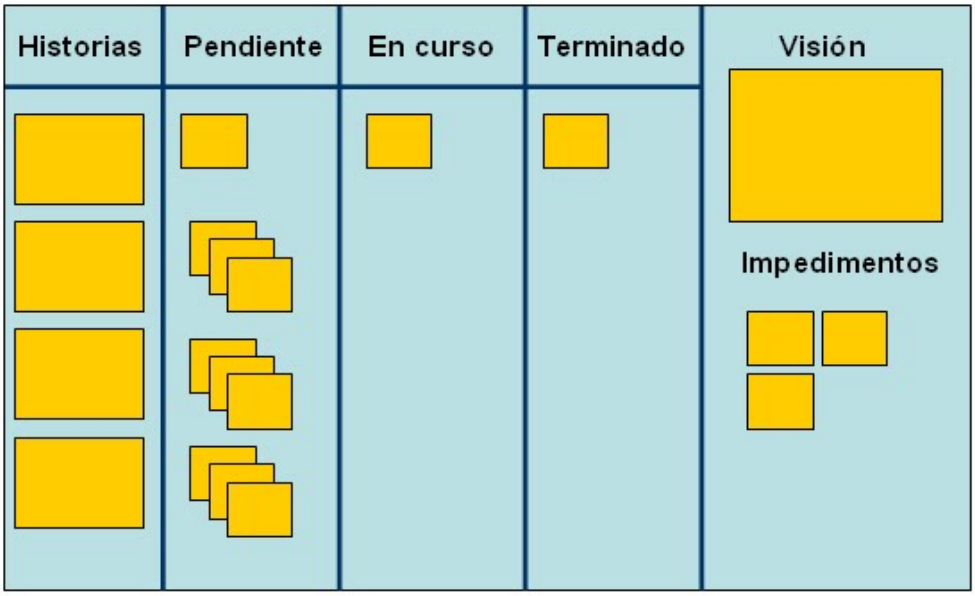
La última es muy necesaria y se refiere al **estado**. Una historia (y sus tareas) pueden estar: pendientes de iniciar, en curso, terminadas e impedidas.

- Por definición, antes de empezar cualquier historia o tarea, estará en estado **pendiente**, es decir, que nadie la ha seleccionado aun del *Sprint Backlog* para resolverla. Las historias pendientes se ordenan por prioridad.
- Cuando un miembro del equipo de trabajo ha seleccionado una tarea para desarrollarla y completarla, la tarea pasa a estar en curso.
- Si la tarea se considera realizada de acuerdo con la definición de hecho (*definition of done*), pasa al estado de terminada. Una historia no puede considerarse terminada si no lo están todas las tareas que la componen.
- Una tarea puede estar impedida, que significa que hay algún elemento que no permite desarrollar el trabajo sobre ella. Es importante documentar el impedimento, identificar la manera de resolverlo y asignar un responsable para resolverlo y continuar.

([38]) Ibídem. Pág. 154.

6.- Tablero de tareas:

Aunque hay muchas aplicaciones y medios informáticos para gestionar el proceso de *Scrum*, una de las más simples es también una de las más útiles. Se trata del **panel, tablero de tareas o *tast board***, un sistema extremadamente sencillo de mostrar toda la información de un vistazo.



Puestos en un lugar visible para todo el equipo, permiten tener en todo momento un estado actualizado de la evolución del *Sprint*. Se compone de etiquetas autoadhesivas, carteles, textos escritos, etc. que contienen historias de usuario y, si el formato lo permite, las tareas en que se dividen. Estos elementos se disponen en columnas que representan cada uno de los estados (pendientes, en curso, etc.).

7.- Planificación detallada:

Por lo general, la planificación del *Sprint* se divide en dos reuniones diferenciadas. La primera, el *Sprint Planning*, sirve para poblar el *Sprint Backlog* con una selección de historias del *Product Backlog*.

La segunda reunión, la **planificación detallada o de táctica** para el *Sprint*, busca dividir las historias de usuario seleccionadas en el *Sprint Backlog* en partes más

manejables llamadas tareas. El objetivo es obtener tareas, entidades pequeñas, descritas en el lenguaje del dominio del trabajo, y no en el lenguaje de negocio.

Una tarea describe un trabajo concreto que debe desempeñar el equipo y sus resultados no tienen por qué ser revisado por el PO. El conjunto de las tareas de una historia sí que da lugar a resultados de producto y si tienen que ser validadas por el *Producto Owner*.

8.- Desarrollo de Sprint:

Los proyectos que apuestan por *Scrum* se dividen en una serie de iteraciones o *Sprints*. En un proyecto convencional, tras dedicar mucho tiempo y esfuerzo a diseñar el trabajo, cada vez con más detalles, se realiza todo el desarrollo del proyecto de manera continuada para entregar al final del proceso un producto acabado.

La realidad es que en ese tipo de proyectos rara vez se cumplen plazos y costes, o lo hacen sacrificando la calidad. Pero como, por encima de todo, los requisitos pocas veces permanecen estables, esto se traduce en cambios y correcciones sobre un producto terminado que no está preparado para ello.

Scrum aporta una aproximación **incremental:** el producto se construye poco a poco en ciclos limitados, al final de los cuales, hay una versión parcial pero funcional del producto.

La duración del *Sprint* es una decisión importante que tiene influencia en el desarrollo del trabajo. Se fija en el momento de iniciar la primera iteración, incluso antes (en el *Sprint* 0) pero esa decisión no es inamovible, puede ajustarse en función de las necesidades del proyecto y del equipo.

Los *Sprints* cortos permiten detectar lo antes posible problemas en el curso del desarrollo. Son más indicados en las etapas iniciales del proyecto, en actividades de innovación o poco definidas, cuando los requisitos pueden ser más inestables y sujetos a variación. También es conveniente cuando el equipo es inmaduro, se desconocen las técnicas y herramientas usadas, o sus miembros tienen poca costumbre de trabajar juntos. Al sucederse con mayor rapidez los momentos de planificación y

revisión de resultados, pueden detectarse antes elementos o condicionantes negativos, y actuar a tiempo para corregirlos.

Cuando hay una relativa estabilidad de requisitos, el entorno es conocido, el equipo maduro y sus miembros acostumbrados a trabajar juntos, puede optarse por *Sprints* más largos. Esto supone depositar una confianza mayor en el equipo ya que la revisión de resultados se demora más, y con ella la posible corrección que haya que aplicar.

Decidir la duración del *Sprint* es una tarea compartida por todo el equipo *Scrum* (PO, SM y equipo de trabajo), aunque sea el *Scrum Master*, por su conocimiento de las técnicas de *Scrum* y el equipo, por su mayor dominio en el área de conocimiento del trabajo, quienes tienen más que decir a la hora de alargar o reducir cada iteración.

Sea corto o largo el *Sprint* hay que fijarse en la consecución de sus objetivos. Esto supone cuidar determinados aspectos[39]:

- Para empezar la **estabilidad:** no se puede cambiar su contenido o variar su objetivo, salvo que este muy justificado y tenga un impacto beneficioso real sobre el trabajo en curso.
- Obtención de los resultados: el objetivo de cada *Sprint* es obtener un producto parcialmente desarrollado, que crezca incrementalmente. Si no hubo cambio al final de la iteración habrá que preguntarse seriamente si se ha planteado correctamente el trabajo.
- Mejora continua: porque el tiempo dedicado al *Sprint* es tiempo para identificar y aplicar mejoras en la forma de trabajar, en la productividad y en la calidad.
- Anticipación: un *Sprint* es el paso previo al siguiente. Hay que aprovechar el tiempo de la iteración para anticipar los cambios y retos de la siguiente.

([39]) Ibídem. Pág 182.

Daily Meeting o Scrum Diario

Para facilitar la sincronización entre todos los miembros del equipo, mantener el ritmo y para fomentar la comunicación interna, *Scrum* propone la ***Daily Meeting*, reunión diaria o *Scrum* Diario o *Daily Scrum***. Es una reunión que se realiza diariamente, en la que participan todos los miembros del equipo y *Scrum Master*, a la que puede asistir el PO.

En ella, cada miembro del equipo, de manera voluntaria pasará a comentar: qué actividades ha realizado, qué actividades va a realizar y qué impedimentos ha encontrado para continuar con su trabajo. Estos puntos son los que se deban tratar y ninguno más. No hay que entrar en el detalle de lo que se ha hecho.

Hay que centrarse en el objetivo de la reunión y hacerla breve y productiva.

En la reunión diaria el SM participa como facilitador y debe tomar nota de los impedimentos que haya para seguir su solución.

La *Daily Meeting* permite detectar en momentos muy tempranos problemas que, de otro modo, podrían crecer y convertirse en obstáculos serios. Con ellas se garantiza un conocimiento actualizado del estado de los trabajos por parte de todos los miembros del equipo, lo que es una forma de incrementar su grado de compromiso, la sincronización y la auto-organización.

La información recibida debe actualizarse, preferiblemente en un panel o herramienta que sea fácilmente accesible para todos.

9.- Block de impedimentos:

El *Impediments Backlog*, pila de impedimentos o *backlog* de impedimentos, es un repositorio que recoge todo aquello que impide alcanzar los objetivos del proyecto. Un impedimento puede ser la carencia de una determinada información o herramienta, la imposibilidad de reunirse con una persona clave, etc. Pero también un impedimento es todo aquello que amenace con degradar la calidad de un producto final.

Los impedimentos son identificados por cualquier persona del equipo de *Scrum*, aunque lo habitual es que sean los miembros del equipo de trabajo quienes los

encuentren y los destaquen, ya que es ahí donde con mayor facilidad pueden surgir los obstáculos.

Un impedimento deber ser descrito, si es posible identificar su solución y asignar un responsable, aunque por término general es el *Scrum* Master quien se encarga de seguir su resolución. Para que quede constancia de su existencia y estado, los impedimentos se guardan en una pila, repositorio o *backlog* similar al del producto o al del *Sprint*.

Este repositorio es gestionado por el Scrum Master, y se mantiene actualizado a lo largo del Sprint con todos los impedimentos que se detecten y que se manifiesten en las reuniones diarias.

10.- Sprint review:

Una vez que se ha cumplido con el tiempo para la iteración o Sprint llega el momento de verificar si el trabajo realizado se ajusta a los planes y expectativas iniciales.

En el momento de la revisión, se evaluara hasta qué punto se cumplió con el compromiso y estimaciones.

Se propone la realización de una reunión denomina *Sprint Review* como resultado natural de cada iteración.

La *Sprint Review* es por tanto, el punto de comunión entre los responsables de un producto o proyecto y el equipo. Por medio de esfuerzos negativos o positivos, los responsables del producto asimilan el cómo se está desarrollando su producto y el equipo que lo lleva a acabo interioriza porque y para que se está desarrollando. De esta manera se puede acomodar este aprendizaje en las siguientes iteraciones de desarrollo o *Sprints* adaptándose el proceso a las necesidades del medio.

La *Sprint Review* también denominada *Customer Sprint Review*, es una reunión definida en la metodología *Scrum* que tiene lugar el último día del Sprint. El objetivo principal de esta reunión es la recoleccion de información o *feedback* sobre el estado del proyecto o producto en desarrollo.

Esta revisión sirve para repasar y discutir sobre los puntos clave de la novedad generada durante la última iteración. En otras palabras, analizar en el valor añadido al producto o proyecto.

La finalidad fundamental de la *Review* se puede resumir en tres puntos[40]:

- Recoger comentarios en referencia al objetivo del *Sprint*.
- Recoger la aceptación de las tareas que se seleccionaron al inicio del *Sprint* como trabajo operativo.
- Analizar si se necesitan mejoras en el trabajo realizado hasta la fecha o nuevos ítems en el *Product Backlog* para cubrir necesidades emergentes en el producto o proyecto.

El resultado final de esta reunión siempre debe ser el mismo, tiene que ser un *Product Backlog* adaptado a las nuevas necesidades del producto que se hayan detectado en la reunión de *Review*. También se tendrá un plan de entrega actualizado con la nueva velocidad del equipo y el cambio de prioridades.

11.- Retrospectiva:

Durante una Retrospectiva, todo el equipo se reúne para revisar el pasado, reciente o no tan reciente, y para poder organizar la forma de trabajar más efectiva en el futuro. Con la *Review* podemos mejorar lo que estamos construyendo y con la Retrospectiva nos ayuda a mejorar la forma en que trabajamos. Es decir, debemos buscar la mejora constante tanto en lo que hacemos como en cómo lo estamos haciendo.

Al terminar cada iteración y antes de comenzar con la planificación de la siguiente, es un gran momento para que el equipo analice en común cómo está trabajando. El objetivo de una Retrospectiva es aprender de la experiencia para mejorar constantemente la forma en que se está construyendo el producto y así trabajar *Sprint* tras *Sprint* de manera más efectiva y agradable. Durante las Retrospectivas se detecta todo aquello que no es útil al proyecto para eliminarlo o modificarlo así como para potenciar y maximizar aquello que si lo es. Este mecanismo de búsqueda de la mejora

([40]) Ibídem. Pág. 206.

constante, aumenta la calidad de lo que se construye. Por otro lado la Retrospectiva es una oportunidad extraordinaria para revisar los posibles riesgos del proyecto y mejorar la comunicación y la relación entre las personas del equipo.

En la Retrospectiva, el *Scrum Master* anima al equipo a revisar como fue el último *Sprint* teniendo en cuenta todos los aspectos: cada persona individualmente, las relaciones entre ellas, los procesos seguidos y las herramientas utilizadas.

Una Retrospectiva eficaz debe concluir en acciones concretas de mejora que deberán implementarse, en general, en un corto plazo de tiempo.

En una Retrospectiva deben tratarse con el mismo interés tanto los aspectos operativos como los más personales ya que ambos afectan directamente la productividad del equipo.

Es importante tener en cuenta que si un equipo no se reúne periódicamente para abordar estos temas y revisar cómo está trabajando, los problemas se repetirán, iteración tras iteración. Las Retrospectivas son la clave para mejorar el trabajo en equipo y el cómo se están haciendo las cosas. Si algo no funciona del todo bien: revíselo y mejórelo.

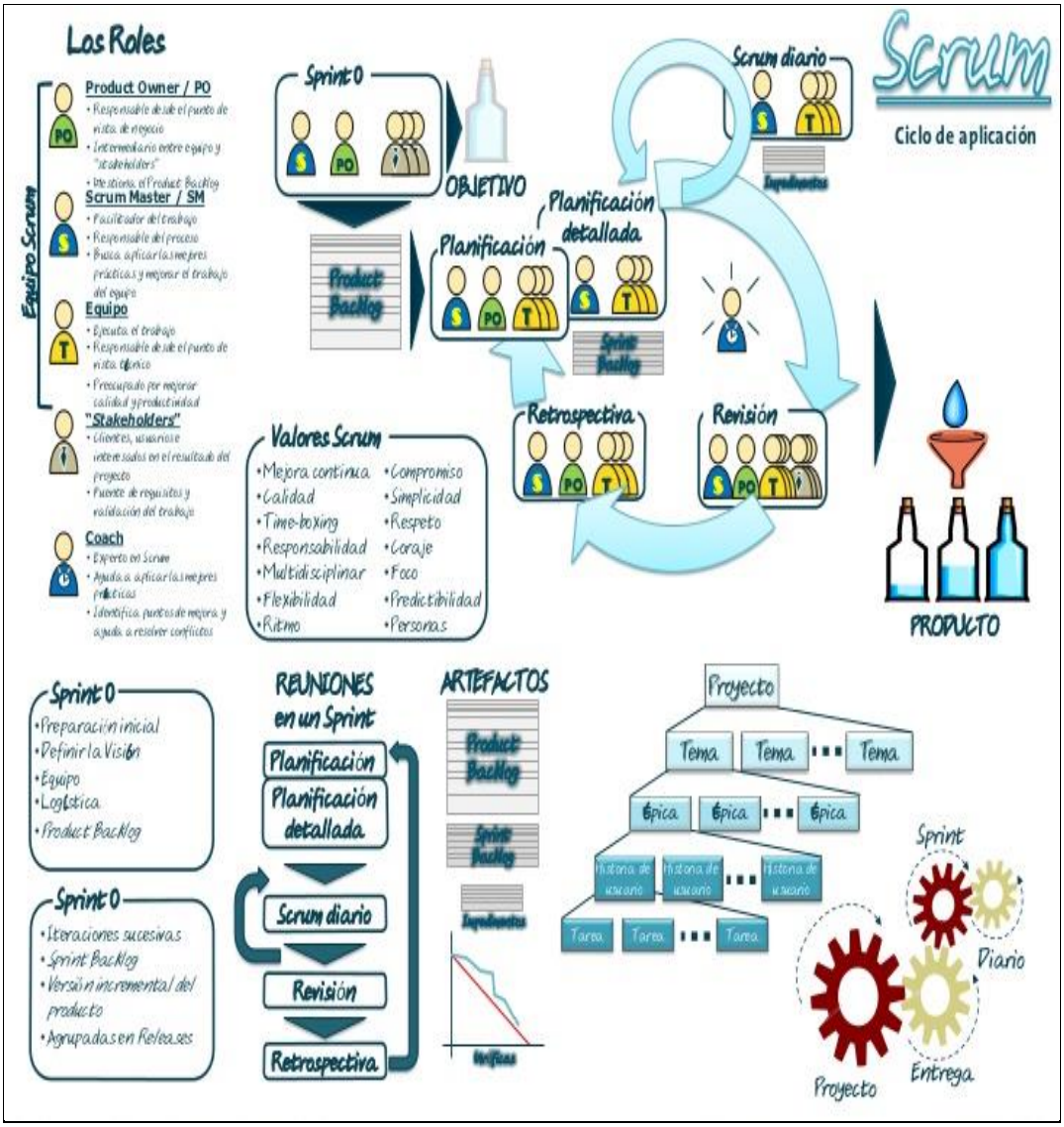
Las Retrospectiva nos ayuda enormemente a compartir diferentes puntos de vista, esta puesta en común nos ayuda a ver las cosas desde diferentes ángulos y a no dar nada por supuesto.

Se pueden realizar Retrospectiva en distintos momentos del proyecto: al final del *Sprint*, después de una *Release* o al final del proyecto.

De una forma muy resumida podemos decir que el objetivo final de una Retrospectiva es obtener de forma clara y concreta la siguiente información[41]:

- Que es lo que estamos haciendo bien (para celebrarlo y continuar trabajando de esta forma)
- Que otras cosas tenemos que mejorar o incluso en ocasiones dejar de hacer.
- Que vamos a hacer en la siguiente iteración teniendo en cuenta la información de los dos puntos anteriores.

([41]) Ibídem. Pág. 228.


Los Roles
Equipo Scrum
Product Owner / PO
• Responsable desde el punto de vista de negocio
• Intermediario entre equipo y "stakeholders"
• Mantenimiento del Product Backlog
Scrum Master / SM
• Facilitador del trabajo
• Responsable del proceso
• Busca aplicar las mejores prácticas y mejorar el trabajo del equipo
Equipo
• Ejecuta el trabajo
• Responsable desde el punto de vista técnico
• Preocupado por mejorar calidad y productividad
"Stakeholders"
• Clientes, usuarios o interesados en el resultado del proyecto
• Fuente de requisitos y validación del trabajo
Coach
• Experto en Scrum
• Ayuda a aplicar las mejores prácticas
• Identifica puntos de mejora y ayuda a resolver conflictos
Sprint 0
OBJETIVO
Product Backlog
Planificación
Planificación detallada
Sprint Backlog
Scrum diario
Ingredientes
Retrospectiva
Revisión
Scrum
Ciclo de aplicación
PRODUCTO
Valores Scrum
•Mejora continua
•Calidad
•Time-boxing
•Responsabilidad
•Multidisciplinar
•Flexibilidad
•Ritmo
•Compromiso
•Simplicidad
•Respeto
•Coraje
•Foco
•Predictibilidad
•Personas
Sprint 0
•Preparación inicial
•Definir la Visión
•Equipo
•Logística
•Product Backlog
Sprint 0
•Iteraciones sucesivas
•Sprint Backlog
•Versión incremental del producto
•Agrupadas en Releases
REUNIONES en un Sprint
Planificación
Planificación detallada
Scrum diario
Revisión
Retrospectiva
ARTEFACTOS
Product Backlog
Sprint Backlog
Ingredientes
Gráficas
Proyecto
Tema
Épica
Historia de usuario
Tarea
Sprint
Diario
Entrega
Proyecto

CAPITULO VII

METRICAS AGILES E IMPLEMENTACION

Sumario: 1.- Métricas ágiles. 2.- Claves para la implementación. 3.- Análisis de la implementación de metodologías ágiles en una empresa.-

1.- Métricas ágiles:

Es evidente que para poder establecer normas o tomar medidas para acentuar la calidad tenemos que ser capaces de medir el producto y el proceso. Este es el objetivo de las métricas de *software*. Según el IEEE, una métrica nos da una "medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo dado".

Existen cuatro razones para medir un proceso de software[42]:

1. Caracterizar
2. Evaluar

([42]) PRIOLO, Sebastián. Op. Cit. Pág 229.

3. Predecir
4. Mejorar

La evaluación no sólo se hace sobre el *software* sino también sobre los trabajadores, su desempeño, la cantidad de errores que se introducen, las diferencias de costos y muchas otras opciones.

Toda medición puede separarse en directa o indirecta. Las medidas directas son las que pueden ser cuantificadas (al menos en apariencia): el costo, el esfuerzo aplicado, la cantidad de líneas de código y los recursos humanos destinados. Hay infinidad de medidas directas y en general son más sencillas de mensurar que las indirectas, donde los elementos tienen un alto grado de subjetividad, como la calidad, la complejidad, la facilidad de uso, la funcionalidad, la capacidad de mantenimiento o la adaptabilidad, entre otras cosas. Las métricas del *software* se ubican en cualquiera de estos dos planos y tienen su propia división.

Medir un desarrollo de características totalmente distintas a los tradicionales implica repensar los atributos que se van a observar y las formas de hacerlo.

El equipo es consciente de que se lo está observando, es muy posible que su mayor esfuerzo se realice apuntando, justamente, a cumplir con los objetivos o a satisfacer esa medida, descuidando otros aspectos (no menos importantes). Para evitar este inconveniente, es recomendable utilizar un cuadro de mando ágil, comúnmente denominado ABS (*Agile Balanced Scorecard*), el ambiente ágil intenta dar valor al producto, y es por eso que la medida más importante que tenemos es la cantidad de funcionalidad agregada. Sin embargo, ésta no es la única métrica para aplicar. Dividimos a las métricas en cinco apartados.

- **Productividad**: Las métricas de productividad tienen por objeto valorar el agregado de funciones al producto y la respuesta al cliente. Se puede mencionar:
  - Cumplimiento de requisitos: medimos la cantidad de elementos que se nos requieren, los que entregamos y el tiempo en que se realiza la entrega.

  - Tiempo de respuesta: ante el cambio observamos el impacto en el equipo y el tiempo que transcurre hasta la entrega.
  - Manejo de prioridades: lapso que se pierde en acomodar tareas y realizarlas.

- **Resultados:** Para la obtención de valores de resultado dependemos en menor o mayor medida de la óptica y la opinión del cliente. Podemos mencionar:
  - Aporte de valor: respuesta que nos da el cliente sobre el producto.
  - Valor acumulado: este valor es un dato que debe ser estimado sobre el beneficio que el desarrollo le genera al negocio.
  - Total de proyecto terminado: porcentaje del proyecto que se ha finalizado de acuerdo a los requisitos actuales.
  - Días transcurridos: días de trabajo totales.
  - Días pendientes ideales: estimado sobre los días de trabajo restantes, tomando en consideración las planificaciones al momento de la medición.
  - Desviaciones: variación final del proyecto sobre la estimación inicial.
- **Situación Financiera:** Dentro de las métricas de situación financiera tenemos el ROI (*Return On Investment*, retorno de la inversión pendiente), además de la comparación entre el presupuesto aprobado y el total de recursos utilizados.
- **Calidad:** Las métricas de calidad apuntan al grado en que el cliente se encuentra satisfecho con el desarrollo. Además, comparamos el producto con los valores internos de calidad, contabilizando errores, entregas y todos aquellos elementos simples que se consideren convenientes. Existen tres tipos de calidad:

- Necesaria: la calidad necesaria es aquella que es solicitada por el cliente, ya sea para cubrir sus procesos y normas actuales o para obtener un producto superior.
- Programada: es la calidad que nosotros deseamos que el producto tenga.
- Realizada: es la calidad presente en el producto final.

- **Riesgos:** Dentro de este apartado podemos ubicar los peligros que se convirtieron en realidad e impactaron en el proyecto, los atrasos, los requisitos que nos crearon inconvenientes y no pudieron ser finalizados. Además, es posible integrarlos con todos aquellos elementos que son susceptibles de mejorar.

2.- Claves para la implementación:

Es importante conocer algunas de las claves que pueden hacer fracasar la metodología o su adopción.

- ✓ Selección y formación de los recursos: Los equipos de *Scrum* deben estar conformados por recursos humanos con alta capacitación, con características personales que les permitan adaptarse al desarrollo bajo la metodología. Es normal observar que se generen equipos en base a los profesionales libres dentro de la organización. A pesar de que esto puede dar resultado, es recomendable armar un equipo

adecuado a la metodología. Para esto se puede trabajar junto a los encargados de la selección de recursos, dictaminando algunas de las aptitudes y actitudes requeridas y necesarias.

- ✓ Realización de las reuniones: Las reuniones son críticas y deben ser llevadas a cabo aun cuando parezca que se está perdiendo valioso tiempo de desarrollo. Un equipo ordenado y con información suficiente desarrollará más y mejor.
- ✓ Confección de los documentos: De la confección de los documentos y la actualización o mantenimiento de ellos depende el tipo de compromiso que se tiene con la metodología y con el proyecto mismo. Siempre deben seguirse los pasos definidos por la metodología ya que éstos han sido testeados ampliamente en el ambiente real, obteniendo los resultados esperados.
- ✓ Recursos delicados: En el estado ideal de aplicación de la metodología, los recursos empleados deben estar enfocados completamente en el proyecto. Se debe evitar tener personas trabajando en más de un proyecto al mismo tiempo. En el caso de que no sea posible que se dediquen a un solo proyecto, es recomendable que las tareas o proyectos que se realicen presenten similitudes, para que el conocimiento y el estado adquirido en uno puedan hacer más fácil las tareas en los otros.

3.- Análisis de la implementación de metodologías ágiles en una empresa:

Censys S.A es una empresa tucumana dedicada al desarrollo, comercialización, implementación y mantenimiento de programas de *Software* para entidades financieras, radicada desde 1976.

Debido a los cambios en el mercado, esta empresa llevo a cabo la transición de una metodología Tradicional a una metodología Ágil para responder así, a la demanda y los requerimientos de los clientes.

Inicialmente utilizaban una metodología en cascada, la cual consiste en un enfoque metodológico que ordena rigurosamente las etapas del proceso para el desarrollo del Software, de tal forma que el inicio de cada etapa debe esperar la finalización de la etapa anterior.

La transición a una metodología ágil se realizó de manera gradual a través de pruebas piloto con un equipo de trabajo y un cliente en particular, para después extenderse hacia el resto de los clientes.

La empresa adopta el tipo de metodología *Scrum* y *Kanban*, dependiendo las necesidades de cada cliente y de las particularidades de cada proyecto.

Para iniciarse en el agilismo con *Scrum*, se contrató un consultor especializado, con experiencia en empresas multinacionales. El profesional trabajó en la implementación y actualmente está a cargo del perfeccionamiento y capacitación de los participantes.

El cambio fundamental se produjo en la cultura de la organización, modificando una estructura predominada por lo jerárquico y formal, por una modalidad más flexible, colaborativa y en equipo.

Censys cuenta con recursos humanos muy valiosos, que tenían que reaprender la forma de trabajar, lo que origino el cambio cultural en el que una vez que las personas entendieron la importancia y la necesidad de trabajar de otra manera, se fue ajustando rápidamente.

En esta empresa el cambio es algo natural, lo que en otras puede llevar años o no se puede realizar; en Censys existe una predisposición por parte de las personas al cambio y mejora continua, contando siempre con el apoyo del directorio.

Los indicadores de que la metodología ágil está funcionando correctamente son: incremento de la rentabilidad, incremento de la capacidad productiva, disminución de tasa de errores, disminución de rotación de recursos, entre las más destacadas. La empresa se reserva las estadísticas.

Gracias a los métodos ágiles se logró satisfacer el incremento en la demanda de los clientes y actualmente se cuenta con una capacidad ociosa para situaciones especiales.

Censys fundamenta su permanencia en el mercado, a través de contratos relacionales con sus clientes, con los que sostiene compromisos que van más allá de la palabra escrita, y que se basa en la construcción permanente de la confianza. Contratos que permiten a la organización comportarse en forma innovadora y responsable con visión de futuro.

Actualmente en Argentina no existe un organismo que certifique el funcionamiento ágil, pero desde Censys nos informan que es una empresa que podría someterse a una certificación de estas características.

CONCLUSION

Como se pudo observar en el presente trabajo, en la empresa Censys la implementación de la Metodología Ágil, *Scrum,* generó cambios que se reflejaron principalmente en la cultura organizacional, donde los protagonistas son los equipos de trabajo, quienes deben poner en práctica el aprendizaje continuo, la adaptabilidad y la responsabilidad en la obtención de resultados conjuntos, sin dejar de lado la constante retroalimentación de los clientes, creando así una relación a largo plazo que genere acuerdos favorables.

El mercado está cambiando, volviéndose más dinámico, ágil y demandante, con el fin de que los procesos productivos sean más eficientes y las empresas se puedan adaptar con rapidez a este escenario y para satisfacer oportunamente las expectativas de los clientes, en el campo del desarrollo del *Software* se elige prioritariamente la metodología *Scrum,* ya que ofrece un marco de trabajo con procesos y técnicas que si bien son complejas de implementar son populares por los resultados rápidos y óptimos.

ANEXO

Entrevista realizada al Licenciado en Administración de empresas Alejandro Paez, Gerente de Organización y Procesos de la Empresa Censys.

1- **Para entrar en materia nos puede contar sobre usted y su experiencia en la empresa Censys.**

Censys S.A es una empresa argentina dedicada al desarrollo, comercialización, implementación y mantenimiento de soluciones CORE, para entidades financieras desde 1976.

Mi puesto es el de gerente de Organización y Procesos, que es la gerencia que se encarga de definir todos los circuitos de trabajo, definir las metodologías de trabajo que se van a llevar adelante, anteriormente teníamos implementada la metodología en cascada y actualmente utilizamos una metodología ágil, *Scrum* y *Kanban*.

Somos más de 110 personas que atienden una cartera de 12 clientes en donde tenemos que administrar y coordinar 12500 hs. mensuales de programación, nuestra área se encarga de estar velando por los procesos de trabajo para que los tiempos de trabajo se cumplan para tratar de ser lo más eficientes y eficaces posibles, porque ante una demanda que va incrementándose y un proceso ineficiente lo que se termina afectando es la rentabilidad, lo que hay que hacer es mantener los procesos y los circuitos de trabajo bien aceitados.

Lo que también se encarga esta gerencia es la parte de certificación de normas de gestión.

Actualmente esta área ha empezado a asumir algunas actividades de recursos humanos, teníamos tercerizados la parte de reclutamiento de personal a través de una consultora, pero como no hemos obtenido los resultados deseados hemos empezado nosotros a incorporar esas actividades en nuestra área.

Empecé con los sistemas de gestión en mi anterior trabajo, formaba parte de la secretaría de Estado de planeamiento, después del Ministerio de desarrollo productivo y luego estuve en el Ministerio de seguridad ciudadana, lo que yo venía realizando era la definición de procesos de trabajo en las distintas áreas y distintos

proyectos, en cada ministerio o secretaria de estado se definía cuáles eran los procesos o áreas más ineficientes y sobre eso se trabajaba, también definíamos la certificación de normas para el proceso principal de trabajo de la repartición o proyecto más afectado.

¿Se diferenciaba por certificar esas normas?

En ese momento se pudo lograr que el estado provincial tenga la primera repartición pública certificando su proceso principal con la norma ISO 9001 y de ahí se fueron incorporando otras áreas, no menos de 10 reparticiones fueron las que lograron sus certificaciones en ese momento.

Luego me pase a la parte privada donde se trabajaba con la metodología en cascada, en donde no habían normas de calidad certificada y ningún tipo de norma gestión certificada y cuando la empresa decide incorporarse a los beneficios de la ley de *software* uno de los requisitos que se exige es el de tener certificada a la empresa con las normas de gestión de calidad, entonces desde el año 2011 hacia la actualidad hemos certificado dos normas ISO, la 9001 y la 27001 y las mantenemos actualizadas y recertificadas.

2- ¿Cuál fue el elemento o situación por el cual decidieron pasar de una metodología tradicional a una metodología ágil?

Actualmente el mercado está cambiando, se volvió más dinámico y ágil, más demandante y eso provocaba que a nuestros procesos productivos debamos mantenerlos actualizados para poder responder a esa demanda, ya que si no se respondía a esa demanda se perderían clientes.

3- ¿Qué tipo de metodología ágil implementaron?

Empezamos a evaluar distintas formas de trabajo , revisamos cual era nuestro negocio y observamos que la metodología *Scrum* era la que mejor se adaptaba a nuestras necesidades y a la de nuestros clientes, después con la experiencia pudimos observar que había ciertos tipos de proyectos o unidades de negocios que se manejaban mejor con otro tipo de metodología ágil que en este caso es *Kanban*, todas las

metodologías se aplican sobre los procesos principales de trabajo, el proceso principal de Censys es el de diseño, desarrollo y mantenimiento de *software* bancario, y la certificación de los sistemas de gestión se aplican sobre los procesos principales de trabajo.

4- ¿Cuáles son los principales problemas o dificultades a lo que se enfrentaron al realizar el cambio de metodología?

Cuando se decidió pasar de una metodología en cascada a una metodología ágil, con lo que primero que nos encontramos es que no se trató sólo de un cambio metodológico de trabajo, no es simplemente una actividad que se hacía de un manera y ahora se hace de otra forma, lo que requiere antes que nada es un cambio cultural, porque las metodologías agiles fomentan el trabajo en equipo, la autogestión y la definición de objetivos por unidades de negocios, en la metodología en cascada se tiene una estructura piramidal claramente definida en donde las actividades se van desarrollando a medida que va bajando el nivel de detalle y se va llegando a actividades más operativas, en la metodología ágil la pirámide se invierte, en donde los actores principales son los equipos de trabajo que interactúan directamente con el cliente, equipos conformados por personas con distintos roles y como áreas de soporte estaban las estructuras jerárquicas que eran las más altas, todo lo que era gerencia pasa a ser soporte en todos los equipos de trabajo para garantizar una mejor atención a los clientes.

5- ¿Cuándo se decide pasar de una metodología tradicional a una ágil por donde se empieza?

El cambio fue gradual, se hicieron pruebas piloto con un equipo de trabajo y un cliente en particular y después se fue llevando hacia el resto de los clientes.

6- ¿Cuánto tiempo llevo la transición?

Cuando se inicia este tipo de cambio nunca terminan, nuestros procesos de trabajo se están adecuando constantemente, lo que buscan las normas de gestión es la

mejora continua, y evidenciar no es tarea sencilla hay distintas herramientas que se usan, siempre estamos buscando la manera de ser cada vez más eficientes y más eficaz.

7- ¿Que nos puede decir acerca de la selección y formación del personal tanto de jefes de la empresa como del resto de los miembros de los equipos?

La particularidad que siempre destacó a Censys, son sus talentos, por el nivel de conocimientos que poseen, no muchas personas conocen sobre bancos y financieras, sobre sus productos, servicios, formas de trabajo, principios, valores.

En la empresa se encontraban personas con muchos conocimientos teóricos pero que tenían que reaprender la forma de trabajar, esto provoco un profundo trabajo de cambio cultural que una vez que las personas entendieron la importancia y la necesidad de trabajar de otra manera se fue ajustando rápidamente.

8- En algún momento se observó resistencia por parte de éstos.

Siempre se observan personas que se resisten al cambio, Censys tiene 40 años y hay talentos que tienen 38 años de actividad, había personas que llevaban muchos años trabajando de una manera y de repente se les pidió que cambien la forma en que venían realizando sus actividades.

Para la gente que conforma Censys el tema del cambio es algo natural, lo que en otras empresas puede llevar años o no se puede dar, aquí cuando se pide que se debe cambiar o mejorar algo la gente se suma, porque además se cuenta con el apoyo del directorio, desde la cabeza de la organización hasta el último equipo de trabajo.

9- ¿Cuáles considera que son buenos indicadores de que la metodología ágil está funcionando?

Incremento de la rentabilidad, incremento de la capacidad productiva, disminución en las tasas de errores, disminución en la rotación de recursos.

10- ¿Cómo reaccionaron los clientes ante el cambio de metodología? ¿ A que hace referencia los contratos relacionales con los clientes?

Gracias a las metodologías agiles hemos logrado atender el incremento en la demanda de nuestros clientes actuales, ya que ellos venían demandando mayor esfuerzo que no podíamos atender, hoy hemos podido atender ese incremento y además tener una capacidad de sobra por si vuelve a incrementarse, también se logró incorporar nuevos clientes.

Nuestros clientes tienen con Censys una relación de más de 18 años, es muy importante crear la relación a largo plazo cliente- proveedor, en donde se debe llegar siempre a acuerdos contractuales que sean favorables para ambas partes, se generan vínculos muy fuertes de hecho los números y los años que llevamos con nuestros clientes lo ponen en evidencia.

Censys fundamenta su permanencia en el mercado, a través de contratos relacionales con sus clientes, con los que sostiene compromisos que van más allá de la palabra escrita, y que se basan en la construcción permanente de la confianza. Contratos que permiten a la organización comportarse en forma innovadora y responsable, con visión de futuro, de manera confiable para:

- Mantener sistemas de trabajo de alto performance.
- Disposición permanente aportar soluciones.
- Permanecer al lado de los usuarios en los momentos buenos, y en los difíciles.
- Invertir en iniciativas de largo plazo sin desatender las necesidades urgentes.

11- ¿Poseen alguna Certificación Agile?

No hay un organismo que certifique el funcionamiento agile en Argentina, pero tranquilamente podríamos someternos a una certificación de estas características. En el aspecto agile no existe un organismo de referencia como sucede en el caso de certificar normas de calidad en donde existen organismos de referencia como IRAM.

12- ¿Qué lineamientos siguen o siguieron para aplicar *Scrum*?

Se contrató un consultor especializado, que trabajó en la implementación y trabaja hasta el día de hoy en el tema de perfeccionamiento constante, es una persona que ya había trabajado en empresas multinacionales y que tenía numerosos casos de éxito, nos acompañó en todo el proceso.

13- En general, ¿Qué nivel de madurez cree usted que en Tucumán se ha alcanzado respecto a la adopción de agile en nuestras organizaciones?

Son muy pocas la empresas que trabajan con metodologías ágiles, y las que lo tienen implementado no son empresas de aquí, por ejemplo Globant que es una empresa que nació en argentina pero cotiza en bolsa, después hay empresas de desarrollo de *software* pero son pequeñas de 10 a 15 personas, aquí en Tucumán no hay empresas grandes que trabajen con metodologías agiles. Las metodologías ágiles no son privativas de la industria del *Software*, se pueden aplicar en cualquier industria o procesos de trabajo. La industria automotriz es la cuna de la metodología ágil, Toyota para ser más preciso. Las metodologías ágiles es un tema relativamente en Argentina.

14- Diferencia entre *Scrum* y *kanban*

En Scrum las tareas se repiten en ciclos, los ciclos se denominan iteraciones y éstas son de 2 semanas de trabajo generalmente, en *scrum* se observan ciclos de vida donde al final se compromete la entrega de algo y en donde ese equipo de trabajo está haciendo algo puntualmente para un cliente, en *Kanban* no se observa este ciclo repetitivo sino que existe una planificación de lo que hay q realizar y esa actividad se puede aplicar a otros clientes, es decir se puede hacer un desarrollo pero que no es privativo de un solo cliente sino que se puede hacer para otros clientes, por ejemplo, nosotros tenemos un equipo que se encarga de atender todo lo que es la normativa del banco central, donde el banco saca una normativa que establece que este tipo de operaciones se deben reportar de una determinada forma, yo podría hacer q cada equipo *kanban* haga un desarrollo para cumplir con esta normativa pero sería ineficiente porque tendría a más de un equipo trabajando en un mismo tema, entonces se establece

que un equipo de trabajo se encarga de hacer el desarrollo y que se puede aplicar a todos los temas y que no tengo que hacer un ciclo iterativo con una pre entrega sino que tengo que llegar a determinada fecha para cumplir con la normativa. Un equipo que trabaja para todo los clientes, no tengo a todos los equipos trabajando en la misma temática.

15- Generalmente cual es tamaño de los equipo en Censys?

Hay diferentes tamaños de equipos no pueden ser más de 15 personas y no menos de 5 personas, generalmente tenemos como máximo 14 personas en nuestros equipos.

16- En el proceso de selección ¿cuáles son las competencias que se buscan en el personal?

Nuestro proceso de selección incluye varias etapas, en una primera etapa nosotros nos encargamos del reclutamiento, hoy en día contamos con una gran base de datos de personas que quieren formar parte de la empresa, de todas las carreras y universidades, esta base se fue creando a partir de personas que se fueron contactando con la empresa para que brindemos información, para realizar trabajos y seminarios y también personas que fuimos conociendo en nuestros recorridos brindando charlas en distintos establecimientos incluso escuelas secundarias y nos dio un gran volumen de postulantes de ciencias económicas, ingeniería, abogacía, etc.

En función del puesto que tenemos que buscar realizamos una entrevista que se denomina técnica, donde el responsable del equipo junto con los técnicos entrevistan a una persona para evaluar si tiene los conocimientos teóricos necesarios para formar parte del equipo, la siguiente etapa es la de evaluar las competencias de estas personas donde lo hacemos a través de una psicóloga, evaluamos si es una persona adaptable, trabaja en equipo, orientada a trabajar por resultados, esto es al inicio para seleccionar a una persona y después durante la vida de trabajo de una persona están siendo evaluadas constantemente mediante capacitaciones, por ejemplo una persona que

ingreso como programador hoy es un líder técnico porque fue mejorando sus competencias y habilidades y desarrollando otras nuevas que fueron necesarias.

17- ¿Quién es el personal que crece más rápido en Censys?

Los que tienen más habilidades de gestión, los que son más extrovertidos, que saben trabajar en equipo, empáticos, saben interactuar con el cliente.

18- Para finalizar tiene algún consejo para aquellos que quieran implantar agile en sus empresas.

Lo que yo aprendí en esta empresa, o a lo mejor ya lo tenía aprendido pero lo termine de confirmar en Censys, es estar buscando el cambio constantemente, en el momento en el cual crees que llegaste a lo mejor que podías estar, es cuando es el momento para iniciar el cambio, nosotros tranquilamente podemos decir llegamos al máximo de nuestra capacidad teórica de producción y hoy estamos x buscar de mejorar los procesos comerciales de Censys, buscamos ver de qué manera generamos mayor demanda a nuestros clientes o a nuevos clientes , eso obviamente llevara a decir ahora hay que mejorar los procesos operativos nuevamente porque incremento esta demanda.

INDICE BIBLIOGRAFICO

A) GENERAL

LAUDON, Kenneth y LAUDON, Jane, "Sistemas de información gerencial", Pearson Educación, (México 2012).

SOMMERVILLE, Ian, "Ingeniería del Software", trad. por. Víctor Campos Olguín, 9º Edición, Addison Wesley, (México 2011).

STAIR, Ralph y REYNOLDS, George, "Principios de Sistemas de Información", Cengage Learning Editores, (México 2010).

B) ESPECIAL

ALAIMO, Diego Martín, "Proyectos agiles con Scrum: flexibilidad, aprendizaje, innovación y colaboración en contextos complejos", Ediciones Kleer, (Ciudad Autónoma de Buenos Aires 2013).

ALVAREZ GARCÍA, Alonso, DE LAS HERAS DEL DEDO, Rafael, y LASA GÓMEZ Carmen, "Métodos Agiles y Scrum", Grupo Anaya S.A. (Madrid 2012).

BECK, Kent, Extreme Programming Explained, Addison-Wesley. (1999)

HUNT Andrew y THOMAS David, The Pragmatic Programmer. Addison Wesley (2000).

KNIBERG, Henrik and SKARING Mattias, Kanban and Scrum Making the most of Both, Enterprise Software Development Series (2010).

LEFFINGWELL Dean, Scaling Software Agility Best Practices for Large Enterprises. Series Editors. (2007).

PALMER Stephen and FELSING John, A Practical Guide to Feature-Driven Development, The Coad Series (2002).

POPPENDIECK, Mary and Tom, Lean Software Development An Agile Toolkit, Series Editors. 2003.

PRIOLO, Sebastián, "Métodos agiles", Gradi S.A. (Buenos Aires 2009).

C) OTRAS PUBLICACIONES

Consultas a bases de información, en Internet: https://agilemanifesto.org/

Consultas a bases de información, en Internet: https://proyectosagiles.org /2009/04/09/buen-gestor-equipo-agil-scrummaster//www.proyectosagiles.org

Consultas a bases de información, en Internet: https://proyectosagiles.org/category/como_empezar-transformacion-agile/

Consultas a bases de información, en Internet: https://proyectosagiles.org/2008/11/16/contrato-agil-scrum/

Consultas a bases de información, en Internet: www.scrumalliance.org/articles/36-leader-of-the-band.

INDICE

CAPITULO III

GESTION AGIL

CAPITULO IV

TIPOS DE METODOLOGIAS AGILES

CAPITULO V

SCRUM

CAPITULO VI

PROCESO DE SCRUM

CAPITULO VII

METRICAS AGILES E IMPLEMENTACION1