Scrum Manager

Gestión de proyectos

Rev. 1.4

Scrum Manager
Gestión de Proyectos

Formación

Rev.1.4.0 Enero-2011

Título

Scrum Manager Gestión de Proyectos

Autores

Juan Palacio, Claudia Ruata

Imagen de Portada

Maurice de Beijer.

Revisiones y ampliaciones

1.0.- Enero de 2009
1.1.- Mayo de 2009
1.2.- Julio de 2009
1.3.- Octubre de 2009
1.3.1 – Julio de 2010
1.3.2 – Ocbubre de 2010
1.4.0 – Enero de 2011

Más información, y última versión en: http://www.scrummanager.net

Derechos

Contenido

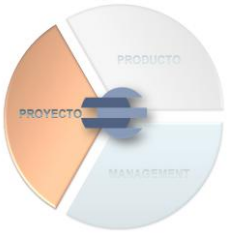
Prólogo

Apuntes de formación Scrum Manager

Este es el libro de texto para la formación en el área de “Proyecto” del marco de gestión “ScrumManager®”.

Es un recurso educativo abierto (OER) y forma parte de la plataforma Open Knowledge Scrum: http://scrummanager.net/ok/

Se puede emplear de forma gratuita para consulta y auto-formación a título personal.



Plataforma abierta para consulta y formación profesional Scrum Manager

Open Knowledge Scrum es una plataforma de acceso libre para consulta y formación, está disponible en http://scrummanager.net/ok/ donde encontrarás la última versión de este curso, además de otros materiales, foros, talleres, etc.

Un punto abierto en la Red para consultar y compartir conocimiento, y mantenerte profesionalmente actualizado.

Servicios profesionales para formación y asesoría Scrum Manager

Puedes localizar profesionales y empresas certificadas para servicios profesionales de formación y asesoría en la implantación y mejora de Scrum Management, en el directorio de centros de formación autorizados Scrum Manager

http://scrummanager.net/ o solicitar información en la dirección formacion@scrummanager.net

Información para formar parte de la red de centros certificados Scrum Manager en la dirección formacion@scrummanager.net.

Agradecimientos

Nuestro más sincero agradecimiento a los miembros y colaboradores de Scrum Manager por las sugerencias y aportaciones ofrecidas tanto en los cursos abiertos como en los de la plataforma Open Knowledge (http://www.scrummanager.net/ok).

Y en especial a Angel Águeda Barrero porque la mayor parte de las mejoras que incorpora esta edición no serían posibles sin su incansable y continua aportación.

A todos, gracias por hacer posible la mejora continua de este libro.

Los autores

Introducción

Caos, procesos, agilildad...

El origen: la crisis del software

El desarrollo de software es una actividad compleja y reciente, que ha generado su conocimiento en un periodo muy breve, en comparación con otras actividades profesionales: desde la aparición de máquinas que para ser útiles necesitaban ser programadas.

Ilustración 1 - 1967 Ampex Instant Replay Disk Recorder

La aparición de componentes que cada dos años doblan la capacidad de sus antecesores [ley de Moore] nos ha rodeado en menos de cuatro décadas de máquinas capaces de procesar miles de millones de operaciones por segundo (MTOPS).

En 1946 ENIAC ocupaba una superficie de 160 m2, pesaba 30 toneladas, y ofrecía una capacidad de proceso de 30.000 instrucciones por segundo. En 2002 El microprocesador Pentium IV a 2 Ghz ocupaba una superficie de 217 mm2 y tenía una capacidad de proceso de 5.300 MTOPS (“Millions of theoretical operations per second)

En la actualidad son cuatro los factores que imprimen un ritmo acelerado a la industria del hardware.

De ellos, tres son consecuencia de la ley de Moore:

- Incremento constante de la capacidad de operación.
- Miniaturización.
- Reducción de costes para la producción de hardware.

y a éstos se ha sumado en la última década

- el avance de las comunicaciones entre sistemas.

La consecuencia es obvia: ordenadores potentes, que pueden llevarse en el bolsillo y en permanente conexión con grandes sistemas, redes de comunicación públicas, sistemas de localización GPS, etc.

Estas cuatro líneas de avance han extendido el ámbito de aplicación del hardware, e incrementado al mismo ritmo exponencial la complejidad de los sistemas en los que se integra. Los ordenadores ya no son máquinas útiles sólo para la banca o el ejército. Se encuentran presentes en todos los ámbitos, por su capacidad de proceso y de comunicación pueden ofrecer soluciones a sistemas cada vez más complejos.

Este es el escenario creado por la industria del hardware, y que en las tres últimas décadas ha implicado a los desarrolladores de software en retos a los que no han respondido con solvencia.

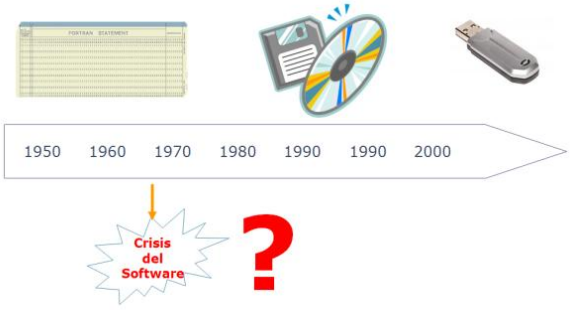


Ilustración 2 - 1968 Crisis del Software

Crisis del software

Este problema se identificó por primera vez en 1968 (Bauer, Bolliet, & Helms, 1968), año en el que la organización OTAN celebró la primera conferencia sobre desarrollo de software, y en la que se acuñó el término “crisis del software” para definir a los problemas que surgían en el desarrollo de sistemas de software, cuyos proyectos siempre terminaban tarde, desbordando los presupuestos y con problemas de funcionamiento. También se acuñó el término “ingeniería del software” para describir el conjunto de conocimientos que existían en aquel estado inicial.

Nuestra profesión es muy reciente, se podría decir que aún adolescente (si no una niña) y que no cuenta todavía con una base de conocimiento maduro.

Cuando en los 60 y sobre todo en los 70 y 80 empezaron a hacerse habituales los ordenadores, surgieron los primeros “héroes,” que sin más información que los manuales del operativo y el lenguaje de programación, se remangaban delante del teclado para desarrollar las primeras aplicaciones.

Hasta los 70 los ordenadores fueron máquinas vanguardistas y excesivamente caras. El ejército y la banca eran los únicos sectores que se las permitían, y fueron los militares los primeros escarmentados, de proyectos en los que el software siempre llegaba tarde, mal y nunca; con demasiados errores y desbordando todas agendas previstas.

Algunas referencias útiles para comprender cuáles eran los conocimientos estables para el desarrollo de software en 1968 son:

- En 1962 se publicó el primer algoritmo para búsquedas binarias (Iverson, 1962).
- C. Böhm y G. Jacopini publicaron en 1966 el documento que creaba una fundación para la eliminación de "GoTo" y la creación de la programación estructurada (Böhm & Jacopini, 1966)
- En 1968 los programadores se debatían entre el uso de la sentencia GoTo, y la nueva idea de programación estructurada; ese era el caldo de cultivo en el que Edsger Dijkstra escribió su famosa carta "GoTo Statement Considered Harmful" en 1968 (Dijkstra, 1968).
- La primera publicación sobre programación estructurada no vio la luz hasta 1974, publicada por Larry Constantine, Glenford Myers y Wayne Stevens (Stevens, Myers, & Constantine, 1974).
- El primer libro sobre métrica de software fue publicado en 1976 por Tom Gilb (Gilb, 1976).
- El primero sobre análisis de requisitos apareció en 1976 (Bell & Thayer, 1976)

La Tesis: Ingeniería, gestión predictiva y procesos

Desde 1968 hasta la fecha han sido muchos los esfuerzos realizados por los departamentos de informática de las universidades, y por organismos de estandarización y asesoría para identificar las causas del problema y definir pautas estándar para la producción y mantenimiento del software.
Los esfuerzos desarrollaron en las tres últimas décadas del siglo pasado tres áreas de conocimiento que se revelaron como estratégicas para hacer frente a la crisis del software:

- Ingeniería del software
- Gestión predictiva de proyectos
- Producción basada en procesos.

Ilustración 3 - Criterios de la tesis

INGENIERÍA DEL SOFTWARE

Término acuñado en la conferencia de la OTAN de 1968, al definir la necesidad de una disciplina científica que, como ocurre en otras áreas, permita aplicar un enfoque sistemático, disciplinado y cuantificable al desarrollo operación y mantenimiento del software; es decir, la aplicación de la ingeniería al software.

El proyecto más consensuado hasta la fecha para definir las áreas de conocimiento que comprenderían una ingeniería del software es el SWEBOK.

GESTIÓN DE PROYECTOS PREDICTIVA

La necesidad de profesionalizar la gestión de proyectos surgió en los 50, en el ámbito militar, para abordar el desarrollo de complejos sistemas militares que requería coordinar el trabajo conjunto de equipos y disciplinas diferentes, en la construcción de sistemas únicos.

La industria del automóvil siguió los pasos de la militar, aplicando técnicas de gestión de proyectos para la coordinación del trabajo entre áreas y equipos diferentes.

Comenzaron a surgir técnicas específicas, histogramas, cronogramas, los conceptos de ciclo de vida del proyecto o descomposición en tareas (WBS Work Breakdown Structure).

La gestión de proyectos predictiva o clásica es una disciplina formal de gestión, basada en la planificación, ejecución y seguimiento a través de procesos sistemáticos y repetibles.

En los 80 se definieron los objetivos que la gestión de proyectos debía cumplir para poder considerar que el trabajo concluye con éxito:

- Se ejecuta en el tiempo planificado.
- Sin desbordar el presupuesto estimado.

- Satisfaciendo las necesidades del cliente
  - Realiza las funcionalidades que necesita.
  - Las realiza correctamente y sin errores.

En la actualidad, según el estudio periódico, que desde 1994 realiza Standish Group, escasamente uno de cada tres proyectos de desarrollo de software termina en éxito.

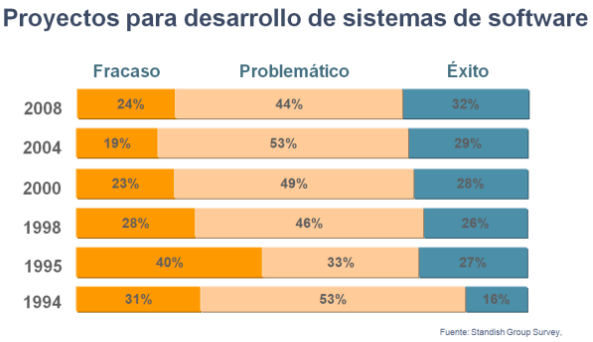


Ilustración 4 - Informes Standish Group

Características de la gestión de proyectos predictiva

- Establece como criterios de éxito: obtener el producto definido, en el tiempo previsto y con el coste estimado.
- Asume que el proyecto se desarrolla en un entorno estable y predecible.
- El objetivo de su esfuerzo es mantener el cronograma, el presupuesto y los recursos.
- Divide el desarrollo en fases a las que considera "ciclo de vida", con una secuencia de tipo: concepto, requisitos, diseño, planificación, desarrollo, cierre.

PRODUCCIÓN BASADA EN PROCESOS

Sobre el principio de calidad de Jurán (Juran, 1951), empleado con buenos resultados en los procesos de producción industrial: "La calidad del resultado depende básicamente de la calidad de los procesos empleados en su producción", se desarrollaron también para la industria del software modelos de procesos (ISO 90003, CMMI, ISO 15504...) para que las empresas puedan alcanzar los cuatro beneficios clave de la producción basada en procesos:

- Repetibilidad de resultados. Al conseguir que la calidad del resultado sea consecuencia del proceso, producir aplicando el mismo proceso garantiza la homogeneidad de los resultados.
- Escalabilidad. Es una consecuencia de la repetibilidad. No sólo un equipo consigue resultados homogéneos en todos los proyectos, sino que los obtienen todos los equipos.
- Mejora continua. Al aplicar meta-procesos que trabajan sobre los propios procesos de producción, midiendo y analizando los resultados se obtienen los criterios de gestión necesarios para aplicar medidas que mejoran de forma continua la eficiencia y calidad de los procesos base, y por tanto de los resultados.
- Un know-how propio, consiguiendo finalmente una empresa que sabe hacer, porque su modelo de procesos termina conteniendo un activo valioso de la organización: el conocimiento clave para hacer las cosas bien, con eficiencia y de forma homogénea.

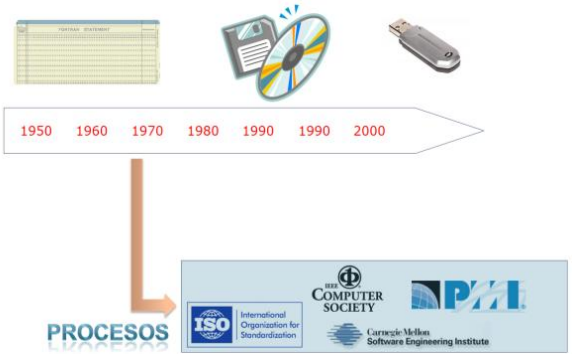


Ilustración 5 - Referentes de conocimiento para gestión de proyectos basados en procesos

La Antítesis: Agilidad

¿El modelo predictivo es el único posible?

¿Los criterios para determinar el éxito sólo pueden ser el cumplimiento de fechas y costes?

¿Puede haber proyectos que no tengan como finalidad realizar un trabajo previamente planificado, con un presupuesto y en un tiempo previamente calculados?

¿Y si el cliente no estuviera interesado en saber si el sistema tendrá 20 ó 200 funcionalidades, si estará en beta 6 meses o 2 años?

¿Si su interés fuera poner en el mercado antes que nadie un producto valioso para los clientes, y estar continuamente desarrollando su valor y funcionalidad?

Quizá en algunos proyectos de software el empeño en aplicar prácticas de estimación, planificación, ingeniería de requisitos sea un empeño vano. Quizá la causa de los problemas no sea tanto por una mala aplicación de las

prácticas, sino por la aplicación de prácticas inapropiadas.

Quizá se estén generando "fiascos" al exigir a los clientes criterios de adquisición, y al aplicar a los proyectos procesos de gestión predictivos, cuando se trata de proyectos que no necesitan tanto garantías de previsibilidad en la ejecución, como valor y flexibilidad para trabajar en un entorno cambiante.

Ilustración 6 - Manifiesto ágil

En marzo de 2001, 17 críticos de los modelos de mejora basados en procesos, convocados por Kent Beck, que había publicado un par de años antes el libro "Extreme Programming Explained" en el que exponía una nueva metodología denominada Extreme Programming, se reunieron en Salt Lake City para discutir sobre el desarrollo de software.

En la reunión se acuñó el término "Métodos Ágiles" para definir a los que estaban surgiendo como alternativa a las metodologías formales: CMM-SW (Paulk, B., Chrissis, & Weber, 1996) precursor del CMMI, PMI, SPICE (proyecto inicial de ISO 15504), etc. a las que consideraban excesivamente "pesadas" y rígidas por su carácter normativo y fuerte dependencia de planificaciones detalladas, previas al desarrollo.

Los integrantes de la reunión resumieron en cuatro postulados lo que ha quedado denominado como "Manifiesto Ágil", que son los principios sobre los que se basan estos métodos.

Hasta 2005, entre los defensores de los modelos de procesos y los de modelos ágiles han sido frecuentes las posturas radicales, quizá más ocupadas en descalificar al otro, que en estudiar sus métodos y conocerlos para mejorar los propios.

Manifiesto Ágil

Estamos poniendo al descubierto mejores métodos para desarrollar software, haciéndolo y ayudando a otros a que lo hagan. Con este trabajo hemos llegado a valorar:

- ***A los individuos y su interacción, por encima de los procesos y las herramientas.***
- ***El software que funciona, por encima de la documentación exhaustiva.***
- ***La colaboración con el cliente, por encima de la negociación contractual.***
- ***La respuesta al cambio, por encima del seguimietno de un plan.***

Aunque hay valor en los elementos de la derecha, valoramos más los de la izquierda

Valoramos más a los individuos y su interacción que a los procesos y las herramientas.

Este es posiblemente el principio más importante del manifiesto.

Por supuesto que los procesos ayudan al trabajo. Son una guía de operación. Las herramientas mejoran la eficiencia, pero en trabajos que requieren conocimiento tácito, sin personas con conocimiento técnico y actitud adecuada, no producen resultados.
Las empresas suelen predicar muy alto que sus empleados son lo más importante, pero la realidad es que en los años 90 la teoría de producción basada en procesos, la reingeniería de procesos ha dado a éstos más relevancia de la que pueden tener en tareas que deben gran parte de su valor al conocimiento y al talento de las personas que las realizan.

Los procesos deben ser una ayuda y un soporte para guiar el trabajo. Deben adaptarse a la organización, a los equipos y a las personas; y no al revés. La defensa a ultranza de los procesos lleva a postular que con ellos se pueden conseguir resultados extraordinarios con personas mediocres, y lo cierto es que este principio es peligroso cuando los trabajos necesitan creatividad e innovación.

Valoramos más el software que funciona que la documentación exhaustiva.

Poder ver anticipadamente cómo se comportan las funcionalidades que se esperan sobre prototipos o sobre partes ya elaboradas del sistema final ofrece un "feedback" muy estimulante y enriquecedor que genera ideas y posibilidades imposibles de concebir en un primer momento, y difícilmente se podrían incluir al redactar un documento de requisitos detallados antes de comenzar el proyecto.

El manifiesto no afirma que no hagan falta. Los documentos son soporte de documentación, permiten la transferencia del conocimiento, registran información histórica, y en muchas cuestiones legales o normativas son obligatorios, pero se resalta que son menos importantes que los productos que funcionan. Menos trascendentales para aportar valor al producto.

Los documentos no pueden sustituir, ni pueden ofrecer la riqueza y generación de valor que se logra con la comunicación directa entre las personas y a través de la interacción con los prototipos. Por eso, siempre que sea posible debe preferirse reducir al mínimo indispensable el uso de documentación, que genera trabajo que no aporta un valor directo al producto.
Si la organización y los equipos se comunican a través de documentos, además de perder la riqueza que da la interacción con el producto, se acaba derivando a emplear a los documentos como barricadas entre departamentos o entre personas.

Valoramos más la colaboración con el cliente que la negociación contractual.

Las prácticas ágiles están especialmente indicadas para productos difíciles de definir con detalle al principio del proyecto, o que si se definieran así tendrían al final menos valor que si se van enriqueciendo con retroinformación continua durante el desarrollo. También para los casos en los que los requisitos van a ser muy inestables por la velocidad del entorno de negocio del cliente.

Para el desarrollo ágil el valor del resultado no es consecuencia de haber controlado una ejecución conforme a procesos, sino de haber sido implementado directamente sobre el producto.

Un contrato no aporta valor al producto. Es una formalidad que establece líneas divisorias de responsabilidades, que fija los referentes para posibles disputas contractuales entre cliente y proveedor.
En el desarrollo ágil el cliente es un miembro más del equipo, que se integra y colabora en el grupo de trabajo. Los modelos de contrato por obra no encajan.

Valoramos más la respuesta al cambio que el seguimiento de un plan

Para un modelo de desarrollo que surge de entornos inestables, que tienen como factor inherente el cambio y la evolución rápida y continua, resulta mucho más valiosa la capacidad de respuesta que la de seguimiento y aseguramiento de planes pre establecidos. Los principales valores de la gestión ágil son la anticipación y la adaptación; diferentes a los de la gestión de proyectos ortodoxa: planificación y control para evitar desviaciones sobre el plan.

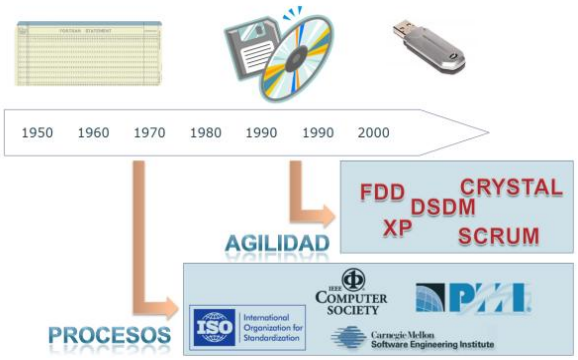


Ilustración 7 - Modelos referentes de procesos y agilidad

Conocimiento en evolución continua

Cuestionar lo conocido es el motor de la evolución del conocimiento.

No es nuevo. Ya lo formuló Platón, es la base de la teoría dialéctica , y como recuerdan Nonaka y Takeuchi, (Nonaka & Takeuchi, 2004) este patrón dialéctico de tesis, antítesis y síntesis dirige la evolución del conocimiento: antítesis que cuestionan las tesis anteriores, y producen nuevas posturas de síntesis que a su vez harán el papel de tesis en el siguiente ciclo evolutivo; formando así una espiral de evolución y perfeccionamiento continuo.

La evolución del conocimiento sigue el patrón dialéctico de tesis – antítesis y síntesis.

Los modelos basados en procesos han sido la "tesis" que inicia el conocimiento para desarrollar sistemas de software.
La agilidad es su antítesis, y estamos generando en estos años la síntesis: el resultado que se enriquece de ambos, y logra un conocimiento más completo y depurado

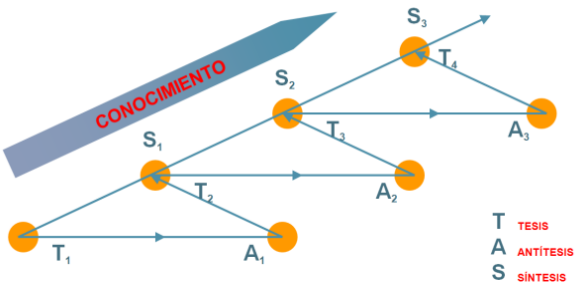


Ilustración 8 - Espiral de conocimiento

En los 80 y 90 comienza a cristalizar la primera base de conocimiento: la tesis.

- Los modelos de procesos específicos: ISO9000-3 CMM's, SPICE-ISO 15504 , Bootstrap, etc.
- Aplicación del mismo patrón predictivo de gestión de proyectos, aplicado en otras ingenierías: PMI , IPMA .
- Primer borrador de consenso sobre el cuerpo de conocimiento de la Ingeniería del Software: SWEBOK (Software Engineering Coordinating Comittee, 2000)

En los 90, llega la difusión y aplicación de este conocimiento. En algunos ámbitos da buenos resultados, y en otros genera la réplica "dialéctica": El Manifiesto Ágil, que cuestiona la validez de los modelos basados en procesos y la gestión predictiva para el desarrollo de software.
Se radicalizan las posturas entre ambas líneas y se genera (y se está generando) la tensión entre contrarios que mueve la evolución dialéctica del conocimiento.

Algunos ejemplos de esta tensión:

"La diferencia entre un atracador de bancos y un teórico de CMM es que con el atracador se puede negociar"…
"La evaluación en CMM depende más de una buena presentación en papel que da la calidad real del producto de software. Tiene que ver más con el seguimiento a ciegas de una metodología que con el desarrollo y puesta en producción de un sistema en el panorama tecnológico".

(Orr., 2003)

"Si uno pregunta a un ingeniero de software típico si cree que CMM se puede aplicar a los métodos ágiles, responderá o con una mirada de sorpresa o con una carcajada histérica".

(Turner & Jain, 2002)

En los últimos años se apuntan ya las tendencias de la evolución hacia la síntesis:
En estos momentos autoridades de la Ingeniería del Software como Barry Boehm y Richard Turner hablan de balancear la agilidad y la disciplina (Boehm & Turner, 2003)

ISO comprueba y anuncia que los modelos desarrollados funcionan en unos entornos, pero no en otros, y ha creado ya comités para desarrollar versiones más ligeras (Laporte & April, 2005).

Surgen iniciativas de normalización como MoProSoft que buscan puntos intermedios entre ambos extremos.

Muchos profesionales plantean dudas sobre ambos extremos, y prueban mezclas y soluciones híbridas.

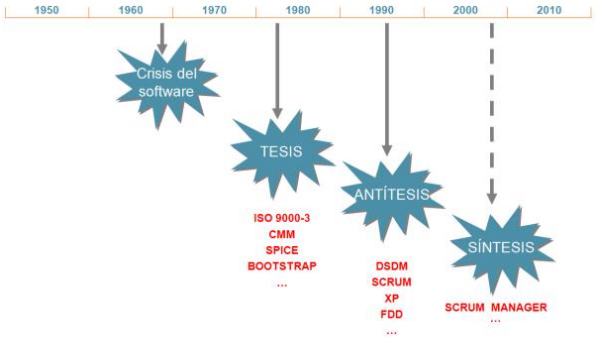


Ilustración 9 - Evolución dialéctica del conocimiento para la gestión de proyectos de software

Estamos determinando la primera oposición en la espiral dialéctica del conocimiento de la Ingeniería del software. Es un momento confuso, en el que ya no está tan claro el norte y resulta difícil orientarse. Era más cómodo en 1995 por ejemplo. Con la tesis desarrollada, y sin haber despertado aún su antítesis ágil, sentíamos que habíamos alcanzado la verdad. Que ya sabíamos cómo desarrollar software. Que era cuestión aplicar pautas de ingeniería en fases secuenciales, con gestión predictiva...

Ahora estamos a mitad de resolución entre esa tesis y su antítesis ágil.

La contradicción produce desconcierto, pero además en nuestro caso, la velocidad de comunicación facilitada por Internet, y el apresuramiento

general del entorno, hace que se solapen las tres tendencias del ciclo.

ISO 15504, CMMI, Scrum, DSDM, Extreme Programming, etc. son grandes aportaciones y es mucho lo que se puede aprender de ellos, pero es iluso pensar de cualquiera de ellas que es La Solución. El conocimiento siempre estará evolucionando, y no tardarán mucho en quedar mejorados. Los libros sobre CMMI o Scrum de hoy, cuando los leamos dentro de pocos años serán textos de conocimiento desfasado.

Scrum Manager no es un modelo basado en procesos, o en agilidad. No es ni tesis ni antítesis. En este momento hace síntesis aprendiendo tanto de los aciertos como de los errores de ambos; y que es posible gracias al conocimiento aportado, y se une a él para avanzar en la espiral del conocimiento que mejora nuestro trabajo.

La empresa como sistema

La realidad sistémica de las organizaciones es un principio de Scrum Manager.

Una empresa es un sistema de múltiples componentes y facetas y las acciones y estrategias en cualquier área tienen implicaciones y necesitan respuestas y alineación con el resto.

La empresa se debe gestionar como una realidad única, y no como un conjunto de departamentos más o menos independientes y comunicados.

Implantar agilidad en la empresa como sistema tiene muchas más implicaciones que implantar agilidad en el departamento de programación.

Desde la visión y cultura de la empresa, hasta las técnicas de programación, pasando por el área de recursos humanos, comercial, contabilidad, calidad, etc. porque implica, o debería implicar, a aspectos como la selección y gestión de las personas, la venta y formalización de contratos, los modelos, prácticas de trabajo, su institucionalización y formación...

Dos aspectos muy importantes antes de comenzar la implantación de un modelo ágil son:

En primer lugar realizar un análisis del perfil de producción y de la identidad de la empresa: cuál es la relevancia de los procesos y de las rutinas, del trabajo y del talento; cuáles las características de los proyectos en cuanto a innovación, estabilidad de requisitos, tamaño...

En segundo, un diseño y una estrategia de implantación, tomando el criterio de flexibilidad de Scrum Management: tomar, modificar o desarrollar las prácticas más adecuadas para usar los principios de la agilidad. En definitiva adaptar las prácticas a la organización, y no al contrario.

Cada empresa tiene su propia estructura organizativa, más o menos compleja y más o menos parecida a otras.
Para Scrum Manager, las áreas de la organización que deben gestionarse de forma sistémica en la implantación o mejora de modelos de producción son las polarizadas en estos tres grupos:

- Gerencia
- Proyecto
- Producto

Gerencia: Áreas de responsabilidad sobre la visión, estrategia, táctica, valores y cultura de la organización.

Proyecto: Áreas de responsabilidad en la comunicación y gestión de los recursos y las personas implicadas en el desarrollo de los proyectos de la empresa.

Producto: Áreas de responsabilidad en la ingeniería y construcción de los productos o servicios desarrollados por la empresa.

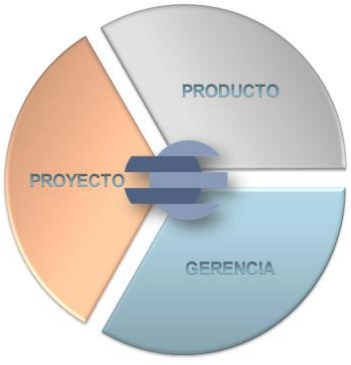


Ilustración 10 - Áreas de conocimiento de Scrum Manager

Reconsiderando: Personas, Procesos y Tecnología

Scrum Manager reconsidera dos vértices del triángulo clásico de los factores de producción: Personas - Procesos y Tecnología.

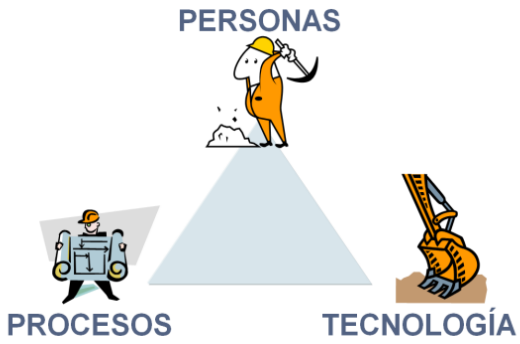


Ilustración 11 - Los tres factores de la producción

Procesos:

No todo lo que etiquetamos como procesos son la misma cosa. En algunos las personas ayudan al "proceso", y en otros son los "procesos" los que ayudan a las personas.
En el primer caso el proceso es el protagonista, el que sabe cómo hacer el trabajo y la persona se integra en el sistema como instrumento, como operario de apoyo. En el segundo el artífice es la persona y el proceso una ayuda, una herramienta que simplifica aspectos rutinarios para que pueda lograr más eficiencia y no diluir el esfuerzo en rutinas mecánicas.

La principal diferencia entre unos y otros es el tipo de conocimiento con el que trabajan. La clasificación de Nonaka y Takeuchi entre explícito (contenido en los procesos y la tecnología), y tácito (contenido en la persona).

Scrum Manager abre una distinción, considerando que los "procedimientos" de trabajo pueden ser: "procesos" cuando tienen el conocimiento clave y por tanto operan en sistemas de conocimiento explícito, y "rutinas" ayudan a las personas, que son quienes tienen el conocimiento clave, y por tanto operan en sistemas de conocimiento tácito.

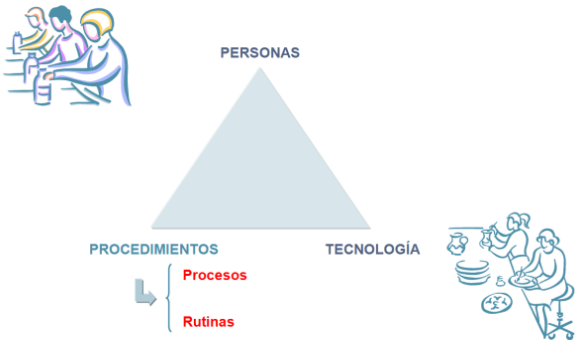


Ilustración 12 - Scrum Manager: Procesos = Procedimientos

Los modelos de gestión y mejora basados en procesos, centran el foco de la producción en los procesos y la tecnología, como los elementos más valiosos de la producción.

Su antítesis, las prácticas ágiles, focalizan el valor de la producción en las personas.

Desde el punto de vista de Scrum Manager, ambas opciones son posibles, pero cada una para un entorno diferente.
En entornos de producción industrial que centran el conocimiento en los procesos y tecnología empleada, las personas aportan trabajo para auxiliar a los procesos y la tecnología, y el valor del resultado depende principalmente de los primeros.

Sin embargo para las empresas del conocimiento que trabajan en escenarios rápidos e innovadores, el principal valor es el talento de las personas.

Personas

En este vértice Scrum Manager apunta que "personas" no es un factor único. Son dos: trabajo y conocimiento; y que el grado de valor de cada uno en el producto es determinante en decisiones de gestión y modelos de producción.

Ilustración 13 - El factor personas en Scrum Manager

Mapa desde el escenario visto desde Scrum Manager

Desde los 80 son muchos los modelos y prácticas que desde organizaciones de investigación, estandarización, y la propia industria se han desarrollado como propuestas de solución a los problemas habituales de los proyectos de software.

Son tantos que su mera relación se antoja como una sopa de letras en la que resulta difícil identificar qué es cada uno: PMBOK, CMM, DSDM, Crystal, ISO 15504, RUP, XP, Scrum, ITIL, ASD, PRINCE 2, LEAN, TDD, etc.

Resulta que los "árboles" no nos dejan ver que el "bosque" tiene una estructura, o un "mapa" relativamente simple que nos permite comprender qué es cada uno, y hace más fácil tomar decisiones para apuntar al conocimiento de unos u otros según queramos trabajar más sobre procesos o rutinas; sobre predicción o agilidad.

Los criterios que dibujan el mapa son:

- Hay Modelos y Prácticas. Los primeros dicen QUÉ hay que hacer, y los segundos cómo se deben hacer.
- Unos trabajan con conocimiento explícito (PROCESOS) y otros con conocimiento tácito (RUTINAS).

y en ambos casos los hay:

- Enfocados en una de las áreas clave identificadas en Scrum Manager (gestión de proyecto, desarrollo de producto o gestión de la organización)
- De ámbito global que cubren las tres áreas clave de la organización.

CRITERIO: MODELOS / PRÁCTICAS

Los modelos no definen "CÓMO" hacer las cosas, sino "QUÉ" cosas se deben hacer para conseguir los mejores resultados. Relacionan, como por ejemplo CMMI, que se debe llevar a cabo: gestión de la configuración, gestión de proyecto, medición y análisis, desarrollo de requisitos, validación, etc. pero sin prescribir formas, modelos ni herramientas concretas.

Unos se centran sólo en un área de la organización (generalmente gestión de proyecto).

Modelos centrados en un área (gestión de proyecto)

- Ágiles: Crystal, DSDM, Lean, modelos ágiles de Unified Process (Open UP, OUM, AUP), MSF for Agile
- Predictivos: PMBOK, PRINCE2

Y otros abarcan todas las áreas de la organización implicadas en el desarrollo de software

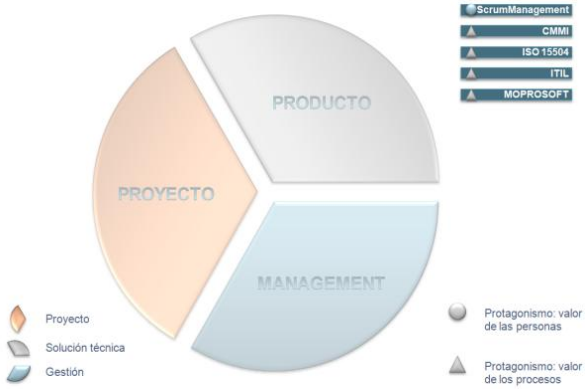


Ilustración 14 - Modelos basados en procesos o personas

Los modelos más conocidos de este tipo son CMMI e ISO 15504.

Tienen dos finalidades:
1.- Para mejorar: Como guión en los procesos de mejora, que indica cuáles son las áreas que se deben ir abordando.
2.- Para evaluar: Como criterio para medir a una organización cómo de bien lo hace, según cuántas de las cosas que dice el modelo, está cubriendo adecuadamente.

Relación de modelos con carácter glogal:

- Basados en procesos: CMMI, ISO 15504, OPM3, P3M3
- Ágiles: Scrum Manager (La necesidad de abordar la agilidad desde la visión de la empresa en su conjunto es una de las razones de evolución hacia Scrum Manager)

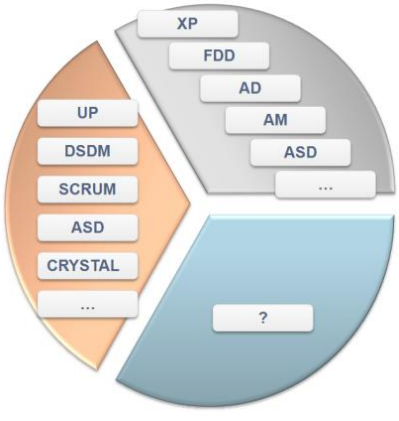


Ilustración 15 - Prácticas ágiles y áreas de la empresa

Las prácticas dicen "CÓMO" hacer las cosas: cómo describir y gestionar los requisitos (historias de usuario, elementos de backlog...), cómo estimarlos, cómo realizar las reuniones para validar con el cliente, cómo realizar el mantenimiento del código, las pruebas, la integración...

- Ágiles: eXtreme Programming, Scrum, FDD, CDT, EVO, TDD
- Predictivas: ITIL

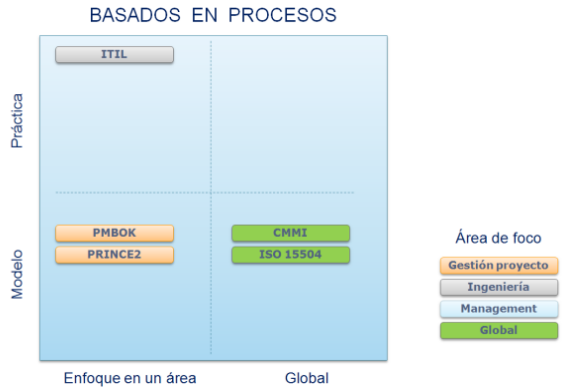


Ilustración 16 - Modelos y prácticas basados en procesos

CRITERIO: PROCESO O RUTINA

Como se ha expuesto en la lección anterior, Scrum Manager diferencia en los procedimientos de trabajo entre procesos y rutinas.
Cuando el procedimiento de trabajo contiene la mayor parte del conocimiento necesario para obtener el resultado (conocimiento explícito), se trata de un proceso.

Cuando son las personas las poseedoras del conocimiento clave para obtener el resultado (conocimiento tácito) entonces los procedimientos de trabajo son rutinas.

Ilustración 17 - Modelos y prácticas basados en rutinas

Scrum Manager: agilidad flexible y sistémica

Scrum Manager es un modelo general de gestión de entornos de producción basados en rutinas, esto es, entornos en los que es más relevante el conocimiento tácito de las personas, que el explícito contenido en los procesos y la tecnología.

Scrum Manager se clasifica mejor como modelo que como conjunto de prácticas, porque a diferencia de éstas, no prescribe "CÓMO" deben hacerse las cosas, (backlogs, historias de usuario, reuniones...) ni que éstas deban hacerse de una determinada manera; sino "QUÉ" cosas deben hacerse. Se centra más en los principios de la agilidad y los procesos, que en prácticas concretas, y sobre los principios, y las características de la empresa y el proyecto, determina los procedimientos de trabajo.

Esta es su característica de flexibilidad: adaptar los procedimientos a la empresa, y no al contrario.

Y se trata además un modelo sistémico o global, porque ***la implantación de Scrum Manager implica no sólo al área de gestión de proyectos, o la de solución técnica; sino a las dos, junto con el "management" o gerencia de la empresa: su organización, estrategia y cultura.***

Ilustración 18 - Características del marco Scrum Mánager

Concepto clásico de gestión de proyectos

Introducción: Proyectos y operaciones

Ilustración 19 - Operaciones y proyectos

El desarrollo de productos, la prestación de servicios, o incluso la organización de la propia empresa son trabajos que pueden tomar la forma de proyectos o de operaciones.

Ambos compantes tres características:

- Los realizan personas.
- Se emplean recursos limitados.
- Se llevan a cabo siguiendo una estrategia de actuación.

Aunque comparten estas tres características, la diferencia clave entre operaciones y proyectos es que:

Las operaciones se ejecutan de forma repetitiva para obtener resultados de similares características
Los proyectos producen resultados únicos

Se considera proyecto a la ejecución de un trabajo que además de requerir personas, recursos y ejecución controlada:

Es un desarrollo único

La teoría clásica de gestión de proyectos, añade a la característica anterior otra, que desde la perspectiva de gestión predictiva tiene sentido, pero no tanto, como se verá, desde la perspectiva de gestión de proyectos ágil.

Se desarrolla en un marco temporal pre-establecido.

Definición clásica de proyecto

Conjunto único de actividades necesarias para producir un resultado definido en un rango de fechas determinado y con una asignación específica de recursos

Las construcciones de ingeniería civil, como puentes o edificios, son ejemplos clásicos de obras realizadas como proyectos, y en general lo es el desarrollo de cualquier sistema singular.

Un proyecto tiene objetivos y características únicas.
Algunos necesitan el trabajo de una sola persona, y otros el de cientos de ellas; pueden durar unos días o varios años.

Algunos ejemplos de proyectos:

- Diseño de un nuevo ordenador portátil.
- Construcción de un edificio.
- Desarrollo de un sistema de software.
- Implantación de una nueva línea de producto en una empresa.
- Diseño de una campaña de marketing.

Origen de la gestión de proyectos

Los proyectos han existido siempre.
Cualquier trabajo para desarrollar algo único es un proyecto, pero la gestión de proyectos es una disciplina relativamente reciente que comenzó a forjarse en los años sesenta.

La necesidad de su profesionalización surgió en el ámbito militar.
En los 50, el desarrollo de complejos sistemas militares, requería coordinar el trabajo conjunto de equipos y disciplinas diferentes, en la construcción de sistemas únicos.
Bernard Schriever, arquitecto del desarrollo de misiles balísticos Polaris, es considerado el padre de la gestión de proyectos, por la introducción del concepto de “concurrencia”, para integrar todos los elementos del plan del proyecto en un solo programa y presupuesto.

El objetivo de la concurrencia era ejecutar las diferentes actividades de forma simultánea, y no secuencialmente, y al aplicarla en los proyectos Thor, Atlas y Minuteman se redujeron considerablemente los tiempos de ejecución.

La industria del automóvil siguió los pasos de la militar, aplicando técnicas de gestión de

proyectos para la coordinación del trabajo entre áreas y equipos diferentes.

Comenzaron a surgir técnicas específicas, histogramas, cronogramas, los conceptos de ciclo de vida del proyecto o descomposición en tareas (WBS Work Breakdown Structure).

En 1960, Meter Norden, del laboratorio de investigación de IBM, en su seminario de Ingeniería de Presupuesto y Control presentado ante American Management Association, indicó:

- ***Es posible relacionar los nuevos proyectos con otros pasados y terminados para estimar sus costes***
- ***Se producen regularidades en todos los proyectos***
- ***Es absolutamente necesario descomponer los proyectos en partes de menor dimensión para realizar planificaciones.***

Organizaciones referentes en la gestión de proyectos

La construcción de sistemas complejos que requerían el trabajo sincronizado de varias disciplinas hizo evidente en los 60 la necesidad de nuevos métodos de organización para evitar problemas recurrentes:

- Desbordamiento de agendas.
- Desbordamiento de costes.
- Calidad o utilidad del resultado obtenido.

Ilustración 20 - Criterios de éxito en la gestión de proyectos tradicional

Para dar respuesta a esta necesidad, desde los años 60 han surgido organizaciones que contribuyen al desarrollo del cuerpo de conocimiento de una gestión de proyectos, para ofrecer garantías de previsibilidad y calidad de los resultados.

Este conocimiento se ha configurando como el currículo de una nueva profesión: La gestión de proyectos **predictiva**.

Las organizaciones más relevantes en esta línea son:

- International Project Management Association (IPMA), fundada en 1965
- Project Management Institute (PMI) constituído en 1969
- Más tarde surgió PRINCE2, que comenzó a trabajar en 1989.

IPMA y PMI surgieron como organizaciones profesionales para desarrollar metodologías y procesos para la gestión de proyectos.
OGC ha tenido la evolución inversa. Comenzó siendo un método (precursor de PRINCE), alrededor del cual se ha terminado creando una organización. Se desarrolla en 1975, pero no es hasta 1989 que pasa a llamarse PRINCE y es en 1996 cuando toma la denominación PRINCE2 y la orientación a todo tipo de proyectos.

Modelo válido para cualquier industria

También en este sentido la evolución ha sido diferente para PRINCE2:
PMI e IPMA tuvieron desde el principio como finalidad el desarrollo de un conocimiento de gestión válido para cualquier proyecto.
Sin embargo, PRINCE2 comenzó siendo un modelo de referencia para proyectos específicos de Tecnologías de la Información, desarrollado por la Central Computer and Telecommunications Agency (CCTA) del Gobierno Británico; y a partir de una revisión llevada a cabo en 1996 se decidió ampliar su ámbito de validez, para cualquier tipo de proyecto.

Planificación y seguimiento

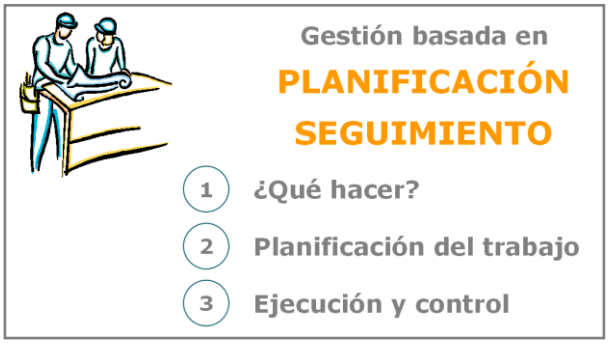


Ilustración 21 - Focos de actuación de la gestión de proyectos tradicional

La gestión de proyectos desarrollada en las últimas décadas del siglo pasado **se basa en la planificación del trabajo, y en el posterior seguimiento y control de la ejecución**.

La planificación se realiza sobre un análisis detallado del trabajo que se quiere realizar y su descomposición en tareas.
Parte por tanto de un proyecto de obra, o de unos requisitos detallados de lo que se quiere hacer.
Sobre esa información se desarrolla un plan adecuado a los recursos y tiempos disponibles; y durante la construcción se sigue de cerca la ejecución para detectar posibles desviaciones y tomar medidas para mantener el plan, o determinar qué cambios va a experimentar.
Se trata por tanto de una gestión "predictiva", que vaticina a través del plan inicial cuáles van a ser la secuencia de operaciones de todo el proyecto, su coste y tiempos.

Su principal objetivo es conseguir que el producto final se obtenga según lo "previsto"; y basa el éxito del proyecto en los tres puntos apuntados: agendas, costes y calidad.

Gestión predictiva o clásica

La gestión de proyectos predictiva o clásica es una disciplina formal de gestión, basada en la planificación, ejecución y seguimiento a través de procesos sistemáticos y repetibles.

- Establece como criterios de éxito: obtener el producto definido, en el tiempo previsto y con el coste estimado.
- Asume que el proyecto se desarrolla en un entorno estable y predecible.
- El objetivo de su esfuerzo es mantener el cronograma, el presupuesto y los recursos.
- Divide el desarrollo en fases a las que considera "ciclo de vida", con una secuencia de tipo: concepto, requisitos, diseño, planificación, desarrollo, cierre.

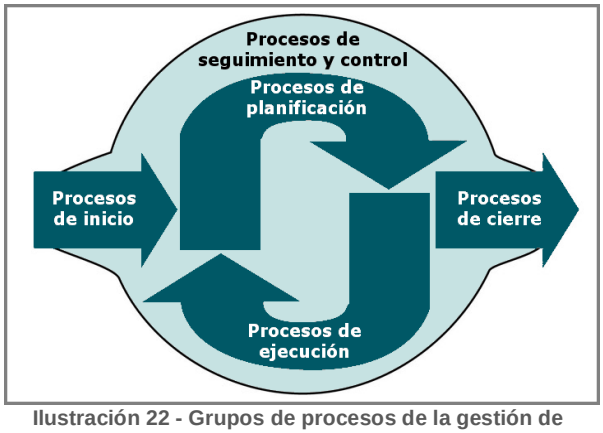


Ilustración 22 - Grupos de procesos de la gestión de proyectos PMBOK 2004

Ámbito de la gestión de proyectos

La solvencia demostrada por la gestión de proyectos en la industria militar, y en la automovilística para solucionar los problemas habituales de calidad, tiempos y costes, coincide en el tiempo con la presión que todas las industrias experimentan en mayor o menor medida para reducir la agenda de salida al mercado y los costes de producción. Como resultado, en todos los sectores: farmacéutico, químico, servicios, tecnologías de la información, etc. se adoptan técnicas de gestión de proyectos, dándoles de facto validez para todos los ámbitos.

Ilustración 23 - Cuerpo de conocimiento aplicable a todos los proyectos

Resumen

- Definición clásica de proyecto: construcción de un resultado único, en unas fechas previstas y con unos recursos previstos de antemano.
- La profesionalización de la gestión de proyectos surgió en los 50 para dar respuesta a las necesidades de la industria militar, y en los años posteriores el resto de industrias han adoptado sus principios.
- Las organizaciones más conocidas por la investigación, y creación de comunidades profesionales para la gestión de proyectos son: PMI (Project Management Institute), Internacional Project Management Association (IPMA) y OGC.
- Características de la gestión de proyectos desarrollada en la segunda mitad del siglo pasado:
  - Gestión basada en la aplicación sistemática de procesos repetibles y escalables.
  - Los criterios de éxito de un proyecto son: calidad, costes y fechas.
  - Carácter predictivo: ejecución según el plan inicial previsto.
  - Desarrollo sobre un entorno estable.
  - El objetivo de la gestión es: desarrollar un plan, y mantener el cronograma y los recursos planificados.
  - Ciclo de vida compuesto por fases secuenciales.

El nuevo escenario

Escenario de desarrollo en los 80

En los 80, el ciclo de vida de los proyectos era el denominado en cascada: el proyecto se divide en fases, y éstas se ejecutan de forma secuencial: definición del producto, diseño, construcción de elementos, integración, pruebas…

Dos características de la construcción de nuevos productos en esta década son:

- Ciclo de vida secuencial.
- División y especialización del trabajo.

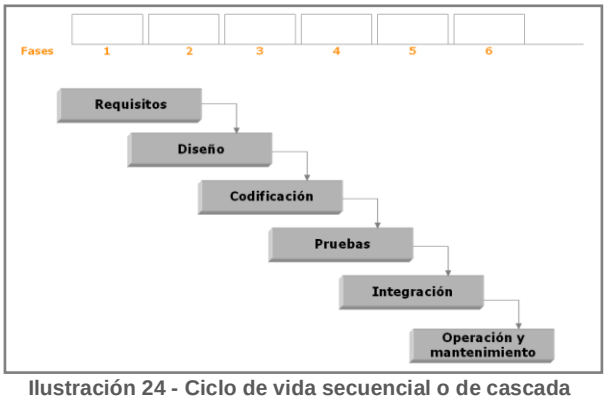


Ilustración 24 - Ciclo de vida secuencial o de cascada

Cada fase la realiza un departamento, personas o equipos diferentes, profesionalmente especializados en los conocimientos necesarios.

La gestión de proyectos desarrolla modelos de estructuras organizativas de tipo matricial, con diferentes variaciones, para facilitar la comunicación y coordinación entre equipos diferentes.

En las mismas fechas, a la par de la consolidación del conocimiento de gestión de proyectos, se estaban desarrollando las teorías de producción basada en procesos, promovidas por Michel Hammer, como mejor medio para garantizar la calidad, eficiencia y repetitividad.

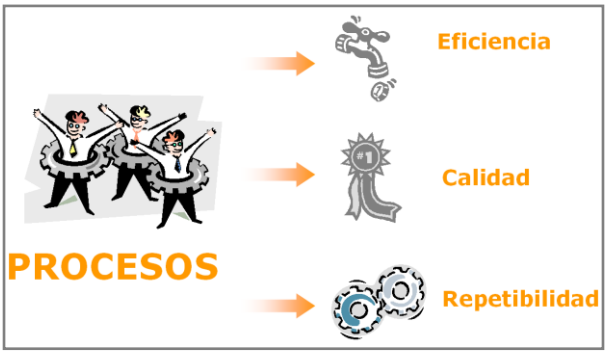


Ilustración 25 – Fortalezas de la producción basada en procesos

División del trabajo, especialización y producción basada en procesos, fueron premisas que, como axiomáticas, asumió la gestión de proyectos, y por esta razón, los puntos clave de la gestión predictiva o clásica son:

- Estimar cuál va a ser el trabajo necesario, y a continuación gestionar la ejecución para que se cumplan la estimación inicial.
- El trabajo se desarrolla en fases.
- División del trabajo en equipos de especialistas.
- Desarrollo basado en procesos.

Ilustración 26 - Características de la producción basada en procesos

The New New Product Development Game

Es el título del artículo publicado en 1986 por Hirotaka Takeuchi e Ikujijo Nonaka; que a su vez daba continuación a otro anterior de los mismos autores junto con Kenichi Imai: "*Managing the New Product Development Process: How Japanese Companies Learn and Unlearn*" (Imai & Takeuchi, 1985).

La publicación de "The New New Product Development Game" (Takeuchi & Nonaka, 1986) ha marcado el punto de inicio de una nueva forma de gestionar proyectos en entornos rápidos e inestables.

Cuando la teoría de gestión de proyectos estaba alcanzando una cierta madurez, los autores observaron que algunas empresas, en mercados muy competitivos, relacionados con productos de vanguardia tecnológica, trabajaban ignorando esa teoría.

"*Muchas compañías han descubierto que para mantenerse en el actual mercado competitivo necesitan algo más que los conceptos básicos de calidad elevada, costes reducidos y diferenciación. Además de esto, también es necesario velocidad y flexibilidad.*"
"*En 1981 las encuestas realizadas a 700 empresas americanas revelan que el 30% de sus beneficios se debe a nuevos productos*".

Hasta entonces, el desarrollo de nuevos productos se realizaba como una carrera de relevos, en la que un grupo de especialistas funcionales pasaban el relevo al siguiente.
El proyecto avanzaba secuencialmente de fase en fase: creación del concepto, pruebas de viabilidad, diseño del producto, diseño del proceso, desarrollo de prototipo y producción final.

Es un modelo de trabajo segmentado por especialización de funciones.
La gente de marketing explora y estudia las necesidades de los clientes, para crear el concepto del producto. Los ingenieros de investigación y desarrollo elaboran un diseño adecuado, los ingenieros de producción llevan a cabo la solución técnica…

La figura siguiente representa el ciclo de vida al construir un producto con un patrón de gestión secuencial, y cuál es la diferencia con la nueva forma, observada por Nonaka y Takeuchi en empresas que ignoraban los principios de la gestión clásica de proyectos.

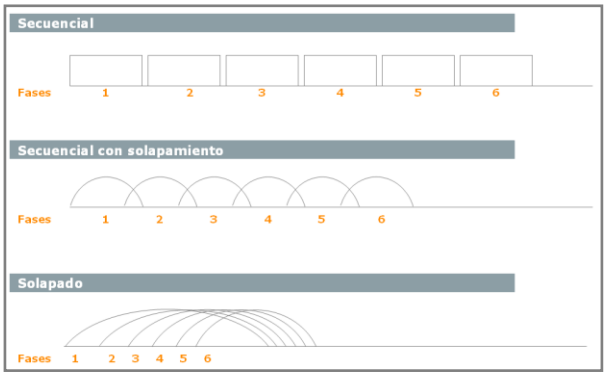


Ilustración 27 - Producción con fases secuenciales o solapadas

Los desarrollos secuenciales puros suelen ser más teóricos que prácticos, y en realidad quienes los adoptan, generalmente producen ciclos "secuenciales con solapamiento", donde una fase no suele necesitar para empezar que esté completamente terminada la anterior.

Nonaka y Takeuchi observaron que empresas americanas y japonesas tecnológicas, de primera línea, que aventajaban a sus competidores en innovación y rapidez, compartían pautas de trabajo comunes, ajenas al clásico patrón secuencial.

Analizaron la forma de trabajo de: Fuji-Xerox, Canon, Honda, Nec, Epson, Brother, 3M, Xerox y Hewlett-Packard y en concreto la forma en la que abordaban el desarrollo de 6 productos:

- La fotocopiadora Fuji-Xerox FX-3500. (1978)
- La copiadora personal Canon PC-10 (1982)
- El coche urbano de 1200cc de Honda (1981)
- El ordenador personal NEC PC 8000 (1979)
- La cámara Canon AE-1 (1976)
- Cámara Canon Auto Boy (1979).

En estas empresas el trabajo no recorría fases a través de diferentes equipos especializados.

"*El producto emerge de la interacción constante de un equipo de élite, multidisciplinario que trabaja conjuntamente desde el principio hasta el final*"

Nonaka y Takeuchi compararon la forma de trabajar de estos equipos únicos y multidisciplinarios, con los equipos de rugby, y el ambiente y entorno de trabajo que les proporcionaba la empresa lo llamaron "**campo de scrum**[1]".

Características del nuevo escenario

Ilustración 28 - Características del nuevo escenario

En los 40, 50 y 60 los productos tardaban años en quedar obsoletos, y las empresas los producían con variaciones mínimas a lo largo del tiempo.

[1] Scrum es un término empleado en rugby para definir una determinada formación del equipo.

Apple ha desarrollado 6 versiones de su popular iPod, en sólo 6 años.

Hoy determinados productos permanecen en un continuo estado "beta". El entorno tecnológico es tan inestable, que las novedades se lanzan tras el menor tiempo de desarrollo posible, dejando que vayan evolucionando a través de versiones, en el propio mercado. Que sea éste quien diga de forma continua cómo deben modificarse los "requisitos".

En estas circunstancias, las diferencias de liderazgo entre unas empresas y otras no radica tanto en la eficiencia y previsibilidad con la que se gestionan el lanzamiento de nuevos productos, sino en la capacidad de agilidad y cambio durante su construcción; y el principal valor para ocupar puestos de cabeza es la innovación.

Campos de scrum vs. modelo clásico de desarrollo

Estos son los principales contrastes que diferencian el desarrollo tradicional del ágil:

Ilustración 29 - Desarrollo tradicional vs. desarrollo ágil

No lo realizan equipos diferentes especializados. Es un equipo único, formado por personas muy competentes, con perfiles y conocimientos que cubren las disciplinas necesarias para llevar a cabo el trabajo.

No hay fases. En realidad **las fases pasan a ser tareas que se ejecutan cuando se necesitan**. No se hace primero el diseño del concepto o los requisitos, más tarde el análisis, luego el desarrollo, etc.
Lo que aplicado al software serían las fases de: requisitos del sistema, requisitos del software, análisis, diseño, construcción, pruebas e integración; y se ejecutarían de forma secuencial, pasan a tareas que se llevan a cabo en el momento que hacen falta. Normalmente a lo largo de pequeñas iteraciones durante todo el desarrollo.

No se espera a disponer de requisitos detallados para comenzar el análisis, ni a tener éste para pasar a la codificación. Muchas veces los requisitos no se pueden conocer si no avanza el desarrollo y se va viendo y "tocando" el resultado.
Otras veces el mercado es tan rápido que a mitad de trabajo las tendencias o la competencia obligarán a modificar el producto.
Además, la participación de todo el equipo en el diseño aporta gran cantidad de talento innovador; un valor clave en el mercado de productos y servicios TIC.

Los equipos ágiles empiezan a trabajar sin conocer con detalle cómo será el producto final. Parten de la visión general, y sobre ella, producen regularmente incrementos de funcionalidad que incrementan el valor al producto.

Fases de desarrollo solapadas

El concepto de "fase" que implica un trabajo secuencial, se cambia ahora por el de "actividad". Requisitos, análisis, diseño, desarrollo no son fases ejecutadas en un orden determinado. Son actividades que se pueden realizar en cualquier momento, de forma simultánea; "a demanda" cuando las necesita el equipo.

En el ciclo de vida secuencial de software se habla de "modificación de requisitos".
Este término lleva implícito el concepto de que estamos "cambiando" algo que quedó cerrado en la *fase* de requisitos.
En el desarrollo ágil, los requisitos evolucionan, se desarrollan y enriquecen durante todo el ciclo de vida, igual que el diseño y el código.

Takeuchi y Nonaka observaron dos tipos de solapamiento: uno que denominaron "sashimi[2]" estableciendo analogía con el plato típico japonés porque se produce un solapamiento bastante amplio, de tal forma, que en cualquier punto del ciclo de vida, se encuentran de forma simultánea varias fases; y otro que denominaron "rugby[3]", que deja perdido por completo el concepto de fases, y en el que el equipo trabaja concurrentemente en todas las actividades desde el primer día.

[2] Nombre que dieron al tipo de solapamiento que empleaba el equipo de desarrollo de la FX-3500 en Fuji-Xerox.
[3] Con este nombre denominaron a la combinación simultánea de todas las fases desde el primer día, empleada por los equipo de Honda.

En el solapamiento “sashimi” aún se mantiene el concepto de fase, aunque con un solapamiento muy amplio.

Ilustración 30 - Solapamiento de fases

En el solapamiento “rugby” no son ya fases, sino definitivamente tareas.

En el desarrollo tradicional:

- Las transiciones entre fases, acaban funcionando como fronteras. Cada equipo se siente responsable de su parte de trabajo, de lo que debe entregar a la siguiente fase, pero no del resultado final.
- Los documentos de diseño, los requisitos o los prototipos pueden acabar siendo barricadas en la frontera de cada fase, que lejos favorecer la comunicación directa fomentan la fragmentación.
- Los retrasos de cada fase son cuellos de botella del proyecto. El solapamiento diluye el ruido y los problemas entre fases.

Características del “campo de scrum”

Las características “ambientales” en las empresas que desarrollan los nuevos productos con modelos de gestión ágil son:

- Incertidumbre consustancial al entorno y la cultura de la organización.
- Equipos auto-organizados.
- Control sutil.
- Difusión y transferencia del conocimiento.

Incertidumbre

Se trabaja en entornos de incertidumbre consustancial.
En estas empresas, la dirección apunta cuál es la meta genérica a la que se pretende arribar, o la dirección estratégica que hay que seguir. No se proporciona el plan detallado del producto.
Al mismo tiempo se dá al equipo un margen de amplia libertad.
Los ingredientes que sirven de acicate para la creatividad y el compromiso son:

- La “tensión” que crea la visión difusa y el reto que supone el grado de dificultad que encierra.
- El margen de autonomía, libertad y responsabilidad.

Auto-organización

Son equipos auto-organizados, sin roles de gestión ni pautas de asignación de tareas.
No se trata de equipos auto-dirigidos, sino auto-organizados. La gestión es la que marca la dirección, pero no la organización.

Parten de cero. Deben empezar por crear su propia organización y buscar el conocimiento que necesitan.
Son similares a una “Start-up” que comienza.

Para lograr la auto-organización los equipos deben reunir tres características:

- Autonomía. Son libres para elegir la estrategia de la solución. En este sentido la dirección de la empresa actúa como un capitalista de capital-riesgo.
- Auto-superación. El equipo va desarrollando soluciones, que evalúa, analiza y mejora.
- Auto-enriquecimiento. La multi-disciplinaridad del equipo favorece el enriquecimiento mutuo y la aportación de soluciones valiosas complementarias.

Control sutil

El equipo dispone de autonomía, pero no debe derivar en caos.
La gestión establece puntos de control suficientes para evitar que el escenario de ambigüedad, inestabilidad y tensión del “campo de scrum” evolucione hacia el descontrol.

Pero debe gestionarse sin un control rígido que impediría la creatividad y la espontaneidad.
El término “control sutil” se refiere a la creación de un ecosistema que potencia y desarrolla el “auto-control entre iguales”, como consecuencia de la responsabilidad y del gusto por el trabajo realizado.

Algunas acciones para generar este ecosistema son:

- Selección de las personas adecuadas para el proyecto.
- Análisis de los cambios en la dinámica del grupo para incorporar o retirar a miembros si resulta necesario.
- Creación de un espacio de trabajo abierto.

- Animar a los ingenieros a “mezclarse” con el mundo real de las necesidades de los clientes.
- Sistemas de evaluación y reconocimiento basados en el rendimiento del equipo.
- Gestión de las diferencias de ritmo a través del proceso de desarrollo.
- Tolerancia y previsión con los errores; considerando que son un medio de aprendizaje, y que el miedo al error merma la creatividad y la espontaneidad.
- Implicar a los clientes en el proyecto.

Difusión y transferencia del conocimiento

Tanto a nivel de proyecto como de organización.

Los equipos son multidisciplinarios, y todos los miembros aportan y aprenden:

- del resto del equipo,
- de las investigaciones para mejorar el valor y el componente innovador que espera el cliente,
- de la experiencia del desarrollo.

Las personas que participan en un proyecto, con el tiempo pasan a otros equipos y proyectos de la empresa, de manera que comparten y comunican el conocimiento a lo largo de toda la organización.

Los equipos y las empresas mantienen libre acceso a la información, herramientas y políticas de gestión del conocimiento

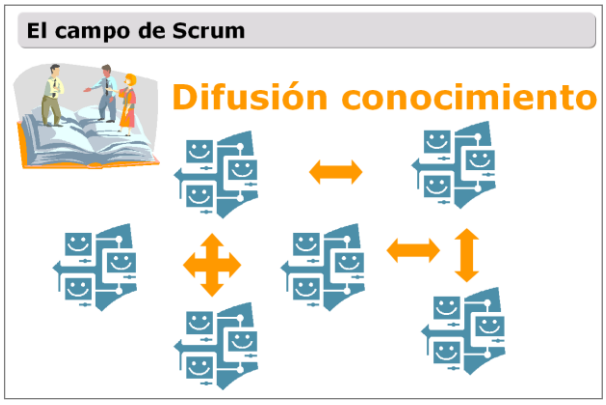


Ilustración 31 - Difusión del conocimiento

Resumen

- Hasta los 80, para el desarrollo de nuevos productos se empleaban:
  - Ciclos de vida secuencial.
  - División y especialización del trabajo.
- En los 80 se desarrolla la teoría de producción basada en procesos para proporcionar eficiencia calidad y repetibilidad.
- En esos años, algunas empresas de tecnología (Caon, Fuji-Xerox, Honda, Epson, HP, etc.) logran más valor y mejores resultados en el desarrollo de nuevos productos, desafiando al desarrollo secuencial y a la división del trabajo.
- Nonaka y Takeuchi son los primeros en identificar estos nuevos entornos de producción a los que denominan “campos de scrum” en el artículo The New New Product Development Game”.
- Las principales diferencias con el desarrollo tradicional de producto son:
  - No trabajan departamentos especializados, sino un único equipo multidisciplinario.
  - Solapamiento de las fases del desarrollo.
  - No se parte de unos requisitos detallados sino de la visión del resultado.
  - No se sigue un plan pre-elaborado.
- Características ambientales en estos entornos llamados “campos de scrum”
  - Incertidumbre
  - Auto-organización
  - Control sutil
  - Difusión del conocimiento
  - Fases de desarrollo solapadas

Gestión de proyectos ágil

Conceptos

Introducción

Muchas empresas trabajan en escenarios que se parecen ya muy poco a los que impulsaron la gestión de proyectos predictiva y necesitan estrategias diferentes para gestionar el lanzamiento de sus productos: estrategias orientadas a la entrega temprana de resultados tangibles, y con la suficiente agilidad y flexibilidad para trabajar en entornos inestables y rápidos.

Ahora necesitan construir el producto al mismo tiempo que cambian y aparecen nuevos requisitos; y como las circunstancias de los mercados y de las empresas no se pueden cambiar, son las formas en las que gestionan sus proyectos las que tienen que cambiar para dar respuesta a las nuevas necesidades.

El cliente conoce la visión de su producto pero por la novedad, el valor de innovación que necesita y la velocidad a la que se va a mover el escenario tecnológico y de negocio, durante el desarrollo, no puede detallar cómo será el producto final.

¡Ah!. Pero, ¿existe el producto final?.

Quizá ya no hay "productos finales", sino productos en evolución, mejora o incremento continuo, desde la primera versión beta.

El resultado es la gestión ágil de proyectos, que no se formula sobre el concepto de anticipación (requisitos, diseño, planificación y seguimiento) sino sobre el de adaptación (visión, exploración y adaptación)

Objetivos de la gestión ágil

La gestión ágil de proyectos tiene como objetivo dar garantías a las demandas principales de la industria actual: valor, reducción del tiempo de desarrollo, agilidad, flexibilidad y fiabilidad.

1.-Valor

La gestión ágil se necesita en los mercados rápidos.
Su objetivo es dar el mayor valor posible al producto, cuando éste se basa en:

- Innovación
- Flexibilidad

La permanencia de estas empresas depende de su capacidad de innovación continua. Del lanzamiento continuo de novedades, que compiten con los productos de otras empresas que también están en continua innovación.

Flexibilidad.
El producto no sólo es valioso por su valor en el momento de su lanzamiento, sino también por su capacidad de adaptación y evolución a través de actualizaciones y ampliaciones.

2.-Reducción del tiempo de salida al mercado

En la década de los 90, el tiempo medio de salida al mercado de los nuevos productos en EE.UU. se redujo de 35,5 a 11 meses (Wujec & Muscat, 2002)

Este tiempo es un factor competitivo clave en determinados sectores.

Las estrategias de la gestión ágil para producir resultados en menos tiempo que la gestión tradicional son:

- Solapamiento de las fases de desarrollo.
- Entrega temprana de las primeras partes del producto, que corresponden con las de mayor urgencia para el cliente, de forma que puede lanzar la primera versión en el menor tiempo posible.

3.-Agilidad

Capacidad para producir partes completas del producto en periodos breves de tiempo

4.-Flexibilidad

Capacidad para adaptar la forma y el curso del desarrollo a las características del proyecto, y a la evolución de los requisitos.

5.- Resultados fiables

El objetivo de la gestión predictiva es ejecutar el trabajo planificado (y conocido de antemano) en el plazo planificado y por el coste previsto.

La gestión ágil no tiene un carácter predictivo o de anticipación. No conoce de antemano el detalle del producto que va a desarrollar, y por eso

su objetivo no es fiabilidad en el cumplimiento de los planes, sino en el valor del resultado.

Los procesos de la gestión tradicional son buenos cuando consiguen desarrollar de forma repetible los productos especificados en el tiempo y con los costes previstos.

Los procesos de la gestión ágil son buenos, cuando consiguen entregar de forma temprana y continua un valor innovador.

Las preferencias de la gestión ágil

La gestión ágil, a diferencia de la tradicional, muestra las preferencias resumidas en el manifiesto ágil:

1.- **La capacidad de respuesta al cambio, sobre el seguimiento de un plan**.

2.- **Los productos que funcionan frente a especificaciones y documentaciones innecesarias**.

3.- **La colaboración con el cliente frente a la negociación contractual**.

4.- **A las personas y su interacción por encima de los procesos y las herramientas.**

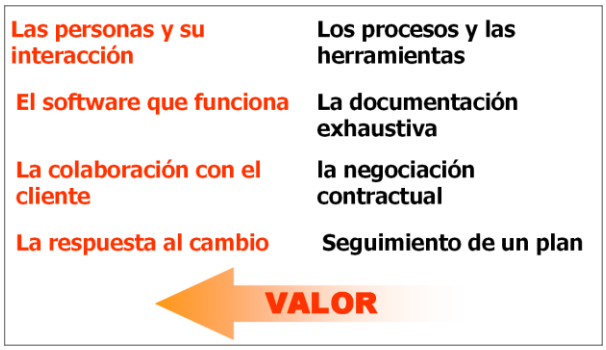


Ilustración 32 - Valores en los que se asienta la agilidad

El ciclo de desarrollo ágil

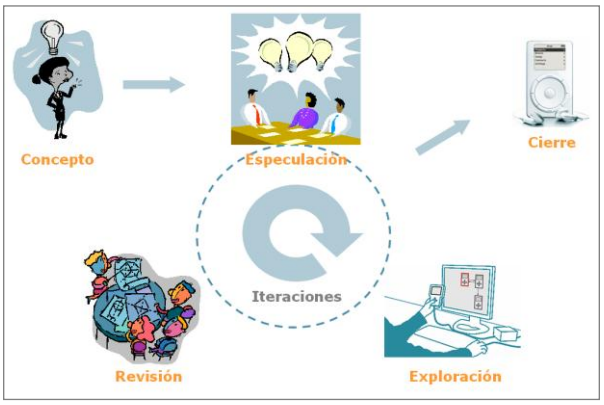


Ilustración 33 - Ciclo de desarrollo ágil

El desarrollo ágil parte de la visión, del concepto general del producto, y sobre ella el equipo produce de forma continua incrementos en la dirección apuntada por la visión; y en el orden de prioridad que necesita el negocio del cliente.

Los ciclos breves de desarrollo, se denominan iteraciones y se realizan hasta que se decide no evolucionar más el producto.

Este esquema está formado por cinco fases:

1.- Concepto
2.- Especulación
3.- Exploración
4.- Revisión
5.- Cierre

1.- Concepto

En esta fase se crea la visión del producto y se determina el equipo que lo llevará a cabo.

Partir sin una visión genera esfuerzo baldío.
La visión es un factor crítico para el éxito del proyecto.
Se necesita tener el concepto de lo que se quiere, y conocer el alcance del proyecto. Es además una información que deben compartir todos los miembros del equipo

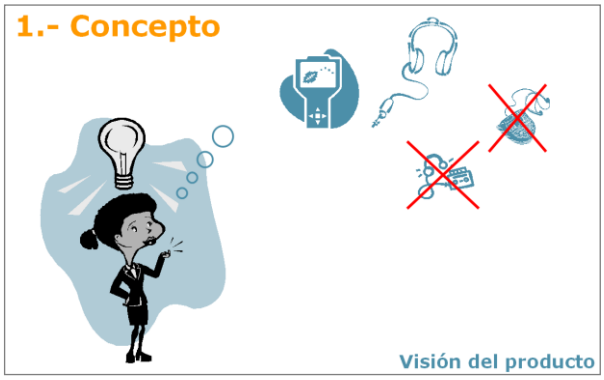


Ilustración 34 - Concepto o visión del cliente

2.- Especulación

Una vez que se sabe qué hay que construir, el equipo especula y formula hipótesis basadas en la información de la visión, que *per se* es muy general e insuficiente para determinar las implicaciones de un desarrollo (requisitos, diseño, costes…).

En esta fase se determinan las limitaciones impuestas por el entorno de negocio: costes y agendas principalmente, y se cierra la primera aproximación de lo que se puede producir.

La gestión ágil investiga y construye a partir de la visión del producto. Durante el desarrollo confronta las partes terminadas: su valor, posibilidades, y la situación del entorno en cada momento.

La fase de especulación se repite en cada iteración, y teniendo como referencia la visión y el alcance del proyecto consiste en:

- Desarrollo y revisión de los requisitos generales.
- Mantenimiento de una lista con las funcionalidades esperadas.
- Mantenimiento de un plan de entrega: fechas en las que se necesitan las versiones, hitos e iteraciones del desarrollo. Este plan refleja ya el esfuerzo que consumirá el proyecto durante el tiempo.
- En función de las características del modelo de gestión y del proyecto puede incluir también una estrategia o planes para la gestión de riesgos.

Si las exigencias formales de la organización lo requieren, también se produce información administrativa y financiera.

Ilustración 35 - Especulación o aportación de todo el equipo

3.- Exploración

Se desarrolla un incremento del producto, que incluye las funcionalidades determinadas en la fase anterior.

4.- Revisión

Equipo y usuarios revisan lo construido hasta ese momento.
Trabajan y operan con el producto real contrastando su alineación con el objetivo.

Ilustración 36 - Revisión

5.- Cierre

Al llegar a la fecha de entrega de una versión de producto (fijada en la fase de concepto y revisada en las diferentes fases de especulación), se obtiene el producto esperado.

Posiblemente éste seguirá en el mercado, y por emplear gestión ágil, es presumible que se trata de un producto que necesita versiones y mejoras frecuentes para no quedar obsoleto. El cierre no implica el fin del proyecto.

Lo que se denomina “mantenimiento” supondrá la continuidad del proyecto en ciclos incrementales hacia la siguiente versión para ir acercándose a la visión del producto.

Principales modelos de gestión ágil

Si hubiera que determinar cuál es el origen de la gestión ágil de proyectos, a falta de mejor información, habría que situarlo en las prácticas adoptadas en los 80 por empresas como Honda, 3M, Canon, Fuji, Nec, Xerox, hp o Epson para el desarrollo de nuevos productos[4].

La industria del software ha sido la primera en seguir su adopción, y muchos de sus profesionales han documentado y propagado las formas particulares en las que han implementado los principios de la agilidad en sus equipos de trabajo.
De esta forma han aparecido en las últimas décadas los nombres:

- AD - Agile Database Techniques
- AM - Agile Modeling
- ASD - Adaptive Software Development
- AUP - Agile Unified Process
- Crystal
- FDD - Feature Driven Development
- DSDM - Dynamic Systems Development Method
- Lean Software Development
- Scrum
- TDD - Test-Driven Design
- XBreed
- XP - eXtreme Programming

Éstos son los modelos que se encuentran inscritos en la organización Agile Alliance (www.agilealliance.org) para promocionar y difundir su conocimiento.

Cada una de ellos expone formas concretas de aplicación de principios ágiles en el desarrollo de software.
Algunos determinan cómo realizar las pruebas, o la duración que emplean para desarrollar cada iteración, o el protocolo para realizar las reuniones de trabajo.

Unos métodos cubren áreas concretas de la ingeniería del software (diseño, desarrollo pruebas), como es caso de AD, AM o XP, y otros se centran en la gestión del proyecto.

Éstos últimos son:

- ASD - Adaptive Software Development
- AUP - Agile Unified Process
- Crystal
- DSDM - Dynamic Systems Development Method
- Scrum
- XBreed

Por ejemplo, el principio de desarrollo ágil iterativo e incremental, tiene reflejo en ciclos de 30 días empleados por scrum, o de entre 1 y 4 meses empleado por los modelos Crystal.

ASD

Adaptive Software Development es el modelo de implementación de patrones ágiles para desarrollo de software, diseñado por Jim Highsmith, (Highsmith, 2000) que materializa las fases de la gestión ágil de la siguiente forma:

ESPECULACIÓN, compuesta por 5 pasos:
1.- Inicio para determinar la misión del proyecto.
2.- Fijación del marco temporal del proyecto.
3.- Determinación del nº de iteraciones y la duración de cada una.
4.- Definición del objetivo de cada iteración.
5.- Asignación de funcionalidad a cada iteración.

COLABORACIÓN
Desarrollo concurrente del trabajo de construcción y gestión del producto.

APRENDIZAJE
En cada iteración se revisa:
- Calidad, con criterios de cliente.
- Calidad, con criterios técnicos.
- Funcionalidad desarrollada.
- Estado del proyecto.

Las características básicas de ASD son:
- Trabajo orientado y guiado por la misión del proyecto.
- Basado en la funcionalidad.
- Desarrollo iterativo.
- Desarrollo acotado temporalmente.
- Guiado por los riesgos.
- Trabajo tolerante al cambio.

AUP

Agile Unified Process es una versión simplificada de Rational Unified Process, desarrollada por Scott Amber (Ambler).

Divide el ciclo de desarrollo en 4 fases:

INICIO: identificación del alcance y dimensión del proyecto, propuesta de la arquitectura y del presupuesto del cliente.

[4] Hirotaka Takeuchi e Ikujiro Nonaka, 1986 "The New New Development Game"

ELABORACIÓN: Confirmación de la idoneidad de la arquitectura.
CONSTRUCCIÓN: Desarrollo incremental del sistema, siguiendo las prioridades funcionales de los implicados.
TRANSICIÓN: Validación e implantación del sistema.

CRYSTAL

Concebido por Alistair Cockburn (Cockburn, 2004), este modelo no describe una metodología cerrada, sino un conjunto de ellas, junto con los criterios para seleccionar y adecuar la más apropiada al proyecto. Los parámetros para determinarla son la criticidad y el tamaño del sistema que se va a construir.

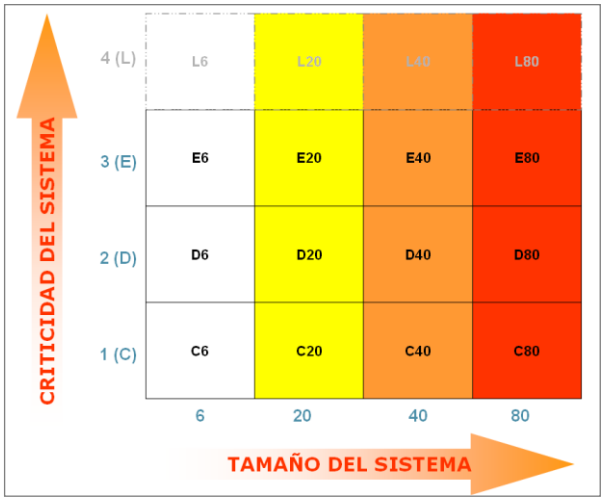


Ilustración 37 - Esquema del modelo CRYSTAL

Los criterios empleados para la medición de estos parámetros son:

Criticidad (dimensión de las pérdidas que ocasionaría un malfuncionamiento del sistema)

1 (c): Pérdida de confort o usabilidad.
2 (d): Pérdidas económicas moderadas.
3 (e): Pérdidas económicas graves.
4 (l): Pérdida de vidas humanas.

Estos criterios corresponden a los niveles de integridad de un sistema definidos por el estándar IEEE 1012-1998.

Dimensión.
Crystal determina el tamaño del sistema por el nº de personas empleadas en su desarrollo. (6 - 20 - 40 - 80)

Fundamentos de Crystal:

- Desarrollo iterativo e incremental.
- Duración máxima de una iteración: 4 meses. Recomienda duraciones entre 1 y 3 meses.
- Especial énfasis en la importancia de las personas sobre los procesos.
- Especial énfasis en la comunicación directa.
- Modelo abierto a la adaptación e introducción de prácticas de otros modelos ágiles (eXtreme Programming, Scrum...)

DSDM

DSDM es el acrónimo que da nombre a un modelo de procesos para desarrollo de sistemas de software, concebido por el DSDM Consortium, que se fundó en Inglaterra en 1994, y que actualmente tiene presencia en Inglaterra, EE.UU., Benelux, Dinamarca, Francia y Suiza; y con interés y contactos para futuras representaciones en Australia, India y China [...]

Es un modelo que estuvo representado en la firma del Manifiesto Ágil: Arie van Bennekum, firmante del manifiesto, era miembro del consorcio en Benelux, consultor y formador de DSDM.

En 2001, año del Manifiesto Ágil, DSDM publicó la versión 4.1 de su modelo, y se consideró una metodología ágil; y aunque mantuvo las siglas, cambió la denominación original (Dynamis Systems Development Method) por Framework for Business Centred Development.

Procesos del ciclo de desarrollo DSDM

El ciclo de desarrollo de DSDM está compuesto de 5 fases, precedidas de un pre-proyecto y un post-proyecto.

1. Pre-proyecto
2. Estudio de viabilidad
3. Estudio de negocio
4. Iteración de modelado funcional
5. Iteración de diseño y desarrollo
6. Implementación
7. Post-desarrollo

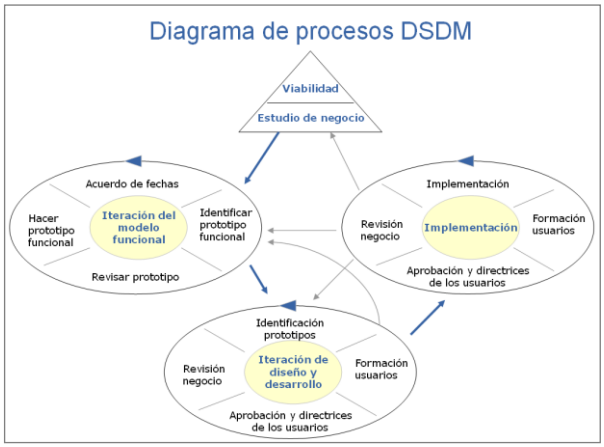


Ilustración 38 - Diagrama del modelo DSDM

SCRUM

En 1995 Ken Schwaber presentó en OOPSLA 95 (Object-Oriented Programming Systems & Applications conference) (Schwaber, 1995), la implementación de Scrum Para software que él empleaba en el desarrollo de Delphi, y Jeff Sutherland en su empresa Easel Corporation (compañía que en los macrojuegos de compras y fusiones se integraría en VMARK, y luego en Informix y finalmente en Ascential Software Corporation).

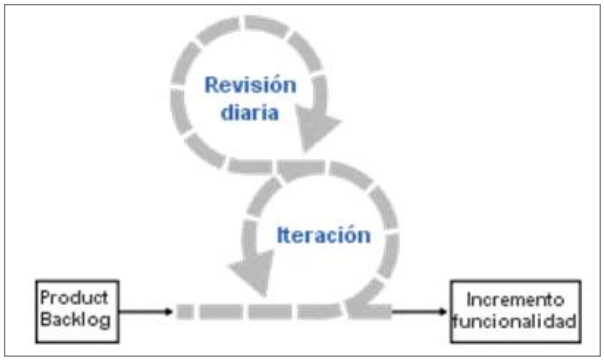


Ilustración 39 - Diagrama de ciclo Scrum

Se basa en el principio ágil de desarrollo iterativo e incremental.
Al período de trabajo para desarrollar un incremento de producto se lo denomina “sprint”, y se recomiendan duraciones entre una y cuatro semanas, si bien pueden contemplarse casos de hasta 60 días.

Establece una reunión al inicio de cada sprint para determinar el trabajo que se va a realizar, otra al final para evaluar el resultado, y revisiones diarias que realiza el equipo en su auto-gestión.

XBreed – Agile Enterprise

Propuesto por Mike Breedle, que colaboró con Ken Schwaber en la definición de Scrum, es una combinación de Scrum para la gestión del proyecto, y Extreme Programming como prácticas de desarrollo.

Esta es una combinación comúnmente empleada independientemente de su definición como Xbreed que hasta la fecha no ha tenido especial relevancia. También denominado “Agile Enterprise”.

Resumen

- La gestión ágil de proyectos no es una gestión de anticipación (requisitos, diseño, planificación y seguimiento) sino de adaptación (visión, exploración y adaptación).
- La gestión ágil tiene como objetivos: valor, reducción del tiempo de desarrollo, agilidad, flexibilidad y fiabilidad.
- La gestión ágil se basa en los principios del manifiesto ágil y centra el valor:
  - Más en las personas y su interacción que en los procesos y las herramientas
  - Más en los resultados que funcionan que en la documentación exhaustiva
  - Más en la colaboración con el cliente que en la negociación contractual
  - Más en la capacidad de respuesta al cambio que en el seguimiento de un plan
- El desarrollo ágil comprende cinco fases: concepto, especulación, exploración, revisión y cierre.
- El desarrollo ágil surgió en empresas de productos tecnológicos, fue identificado por Nonaka y Takeuchi en los años 80 y a partir de los 90 diferentes profesionales del desarrollo del software incorporaron sus principios en sus entornos de trabajo. De esas implementaciones ágiles, las que abordan la gestión del proyecto son: ASD, AUP, Crystal, DSDM, Scrum.

Gestión de proyectos: ¿formal o ágil?

¿Ágil, clásica, predictiva ...?

Al surgir en los 80 una nueva forma de gestionar proyectos, se hizo necesario añadir un "apellido" al término "gestión de proyectos" para matizar si se refiere a la nueva o a la de siempre.

La nueva, al autodenominarse ágil, obligó a dar un apellido al modelo de gestión de proyectos que hasta entonces, por único, no lo había necesitado.

Las denominaciones empleadas para cada tipo son:

Gestión de proyectos	Clásica Tradicional Predictiva Formal ~~Pesada~~

Gestión de proyectos	Ágil Adaptable ~~Adaptativa~~

En algunos ámbitos, hay cierta rivalidad académica o profesional entre defensores de uno y otro modelo. Preferimos por tanto no emplear el término "pesado" que puede aportar connotaciones peyorativas.

También preferimos no emplear "adaptativa" y usar en su lugar "adaptable", para evitar un anglicismo innecesario.

Premisas de la gestión de proyectos predictiva

Premisas sobre las que se desarrolló la gestión de proyectos tradicional:

1.- Todos los proyectos mantienen características y comportamientos regulares (Norden, 1958)

2.- El objetivo de la ejecución de un proyecto es lograr el producto previsto en el tiempo planificado sin desbordar los costes estimados.

Ilustración 40 - Objetivo de la gestión de proyectos predictiva

Características de la gestión de proyectos predictiva

Estas premisas han dado dos características a la gestión de proyectos predictiva:

1.- Universalidad
Los proyectos, pese a su diversidad, comparten patrones comunes de ejecución.
Las prácticas de gestión se basan en estos patrones comunes y resultan válidas para cualquier tipo de proyecto.

Ilustración 41 - ¿La gestión predictiva es válida para todos los proyectos?

2.- Carácter predictivo
La gestión clásica define con detalle cuál es el "producto previsto" y elabora un plan de desarrollo, a partir del cual calcula costes y fechas.
Durante la ejecución realiza actividades de seguimiento y vigilancia para evitar desviaciones sobre lo planificado.

Hay otras premisas

Las dos premisas que cimentan el desarrollo de la gestión de proyectos:

- Todos los proyectos comparten idénticos patrones de ejecución.
- El objetivo es conseguir el producto definido en costes y fechas.

son cuestionables:

1.- No hay una forma única y válida para gestionar cualquier tipo de proyecto

Es cierto que muchas características que diferencian unos proyectos de otros son superficiales y resultan indiferentes para el modelo de gestión; pero hay otras que permiten adoptar estrategias de gestión muy diferentes en cada caso.

Características diferenciales:

- Componente innovador que se espera del resultado.
- Grado de estabilidad de los requisitos durante el desarrollo.
- Coste de prototipado.
- Maleabilidad del producto para modificar su funcionalidad una vez desarrollado.

La gestión de proyectos predictiva, al autoconsiderarse válida para cualquier proyecto no contempla que según las características del proyecto, puedan resultar más apropiados otros criterios de gestión.

- Conseguir el mayor valor innovador del producto no es uno de sus objetivos, y por tanto no aplica prácticas diferentes según el grado innovador que se desee.

- Considera que los requisitos deben permanecer estables durante la ejecución. Si no ocurre esto presupone que se debe a deficiencias en el proceso de la fase de requisitos; porque no contempla posibles evoluciones rápidas o inestabilidades del entorno tecnológico o de negocio.

- El objetivo es la eficiencia, el cumplimiento del plan, y no el valor del producto. Desde este punto de vista, el re-trabajo siempre es caro. No se considera la relación entre el coste del re-trabajo y el valor proporcionado.

2.- Hay proyectos en los que no se quiere "hacer el producto descrito en las fechas y con los costes estimados"

Cuando en 1978 la dirección de Honda encargó el diseño de un nuevo automóvil, sólo dió dos instrucciones al equipo de ingenieros:
"Primero: necesitamos un concepto de automóvil completamente diferente a lo que cualquier otra compañía haya hecho nunca; y segundo: debe ser un coche económico, pero no barato".

El resultado fue el Honda Civic.

Se pedía un proyecto, pero no se quería garantizar la ejecución de un plan, sino obtener el máximo valor en las líneas marcadas.

Hay proyectos en los que importa más el valor y la innovación que el cumplimiento del plan.

La gestión predictiva pide al equipo el cumplimiento del plan.
La gestión adaptable pide al equipo el mayor valor posible para una visión de producto

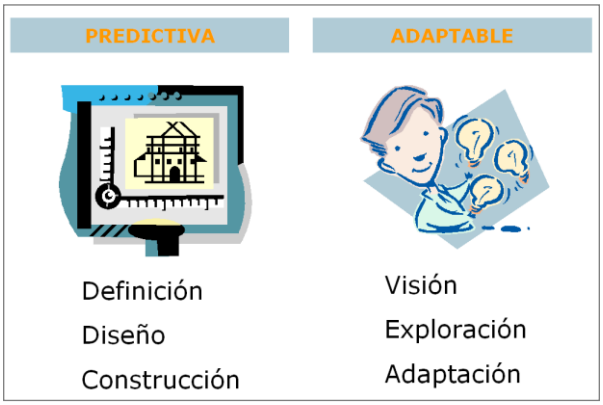


Ilustración 42 - Focos de la gestión predictiva y de la gestión ágil

¿Cuándo y por qué emplear uno u otro estilo de gestión?

Para obtener los mayores beneficios que cada estilo de gestión puede ofrecer, éste tiene que ser compatible no sólo con las características del proyecto, sino también con las de la organización que las va a aplicar.

Características del proyecto

Las características relevantes para decidir el estilo de gestión más adecuado son:

- Principal prioridad de negocio.
- Estabilidad de los requisitos.
- Rigidez del producto.
- Coste de prototipado.
- Criticidad del sistema.
- Tamaño del sistema.

El orden expuesto se corresponde con nuestro criterio de relevancia, siendo por tanto la prioridad de negocio la principal razón de compatibilidad o incompatibilidad, y el tamaño del sistema la menos relevante.

Por la relativa novedad de la gestión ágil, estos criterios no están aún consensuados. Así por ejemplo mientras algunos textos opinan que el tamaño o la criticidad del sistema son aspectos muy relevantes, hay opiniones autorizadas en sentido contrario:

- En la "International Conference on Complex Systems 2006", Jeff Sutherland presentó el informe "Adaptive Engineering of Large Software Projects with Distributed / Outsourced Teams " (Sutherland, Viktorov, & Blount, 2006) que demostraba los buenos resultados obtenidos con prácticas de gestión ágiles en un desarrollo de grandes dimensiones: un millón de líneas de código Java, y un equipo de 50 personas distribuidos en dos empresas ubicadas en países distintos.
- Ken Schwaber, entre otros, contempla el desarrollo de sistemas críticos, incluyendo los requerímientos de conformidad también como resultados entregables de las iteraciones junto con las funcionalidades a las que afectan.

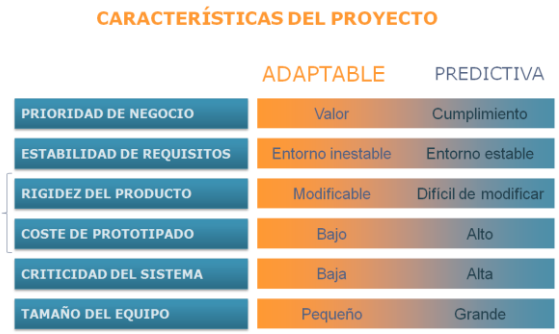


Ilustración 43 - Criterios de idoneidad para gestión ágil o predictiva, dependientes del proyecto

Prioridad de negocio

¿Cuál es la principal prioridad para los intereses de negocio del cliente?
¿Qué tiene más importancia: el cumplimiento de agendas y fechas o el valor innovador del producto?

Este es el primer aspecto que se debe considerar. La gestión predictiva es un modelo construido y especializado en garantizar el cumplimiento de los planes.

La gestión adaptable es un modelo construido y especializado en dar el mayor valor posible al producto.

Por supuesto los dos objetivos son deseables, pero hay que elegir, porque simplemente son excluyentes. No se pueden planificar diagramas de Gantt o rutas críticas sobre una visión general.

Cuanto mayor valor se desea en uno u otro extremo (valor o predicción), más contraproducente resulta emplear el estilo de gestión inadecuado.

Estabilidad de los requisitos

¿Se puede obtener una descripción de requisitos detallada al inicio del proyecto, y ésta se mantendrá estable durante el desarrollo?

O lo que es lo mismo, ¿Se puede saber con certeza y detalle qué es lo que se quiere construir, siendo improbable que cambien los criterios o las necesidades?

Rigidez del producto

¿Cuán fácil resulta modificar el producto?

Esta es una razón importante, porque no es lo mismo modificar software, circuitos electrónicos, construcciones civiles…

Modificar la estructura de una base de datos para añadir algunas tablas no es lo mismo que modificar la estructura de un edificio para rectificar el número de plantas.

Coste de prototipado

Otra cuestión relevante para el modelo de gestión ágil es la relación: coste de prototipar / valor conseguido para el producto. Este factor suele estar relacionado con la rigidez del producto.

Ver, tocar, e interactuar con las partes ya desarrolladas (o con simulaciones o prototipos) genera ideas y posibilidades que sobre el concepto inicial y el papel no llegan a concebirse.

El prototipado y el feed-back que proporciona son extremadamente importantes, sobre todo en el desarrollo de nuevos productos, o de sistemas innovadores.

A medida que el equipo lo va “tocando” y “probando” surgen funcionalidades y posibilidades nuevas que aportan mayor valor al concepto inicial.

En este sentido, el argumento: "la forma más eficiente de desarrollar un trabajo es hacerlo bien a la primera", que se emplea con frecuencia para defender la validez de la gestión predictiva en cualquier proyecto, resulta tendencioso. La afirmación "per se" es obviamente cierta; pero también son ciertas dos circunstancias relacionadas:

- Se puede hacer "bien a la primera" cuando es posible conocer con detalle el resultado sin necesidad de hacer pruebas antes.
- Las posibilidades al hacer un trabajo no son sólo "bien" o "mal". Bien es un término amplio. Puede ser aceptable o suficientemente bien, o lo mejor posible.

Estos factores, junto con la relación entre coste de prototipado y valor que aporta deben tenerse también en cuenta para elegir el modelo de gestión más adecuado para el proyecto.

Ilustración 44 – Criterio de la relación coste / beneficio del prototipado frente a la planificación rígida

Criticidad del sistema

¿Cuál es el grado de criticidad del sistema que va a desarrollar?

Considerando por análisis de criticidad:

La evaluación estructurada de las características del producto (p. ej.: seguridad, complejidad, rendimiento) para determinar la severidad del impacto de un fallo del sistema, de su degradación o de su no cumplimiento con los requisitos o los objetivos del sistema.

O lo que es lo mismo:

Si el sistema falla, se degrada o no consigue realizar las funciones de los requisitos, ¿qué impacto tiene en la seguridad o en el rendimiento?

Un ejemplo de criterios de criticidad, ordenados de mayor a menor, puede ser:

- Causará daño a las personas.
- Causará daño al medio ambiente.
- Producirá pérdidas económicas graves.
- Producirá pérdidas económicas.
- Fallará la finalidad principal del sistema.
- Fallarán funcionalidades auxiliares del sistema.
- Se producirán fallos ergonómicos o de comodidad para los usuarios.

Tamaño del sistema

Una de las principales bases del desarrollo ágil es la preferencia de la comunicación e interacción directa de los implicados en el proyecto.

Los grandes proyectos implican equipos numerosos y en ocasiones físicamente distantes, circunstancias que dificultan la comunicación directa.
No obstante hay desarrollos incipientes de prácticas ágiles que implantan esquemas de agrupamiento y comunicación directa en estructuras celulares de equipos de hasta 6 personas.

Condiciones de la organización

Los elementos empleados por las organizaciones para ejecutar proyectos son: personas, procesos y tecnología.

Los resultados de la gestión ágil dependen más del valor de las personas que de los procesos de la organización.

Las personas tienen características propias:

- Sus resultados son "sensibles" al entorno. La falta de motivación y los ambientes laborales hostiles reducen significativamente el valor intelectual del trabajo.
- Cuando el trabajo depende del talento, la diferencia de valor entre los mediocres y los mejores es muy grande.

Adoptar modelos de desarrollo ágil no consiste sólo en realizar las prácticas formales: equipo único, reuniones periódicas, desarrollo evolutivo de los requisitos, etc.
Si la organización mantiene un modelo de desarrollo basado en procesos y no en personas, y no tiene alineadas con los principios ágiles: la cultura y estructura organizativa, no obtiene los resultados propios del desarrollo ágil.

Ilustración 45 Criterios de idoneidad para gestión ágil o predictiva, dependientes de la organización

Nivel profesional

"*En el mundo del diseño informático, los mejores lo hacen entre 50 y 100 veces mejor que el promedio, y la cifra aumenta, conforme se incrementa la complejidad de la tecnología*"

Pilar Jericó. "La gestión del talento"

"*La diferencia entre los promedios y los mejores ya no es de 1:2, como en el pasado. Es 1:100 o incluso 1:1000*"

Nathan Myhrvold (Ex-director de I+D de Microsoft)

Si el proyecto, más que innovación lo que requiere es la ejecución controlada de un plan detallado, posiblemente sean los procesos de la organización los garantes del resultado; y con un modelo de gestión predictiva, el factor relevante sea la capacidad de los procesos empleados, y no tanto el nivel profesional de las personas del equipo.

Si por ser el valor del producto el objetivo del proyecto, se emplea un modelo de desarrollo ágil, son las personas, y no los procesos, los encargados de proporcionarlo, y en ese caso el equipo debe estar compuesto por personas con el mayor conocimiento y experiencia posible.

Cultura organizativa

Para la ejecución sistemática y controlada de procesos no resulta especialmente relevante el tipo de cultura de la organización.

Sin embargo, para el desarrollo de trabajo basado en el talento de las personas resultan inhibidores los ambientes laborales basados en el control, excesivamente normalizados y jerarquizados.

Entorno de desarrollo

Los entornos de desarrollo basados en procesos son adecuados para modelos de gestión predictiva.

Los entornos de desarrollo basados en las personas son adecuados para modelos de gestión ágil.

Resumen

- Términos empleados para designar a los dos modelos de gestión de proyectos:
  - Predictiva, clásica, tradicional, formal.
  - Ágil, adaptable.
- La gestión de proyectos predictiva se ha desarrollado sobre las premisas
  - Todos los proyectos mantienen características y comportamientos regulares.
  - El objetivo de la ejecución de un proyecto es lograr el producto previsto en el tiempo planificado, sin desbordar los costes estimados.
- Las características de la gestión de proyectos predictiva son: validez para cualquier tipo de proyecto y carácter predictivo.
- La gestión ágil surge al cuestionar la validez de las premisas de la gestión tradicional:
  - No hay una forma única y válida para gestionar cualquier tipo de proyecto.
  - Hay proyectos que tienen como objetivo valor para el producto, y no funcionalidad, fecha y costes.
- Las características relevantes del proyecto para decidir el estilo de gestión más adecuado son: prioridad del negocio, estabilidad de los requisitos, rigidez del producto, coste de prototipado, criticidad del sistema y tamaño del sistema.
- Las características relevantes de la organización para facilitar la elección del modelo de gestión más adecuado son: nivel profesional, cultura organizativa y entorno de desarrollo.

Introducción al modelo Scrum para desarrollo de Software

El origen

Scrum es una marco para la ejecución de prácticas ágiles en el desarrollo de proyectos que toma su nombre y principios de las observaciones sobre nuevas prácticas de producción, realizadas por Hirotaka Takeuchi e Ikujijo Nonaka a mediados de los 80. (ver Gestión Predictiva y Gestión Ágil: El Nuevo Escenario)

Aunque las prácticas observadas por estos autores surgieron en empresas de productos tecnológicos, también se emplean en entornos que trabajan con requisitos inestables y que requieren rapidez y flexibilidad, situaciones frecuentes en el desarrollo de determinados sistemas de software.

En 1995 Ken Schwaber presentó en OOPSLA 95 (Object-Oriented Programming Systems & Applications conference) (Schwaber, 1995), la implementación de Scrum Para software que él empleaba en el desarrollo de Delphi, y Jeff Sutherland en su empresa Easel Corporation (compañía que en los macrojuegos de compras y fusiones se integraría en VMARK, y luego en Informix y finalmente en Ascential Software Corporation).

Las implementaciones de Scrum para desarrollo de software se vienen enriquecendo desde entonces, y poco tienen que ver las implementaciones actuales con la original de Ken (Schwaber, 1995) . Ahora es muy raro que alguien configure un campo de Scrum con los controles originales (paquetes, cambios, riesgos, soluciones…) el Backlog único ha evolucionado a Backlog de producto y Backlog de Sprint. También es habitual usar un backlog estratégico o “Epics” de producto. La evolución añadió a la reunión de revisión de sprint, otra de inicio; y más tarde otra de retrospectiva. Tampoco se suele usar la fase de cierre, etc.

También la prácticas se han enriquecido. En 2001 apareció el gráfico burndown, más tarde empezó a ser habitual el uso de estimación de póquer, luego tableros de control visual kanban…

Introducción al modelo

Scrum es una metodología de desarrollo muy simple, que requiere trabajo duro, porque no se basa en el seguimiento de un plan, sino en la adaptación continua a las circunstancias de la evolución del proyecto.

Como método ágil:

- Es un modo de desarrollo adaptable, antes que predictivo.
- Orientado a las personas, más que a los procesos.
- Emplea el modelo de construcción incremental basado en iteraciones y revisiones.

(ver Gestión Predictiva y Gestión Ágil)

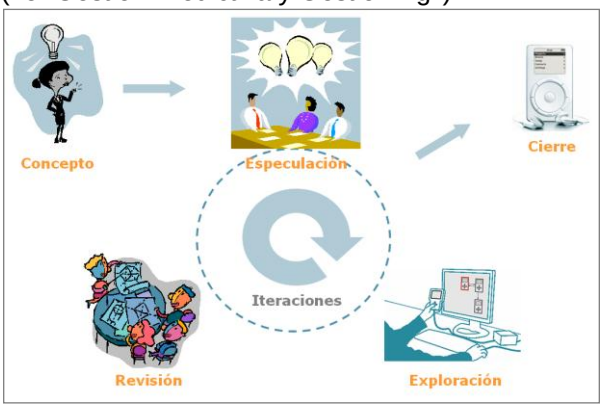


Ilustración 46 - Fases del desarrollo ágil

Comparte los principios estructurales del desarrollo ágil: a partir del concepto o visión de la necesidad del cliente, construye el producto de forma incremental a través de iteraciones breves que comprenden fases de especulación – exploración y revisión. Estas iteraciones (en Scrum llamadas sprints) se repiten de forma continua hasta que el cliente dá por cerrado el producto.

Se comienza con la visión general del producto, especificando y dando detalle a las funcionalidades o partes que tienen mayor prioridad de negocio, y que pueden llevarse a cabo en un periodo de tiempo breve (según los casos pueden tener duraciones desde una semana hasta no más de dos meses).
Cada uno de estos periodos de desarrollo es una iteración que finaliza con la entrega de una parte (incremento) operativa del producto.

Estas iteraciones son la base del desarrollo ágil, y Scrum gestiona su evolución en reuniones breves diarias donde todo el equipo revisa el trabajo realizado el día anterior y el previsto para el siguiente.

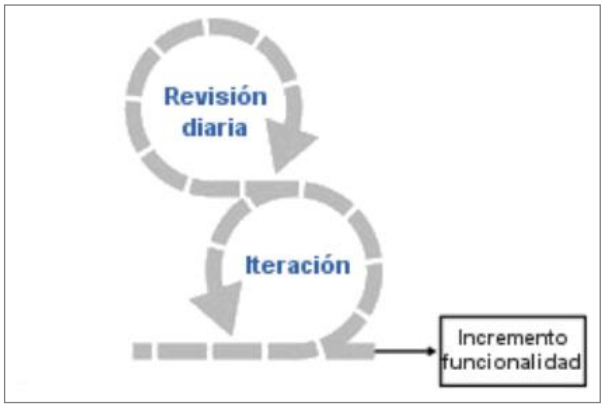


Ilustración 47 - Ciclo central de Scrum

Control de la evolución del proyecto

Scrum controla de forma empírica y adaptable la evolución del proyecto, a través de las siguientes prácticas de la gestión ágil:

Revisión de las Iteraciones

Al finalizar cada iteración (sprint) se lleva a cabo una revisión con todas las personas implicadas en el proyecto. Es por tanto la duración del sprint, el periodo máximo que se tarda en reconducir una desviación en el proyecto o en las circunstancias del producto.

Desarrollo incremental

Las personas implicadas no trabajan con diseños o abstracciones.

El desarrollo incremental implica que al final de cada iteración se dispone de una parte de producto operativa, que se puede inspeccionar y evaluar.

Desarrollo evolutivo

Los modelos de gestión ágil se emplean para trabajar en entornos de incertidumbre e inestabilidad de requisitos.
Intentar predecir en las fases iniciales cómo será el resultado final, y sobre dicha predicción desarrollar el diseño y la arquitectura del producto no es realista, porque las circunstancias obligarán a remodelarlo muchas veces.

¿Para qué predecir los estados finales de la arquitectura o del diseño si van a estar cambiando? Scrum considera a la inestabilidad como una premisa, y se adoptan técnicas de trabajo para permitir la evolución sin degradar la calidad de la arquitectura que también evoluciona durante el desarrollo.

Durante el desarrollo se genera el diseño y la arquitectura final de forma evolutiva. Scrum no los considera como productos que deban realizarse en la primera "fase" del proyecto.
(El desarrollo ágil no es un desarrollo en fases)

Auto-organización

En la ejecución de un proyecto son muchos los factores impredecibles en todas las áreas y niveles. La gestión predictiva confía la responsabilidad de su resolución al gestor de proyectos.
En Scrum los equipos son auto-organizados (no auto-dirigidos), con margen de decisión suficiente para tomar las decisiones que consideren oportunas.

Colaboración

Las prácticas y el entorno de trabajo ágiles facilitan la colaboración del equipo. Ésta es necesaria, porque para que funcione la auto-organización como un control eficaz cada miembro del equipo debe colaborar de forma abierta con los demás, según sus capacidades y no según su rol o su puesto.

Visión general del proceso

Scrum denomina "sprint" a cada iteración de desarrollo y según las características del proyecto y las circunstancias del sprint puede determinarse una duración desde una hasta dos meses, aunque no suele ser recomendable hacerlos de más de un mes.

El sprint es el núcleo central que proporciona la base de desarrollo iterativo e incremental.

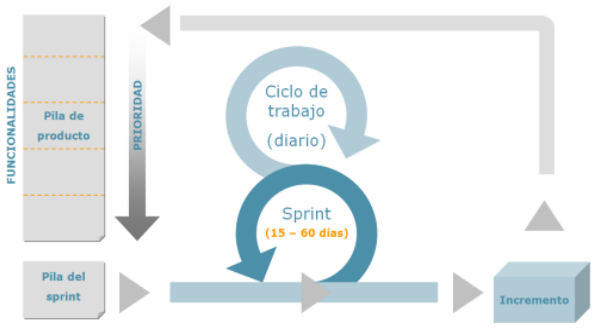


Ilustración 48 - Diagrama de Scrum

Los elementos que conforman el desarrollo Scrum son:

Las reuniones

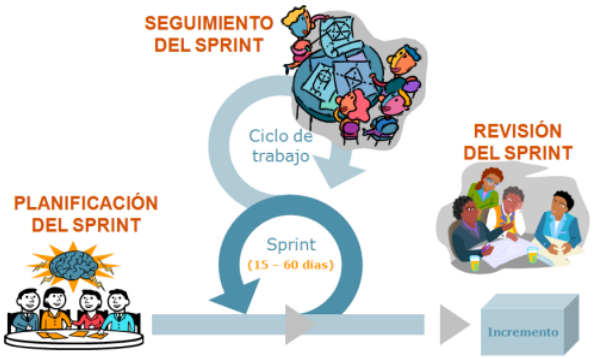


Ilustración 49 - Las reuniones habituales en Scrum

- **Planificación del sprint**: Jornada de trabajo previa al inicio de cada sprint en la que se determina cuál va a ser el trabajo y los objetivos que se deben conseguir en la iteración.

- **Seguimiento del sprint**: Breve revisión diaria, en la que cada miembro describe tres cuestiones:
  1.- El trabajo que realizó el día anterior.
  2.- El que tiene previsto realizar.
  3.- Cosas que puede necesitar o impedimentos que deben suprimirse para realizar el trabajo.
  Cada persona actualiza en la pila del sprint el tiempo pendiente de sus tareas, y con esta información se actualiza también el gráfico con el que el equipo monitoriza el avance del sprint (burn-down)

- **Revisión del sprint**: Análisis y revisión del incremento generado.

Los elementos

- **Pila del producto**: (product backlog) lista de requisitos de usuario que a partir de la visión inicial del producto crece y evoluciona durante el desarrollo.

- **Pila del sprint**: (sprint backlog) lista de los trabajos que debe realizar el equipo durante el sprint para generar el incremento previsto.

- **Incremento**: Resultado de cada sprint

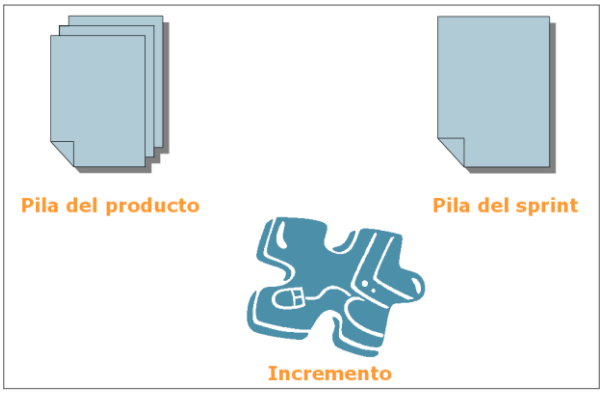


Ilustración 50 - Los elementos de Scrum

Los roles

Todas las personas que intervienen, o tienen relación directa o indirecta con el proyecto, se clasifican en dos grupos: comprometidos e implicados.

En círculos de Scrum es frecuente llamar a los primeros (sin ninguna connotación peyorativa) "cerdos" y a los segundos "gallinas".

El origen de estos nombres es esta metáfora que ilustra de forma gráfica la diferencia entre "compromiso" e "implicación" con el proyecto:

Una gallina y un cerdo paseaban por la carretera.
La gallina preguntó al cerdo: "¿Quieres abrir un restaurante conmigo?".
El cerdo consideró la propuesta y respondió: "Sí, me gustaría. ¿Y cómo lo llamaríamos?".
La gallina respondió: "Jamón con huevos".
El cerdo se detuvo, hizo una pausa y contestó: "Pensándolo mejor, creo que no voy a abrir un restaurante contigo. Yo estaría realmente comprometido, mientras que tu estarías sólo implicada".

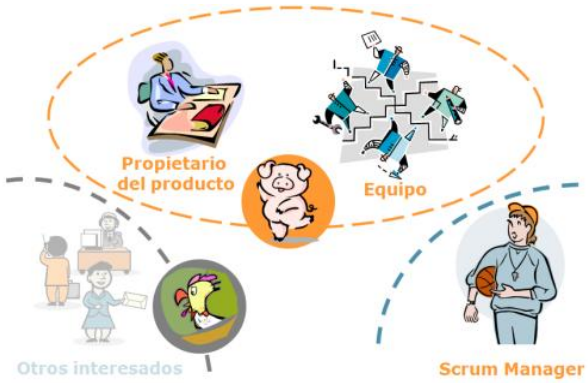


Ilustración 51 - Distribución clásica de roles para Scrum

COMPROMETIDOS (cerdos)	IMPLICADOS (gallinas)
Propietario del producto Equipo	Otros interesados (Dirección general Dirección comercial Marketing Usuarios, etc)

- Propietario del producto: es la persona responsable de lograr el mayor valor de producto para los clientes, usuarios y resto de implicados.
- Equipo de desarrollo: grupo o grupos de trabajo que desarrollan el producto.
- Scrum Manager: Responsable del funcionamiento de la metodología Scrum en la organización.

Algunas implementaciones de modelo Scrum, consideran el rol de gestor de Scrum como “comprometido” y necesario (ScrumMaster)

Con el criterio de Scrum Management, es recomendable que las responsabilidades que cubre el rol de “Scrum Manager” o facilitador para la implantación y mejora de una gestión ágil en toda la organización, estén identificadas en una única persona cuando se comienzan a aplicar prácticas de Scrum en una organización. En organizaciones ágiles maduras puede tener menos sentido.
En cualquier caso, las responsabilidades de Scrum Manager no son del proyecto, sino del grupo de procesos y métodos de la organización, por lo que no debe considerarse ni cerdo ni gallina.

Valores

Scrum es una “carrocería” que da forma a los principios ágiles. Es una ayuda para organizar a las personas y el flujo de trabajo; como lo pueden ser otras propuestas de formas de trabajo ágil: Crystal, DSDM, etc.

La carrocería sin motor, sin los valores que dan sentido al desarrollo ágil, no funciona:

- Delegación de atribuciones (*empowerment*) al equipo para que pueda auto-organizarse y tomar las decisiones sobre el desarrollo.
- Respeto entre las personas. Los miembros del equipo deben confiar entre ellos y respetar sus conocimientos y capacidades.
- Responsabilidad y auto-disciplina (no disciplina impuesta).
- Trabajo centrado en el valor para el cliente y el desarrollo de lo comprometido
- Información, transparencia y visibilidad del desarrollo del proyecto

Resumen

Scrum es un modelo ágil de desarrollo, que toma forma de las prácticas de trabajo, que a partir de los 80 comienzan a adoptar algunas empresas tecnológicas, y que Nonaka y Takeuchi acuñaron como "Campos de Scrum".

El modelo Scrum, aplicado al desarrollo de software, emplea el principio ágil: "desarrollo iterativo e incremental", denominando sprint a cada iteración de desarrollo.

Las prácticas empleadas por Scrum para mantener un control ágil en el proyecto son:

- Revisión de las iteraciones
- Desarrollo incremental
- Desarrollo evolutivo
- Auto-organización del equipo
- Colaboración

Los artefactos del modelo son:
- Elementos:
  - Pila del producto o product backlog
  - Pila del sprint o sprint backlog
  - Incremento
- Roles:
  - Propietario del producto
  - Equipo
  - Scrum Manager
  - Otros interesados
- Reuniones:
  - Planificación del sprint
  - Seguimiento del sprint
  - Revisión del sprint

Los valores que hacen posible a las prácticas de Scrum crear "campos de Scrum" son:

- Autonomía (empowerment) del equipo
- Respeto en el equipo
- Responsabilidad y auto-disciplina
- Foco en la tarea
- Información transparencia y visibilidad

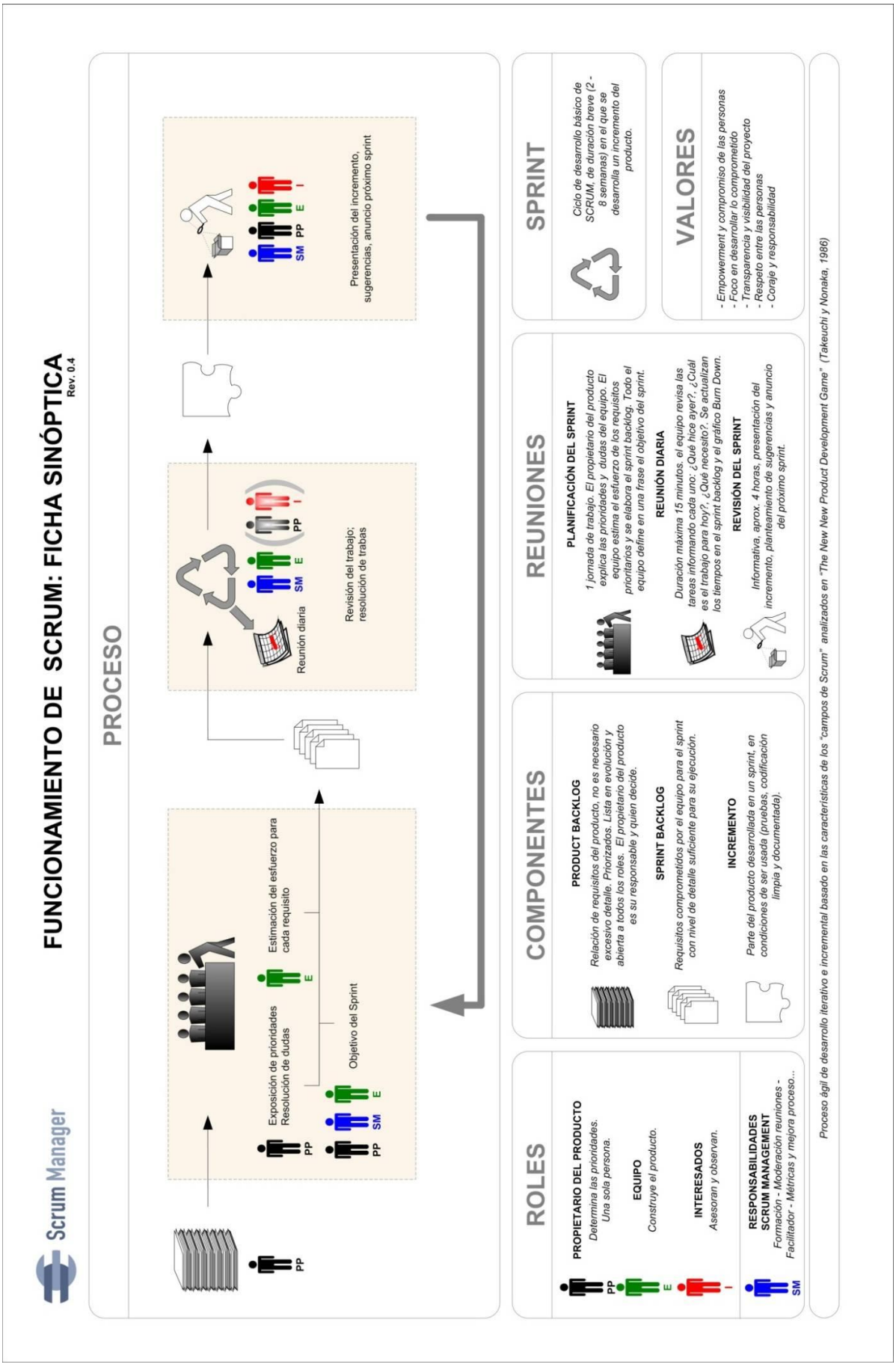


Ilustración 52 - Ficha sinóptica de Scrum

Roles y responsabilidades de proyecto

Introducción

El grado de éxito de Scrum Management en una empresa no depende sólo de los roles y las responsabilidades directamente relacionadas con la gestión de los proyectos (cliente y equipo).
Las organizaciones son realidades sistémicas, inter-relacionadas, y aunque este libro cubre sólo el área de gestión de los proyectos, veremos los roles implicados directamente en la ejecución del proyecto o solución técnica, y el área directiva o de *management* de la organización.

El conjunto de responsabilidades que se deben cubrir de forma coordinada y alineada con la visión de la organización, se clasifican en las tres categorías siguientes:

Responsabilidades generales Scrum Management

Ilustración 53 - Áreas de responsabilidades Scrum Manager

De management

- Equilibrio sistémico de la organización
- Coherencia del modelo
- Medios y formación

De procesos

- Configuración de Scrum
- Mejora continua
- ***Garantía de funcionamiento de Scrum en cada proyecto***

De producción

- ***Producto***
- ***Auto-organización***
- ***Tecnología ágil***

El uso de prácticas y tecnologías ágiles, el trabajo en equipos auto-organizados, disponer de una visión de producto definida y gestionada durante todo el proyecto y garantizar el funcionamiento de scrum durante la ejecución, son responsabilidades directas del ámbito del proyecto.

Que las diferentes áreas de la empresa se encuentren comunicadas y alineadas con una visión común, coherente con un modelo de trabajo ágil, disponga de medios para el diseño e implantación de una implantación ágil adecuada a la empresa, mejora continua del modelo y formación a las personas, son responsabilidades de la organización.

Responsabilidades y roles "del proyecto"

Ilustración 54 - Responsabilidades y roles de proyecto

Éstas son las directamente implicadas en el desarrollo del producto. En las implantaciones rígidas de scrum se asignan a roles fijos denominados "cerdos" (directamente implicados en el proyecto):

- Responsabilidad de funcionamiento de Scrum => A un gestor específico para el funcionamiento de Scrum (Scrum Master)
- Responsabilidad de gestión del producto => a un "propietario de producto", o product manager.
- Responsabilidad de auto-organización y uso de prácticas y tecnologías ágiles => al equipo.

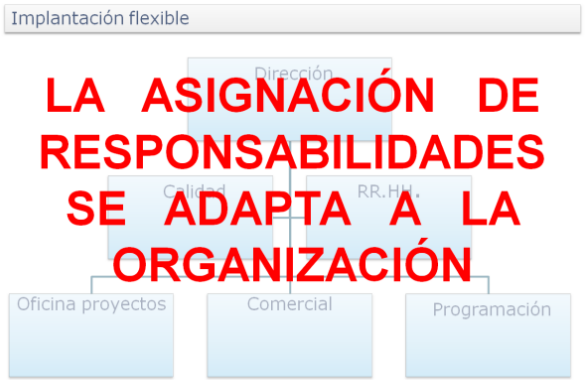
Las del propietario del producto, relativas a la definición desde la visión, la priorización del trabajo y la financiación del proyecto.

Las del equipo, relativas a la auto-organización y uso de prácticas tecnológicas ágiles.

También pertenece al grupo de responsabilidades del proyecto: la garantía de ejecución y funcionamiento correcto de las prácticas Scrum en cada proyecto.
Lo más común en las fases de implantación, cuando los equipos no están familiarizados con el modelo, es la asignación de esta responsabilidad en una persona experta en Scrum, ajena al equipo: el gestor de Scrum, o Scrum Manager.

Ilustración 55 - Responsabilidades Scrum Manager asumidas por los roles habituales de Scrum.

Desde la perspectiva de implantación de prácticas ágiles de Scrum Management, resulta más eficiente adaptar los principios ágiles a la realidad de cada organización, de forma que lo relevante no es importar roles fijos: Product Owner o Scrum Master, sino cubrir adecuadamente todas las responsabilidades.



Una asignación habitual de las responsabilidades de proyecto suele ser sobre los roles:

Garantía de funcionamiento de Scrum => Calidad o procesos

Garantía de gestión de producto => Product manager
Auto-organización y tecnología ágil = Equipo

La visión cerrada de Scrum establece:
Garantía de funcionamiento de Scrum => rol específico: Scrum Master
Garantía de gestión de producto => Product Owner
Auto-organización => Equipo

Tanto si en la implantación de agilidad en la organización, las responsabilidades necesarias se asignan a roles de la estructura de la empresa, o se crean nuevos roles (Product Owner o Scrum Master), lo relevante es que las personas que los desempeñan tengan la experiencia y conocimiento profesional necesario.

El propietario del producto

El propietario del producto o “product owner” es la persona que toma las decisiones del cliente.

Normalmente atribuida a un rol de propietario de producto o product manager.

Para simplificar la comunicación y toma de decisiones es necesario que las responsabilidades de gestión del producto las asuma una única persona.
Si se trata de organizaciones cliente grandes o con varios departamentos, éstas pueden tener la forma de comunicación interna que consideren oportuna, pero en el equipo de desarrollo sólo se integra una persona representando al cliente, y ésta debe tener el conocimiento suficiente del producto y las atribuciones necesarias para tomar las decisiones que le corresponden.

Para ejercer este rol es necesario:

- Conocer perfectamente el entorno de negocio del cliente, las necesidades y el objetivo que se persigue con el sistema que se está construyendo.
- Tener atribuciones suficientes para tomar las decisiones necesarias durante el proyecto.
- Conocer Scrum para realizar con solvencia las tareas que le corresponden:
  - Desarrollo y administración de la pila del producto.

  - Presentación y participación en la reunión de planificación de cada sprint.

- Recibir y analizar de forma continua retro-información del negocio (evolución del mercado, competencia, alternativas…) y del proyecto (sugerencias del equipo, alternativas técnicas, pruebas y evaluación de cada incremento…).

- Es recomendable conocer y haber trabajado previamente con el mismo equipo.

Es quien decide en última instancia cómo será el resultado final, y el orden en el que se van construyendo los sucesivos incrementos: qué se pone y qué se quita de la pila del producto, y cuál es la prioridad de las funcionalidades.

Es responsable de la financiación del proyecto, y las decisiones sobre fechas y funcionalidades de las diferentes versiones del producto, y el retorno de la inversión del proyecto.

En los desarrollos internos para la propia empresa, suele asumir este rol el product manager o el responsable de marketing. En desarrollos para clientes externos: el responsable del proceso de adquisición del cliente.

El equipo

Se recomienda un tamaño de equipo entre 4 y 8 personas.
Más allá de 8 resulta más difícil mantener la agilidad en la comunicación directa, y se manifiestan con más intensidad las rigideces habituales de la dinámica de grupos (que comienzan a aparecer a partir de 6 personas).

No se trata de un grupo de trabajo formado por un arquitecto, diseñador o analista, programadores, pruebas…
Es un equipo multidisciplinario, en el que todos trabajan de forma conjunta para realizar cada sprint.
Las principales responsabilidades, más allá de la auto-organización y uso de tecnologías ágiles, son las que se derivan de la diferencia entre “grupo de trabajo” y “equipo”.

Un grupo de trabajo es un conjunto de personas que realizan un trabajo, con una asignación específica de tareas, responsabilidades y siguiendo un proceso o pautas de ejecución.
Los operarios de una cadena, forman un grupo de trabajo: aunque tienen un jefe común, y trabajan en la misma organización, cada uno responde por su trabajo.

El equipo tiene espíritu de colaboración, y un propósito común: conseguir el mayor valor posible para la visión del cliente.

Un equipo Scrum responde en su conjunto. Trabajan de forma cohesionada y auto-organizada.
No hay un gestor que delimita, asigna y coordina las tareas. Son los propios componentes del equipo los que lo realizan.

En el equipo:

- Todos conocen y comprenden la visión del propietario del producto.
- Aportan y colaboran con el propietario del producto en el desarrollo de la pila del producto.
- Comparten de forma conjunta el objetivo de cada sprint y la responsabilidad del logro.
- Todos los miembros participan en las decisiones.
- Se respetan las opiniones y aportaciones de todos
- Todos conocen el modelo de trabajo con Scrum.

Hay un responsable o líder del equipo que asume las responsabilidades de garantía de funcionamiento del campo de Scrum en el proyecto.

En las fases de implementación de Scrum, con equipos sin demasiada experiencia en desarrollo ágil con Scrum, y en organizaciones con demasiada rotación de personas de los equipos entre proyectos, es recomendable la figura de un gestor de Scrum o Scrum Manager para asumir estas responsabilidades.

Scrum Manager – Team Leader

Es el responsable del funcionamiento de Scrum en el proyecto, cubriendo los aspectos siguientes que la organización necesite según el conocimiento, experiencia con el modelo… o aquellos que no cubra con otras personas con la formación e idoneidad adecuada.

- Asesoría y formación al Propietario del producto.

- Asesoría y formación al equipo.
- Revisión y validación de la pila del producto.
- Moderación de las reuniones.
- Resolución de impedimentos que en el sprint pueden entorpecer la ejecución de las tareas.
- Gestión de la “dinámica de grupo” en el equipo
- Respeto de la organización y los implicados, con las pautas de tiempos y formas de Scrum
- Configuración, diseño y mejora continua de las prácticas de Scrum en la organización.

Lo más habitual es que la garantía de funcionamiento de Scrum en el proyecto se asigne:

- Al rol de un Team Leader, en equipos experimentados en trabajo ágil, en organizaciones que tienen ya una cierta experiencia con agililidad.
- A un puesto específico para contar con esta garantía (Gestor de Scrum o Scrum Master), en equipos y organizaciones en fases tempranas de implementación de Scrum, sin experiencia previa en desarrollo ágil.

Resumen

Las responsabilidades del funcionamiento de Scrum Management en la organización se clasifican en tres niveles y son las siguientes:

De management

- Equilibrio sistémico de la organización
- Coherencia del modelo
- Medios y formación

De procesos

- Configuración de Scrum
- Mejora continua
- Garantía de funcionamiento de Scrum en cada proyecto

De producción

- Producto
- Auto-organización
- Tecnología ágil

El rol de propietario del producto tiene las responsabilidades de producto.

El equipo:

- Auto - organización
- El uso de tecnología y técnicas ágiles en el desarrollo del sistema
- Garantía de funcionamiento de Scrum en el proyecto, cuando no hay un Scrum Manager

El resto de las responsabilidades no son propias del proyecto, y por tanto propias del equipo; sino de la organización.

Los elementos de Scrum

Introducción

Los elementos centrales de ciclo ágil Scrum son:

- Pila del producto (Product Backlog): Lista de funcionalidades que necesita el cliente.
- Pila del sprint (Sprint Backlog): Lista de tareas que se realizan en un sprint
- Incremento: Parte del sistema desarrollada en un sprint

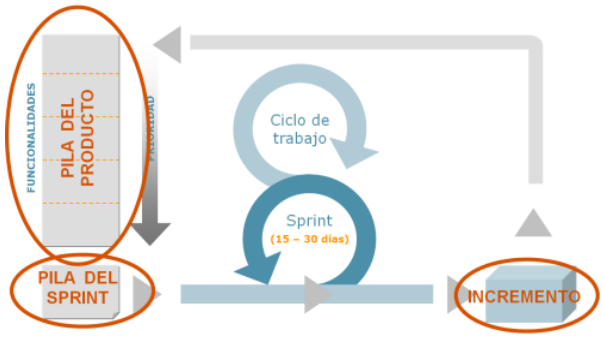


Ilustración 56 - Elementos centrales de Scrum

Este tema describe estos tres elementos. Los dos primeros forman los requisitos del sistema, y el tercero es valor que se le entrega al cliente al final de cada sprint.

Cada incremento es una parte del producto completamente terminada y operativa.

No se deben considerar como incrementos: prototipos, módulos o subrutinas pendientes de pruebas o de integración.

Los requisitos en el desarrollo ágil

La ingeniería del software clásica diferencia dos áreas de requisitos

- Requisitos del sistema
- Requisitos del software

Los requisitos del sistema forman parte del proceso de adquisición (ISO 12207), y por tanto es responsabilidad del cliente la definición del problema y de las funcionalidades que debe aportar la solución.

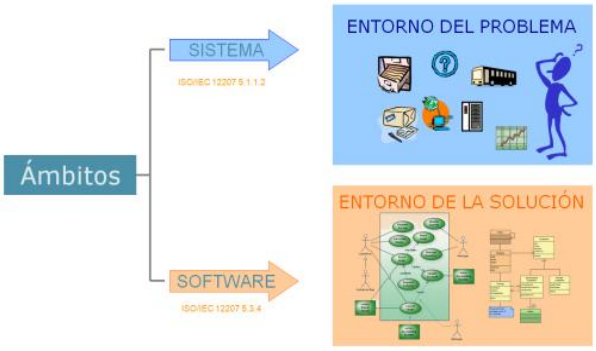


Ilustración 57 - Ámbitos de los requisitos

No importa si se trata de gestión tradicional o ágil. La descripción del sistema es responsabilidad del cliente, aunque se aborda de forma diferente en cada caso.

- En los proyectos predictivos, los requisitos del sistema suelen especificarse en documentos formales; mientras que en los proyectos ágiles toman la forma de pila del producto o lista de historias de usuario.
- Los requisitos del sistema formales se especifican de forma completa y cerrada al inicio del proyecto; sin embargo una pila del producto es un documento vivo, que evoluciona durante el desarrollo.
- Los requisitos del sistema los desarrolla una persona o equipo especializado en ingeniería de requisitos a través del proceso de obtención (elicitación) con el cliente. En Scrum la visión del cliente es conocida por todo el equipo (el cliente forma parte del equipo) y la pila del producto se realiza y evoluciona de forma continua con los aportes de todo el equipo.

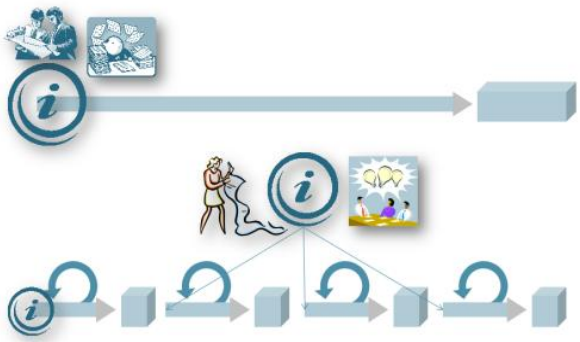


Ilustración 58 - Requisitos en la gestión predictiva vs. requisitos en la gestión ágil

Pero la responsabilidad es del cliente; del "propietario del producto" en el caso de Scrum, que debe decidir qué se incluye en la pila del producto, y el orden de prioridad.

Requisitos y visión del producto

Scrum, aplicado al software, emplea dos formatos para registrar los requisitos:

- Pila del producto (Product Backlog)
- Pila del sprint (Sprint Backlog)

La pila del producto se sitúa en el área de necesidades de negocio desde el punto de vista del cliente. Es el área que en la ingeniería del software tradicional, cubren los requisitos del sistema o ConOps (Concept of Operations).

La pila del sprint cubre la especificación de los requisitos de software necesarios para dar respuesta a las funcionalidades esperadas por el cliente.

Estas listas no tienen por qué cumplir con un determinado “formato scrum-estándar”. Pueden, y deben, adoptar la forma más adecuada al sistema equipo-proyecto.

Algunos equipos ágiles emplean pilas de requisitos, otros historias de usuario, tarjetas kanban, etc…

Lo relevante no es tanto la forma, sino que:

Requisitos del Sistema (pila del producto):

- Las funcionalidades que incluye dan forma a una visión del producto definida y conocida por todo el equipo.
- Las funcionalidades están individualmente definidas, priorizadas y preestimadas.
- Están realizados y gestionados por el cliente (propietario del producto)

Requisitos del software (pila del sprint):

- Incluyen todas las tareas necesarias para construir el incremento de un sprint.
- El equipo ha estimado el esfuerzo de cada tarea.
- El equipo ha asignado cada tarea a un miembro.
- Las duraciones estimadas de las tareas no son ni inferiores, ni superiores a los límites definidos en el equipo.

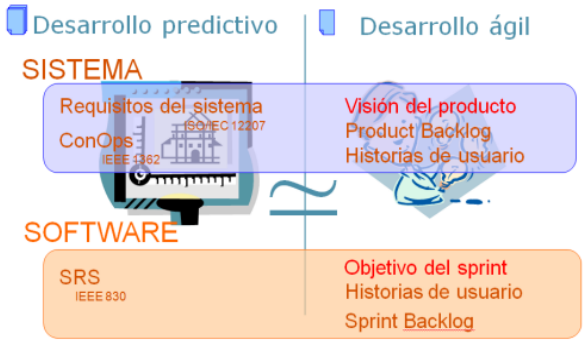


Ilustración 59 - Descripciones de requisitos en la gestión predictiva y en la gestión ágil

Pila del producto: los requisitos del cliente

La pila del producto es el inventario de funcionalidades, mejoras, tecnología y corrección de errores que deben incorporarse al producto a través de las sucesivas iteraciones de desarrollo.

Representa todo aquello que esperan los clientes, usuarios, y en general los interesados. Todo lo que suponga un trabajo que debe realizar el equipo tiene que estar reflejado en esta pila.

Estos son algunos ejemplos de posibles entradas de un backlog:

- Permitir a los usuarios la consulta de las obras publicadas por un determinado autor.
- Reducir el tiempo de instalación del programa.
- Mejorar la escalabilidad del sistema.
- Permitir la consulta de una obra a través de un API web.

A diferencia de un documento de requisitos del sistema, la pila del producto nunca se dá por completada; está en continuo crecimiento y evolución.

Habitualmente se comienza a elaborar con el resultado de una reunión de "fertilización cruzada" o brainstorming; o un proceso de “Exploración” (eXtreme Programming) donde colabora todo el equipo a partir de la visión del propietario del producto.

El formato de la visión no es relevante. Según los casos, puede ser una presentación informal del responsable del producto, un informe de requisitos del departamento de marketing, etc.

Sí que es importante sin embargo disponer de una visión real, comprendida y compartida por todo el equipo.

La pila evolucionará de forma continua mientras el producto esté en el mercado, para darle valor de forma continua, y mantenerlo útil y competitivo.

Para dar comienzo al desarrollo se necesita una visión de los objetivos de negocio que se quieren conseguir con el proyecto, comprendida y conocida por todo el equipo, y elementos suficientes en la pila para llevar a cabo el primer sprint.

Formato de la pila del producto

El desarrollo ágil prefiere la comunicación directa, a la comunicación con documentos.

La pila del producto no es un documento de requisitos, sino una herramienta de referencia para el equipo.

Si se emplea formato de lista, es recomendable que al menos incluya la siguiente información en cada línea:

- Identificador único de la funcionalidad o trabajo.
- Descripción de la funcionalidad.
- Campo o sistema de priorización.
- Estimación

Dependiendo del tipo de proyecto, funcionamiento del equipo y la organización, pueden resultar aconsejables otros campos:

- Observaciones
- Criterio de validación
- Persona asignada
- Nº de Sprint en el que se realiza
- Módulo del sistema al que pertenece
- Etc.

Es preferible no adoptar ningún protocolo de trabajo de forma rígida. El formato del product backlog no es cerrado.

Los resultados de Scrum Management no dependen de la rigidez en la aplicación del protocolo, sino de la institucionalización de sus principios y la implementación en un formato adecuado a las características de la empresa y del proyecto.

Id	Prioridad	Descripción	Est.	Por
1	Muy alta	Plataforma tecnológica	30	AR
2	Muy alta	Interfaz usuario	40	LR
3	Muy alta	Un usuario se registra en el sistema	40	LR
4	Alta	El operador define el flujo y textos de un expediente	60	AR
5	Alta	Etc...	999	XX

Ilustración 60 - Ejemplo de pila de producto

Pila del Sprint

La pila del sprint, (sprint backlog en inglés) es la lista que descompone las funcionalidades de la pila del producto en las tareas necesarias para construir un incremento: una parte completa y operativa del producto.

La realiza el equipo durante la reunión de planificación del sprint, asignando cada tarea a una persona, e indicando en la misma lista cuánto tiempo falta aún para que la termine.

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

Condiciones

- Realizada de forma conjunta por todos los miembros del equipo.
- Cubre todas las tareas identificadas por el equipo para conseguir el objetivo del sprint.
- Sólo el equipo lo puede modificar durante el sprint.
- El tamaño de cada tarea está en un rango de 2 a 16 horas de trabajo.
- Es visible para todo el equipo. Idealmente en una pizarra o pared en el mismo espacio físico donde trabaja el equipo.

Formato y soporte

Tres son las opciones:

- Hoja de cálculo.
- Pizarra física o pared.
- Herramienta colaborativa o de gestión de proyectos.

Y sobre la que mejor se adecua a las características del proyecto, oficina y equipo, lo apropiado es diseñar el formato más cómodo para todos, teniendo en cuenta los siguientes criterios:

- Incluye la información: lista de tareas, persona responsable de cada una, estado en el que se encuentra y tiempo de trabajo que queda para completarla.
- Sólo incluye la información estrictamente necesaria.
- El medio y modelo elegido es la opción posible que más facilita la consulta y comunicación diaria y directa del equipo.
- Sirve de soporte para registrar en cada reunión diaria del sprint, el tiempo que le queda a cada tarea.

Ejemplos

SPRINT	INICIO	DURACIÓN
1	1-mar-07	12

SPRINT BACKLOG			J
			1-mar
Tarea	Estad	Responsal	23
			276
Descripción de la tarea 1	Terminada	Luis	16
Descripción de la tarea 2	Terminada	Luis	12
Descripción de la tarea 3	Terminada	Luis	4
Descripción de la tarea 4	Terminada	Elena	8
Descripción de la tarea 5	Terminada	Elena	16
Descripción de la tarea 6	Terminada	Elena	6
Descripción de la tarea 7	Terminada	Antonio	16
Descripción de la tarea 8	Terminada	Antonio	16
Descripción de la tarea 9	Terminada	Antonio	12
Descripción de la tarea 10	En curso	Luis	12
Descripción de la tarea 11	Pendiente	Luis	8

Ilustración 61 - Ejemplo de pila de sprint en una hoja de cálculo

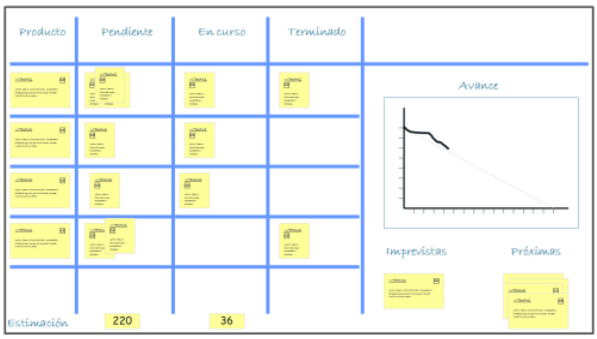


Ilustración 62 - Ejemplo de tablero para seguimiento y registro del sprint

Durante el sprint, el equipo actualiza sobre la pila del sprint, a diario, los tiempos pendientes de cada tarea.
Al mismo tiempo, con estos datos traza el gráfico de avance o "burn-down", que se verá en el tema de "herramientas".

El Incremento

El incremento es la parte de producto producida en un sprint, y tiene como características: que está completamente terminada y operativa, en condiciones de ser entregada al cliente final.
No se trata por tanto de módulos o partes a falta de pruebas, o documentación o…
Idealmente en el desarrollo ágil:

- Cada funcionalidad de la pila del producto se refiere a funcionalidades entregables, no a trabajos internas del tipo "diseño de la base de datos"
- Se produce un "incremento" en cada iteración.

Sin embargo suele ser una excepción habitual el primer sprint. En el que objetivos del tipo "contrastar la plataforma y el diseño" pueden ser normales, e implican trabajos de diseño o desarrollo de prototipos para probar la solvencia de la plataforma que se va a emplear, etc.
Teniendo en cuenta esta excepción habitual, Incremento es:

Parte de producto realizada en un sprint, y potencialmente entregable: TERMINADA Y PROBADA

Si el proyecto o el sistema requiere documentación, o procesos de validación y verificación documentados, o con niveles de independencia que implican procesos con terceros, éstos también tienen que estar realizados para considerar que el producto está "terminado".

Resumen

La pila del producto es la lista de funcionalidades que desea el cliente, ordenadas según la prioridad para él.

Es un documento vivo, en constante evolución durante el desarrollo del sistema.

La pila del sprint es la lista de tareas en las que se han descompuesto las funcionalidades de la pila del producto que se van a desarrollar en un sprint.
Para cada tarea de la pila del sprint se indica la persona que la tiene asignada y el tiempo de trabajo previsto.

Durante el sprint el equipo actualiza a diario en la pila del sprint los tiempos pendientes de cada tarea.
Incremento es la parte de producto desarrollada en un sprint, y se debe encontrar completamente terminada y probada.

Scrum: Las reuniones

Introducción

Scrum realiza el seguimiento y la gestión del proyecto a través de las tres reuniones que forman parte del modelo:

- Planificación del sprint
- Seguimiento del sprint
- Revisión del sprint

Este tema describe los objetivos y protocolos recomendados para cada una.

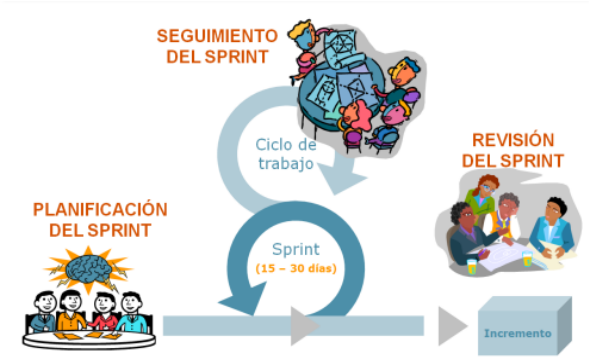


Ilustración 63 - Reuniones en Scrum

Planificación del sprint

Descripción general

En esta reunión se toman como base las prioridades y necesidades de negocio del cliente, y se determina cuáles y cómo van a ser las funcionalidades que incorporará el producto tras el siguiente sprint.

En realidad es una reunión que consta de dos partes:

- En la primera, que puede tener una duración de una a cuatro horas, se decide qué elementos de la pila del producto se van a desarrollar.

- En la segunda se desglosan éstos para determinar las tareas necesarias, estimar el esfuerzo para cada una, y asignarlas a las personas del equipo.

La planificación del sprint no debe durar más de un día. Las características de la reunión son:

Pre-condiciones

- La organización tiene determinados los recursos disponibles para llevar a cabo el sprint.
- El propietario del producto tiene preparada la pila del producto, con su criterio de prioridad para el negocio, y un nº suficiente de elementos para desarrollar en el sprint.
- Siempre que sea posible, el propietario del producto debe haber trabajado antes con el equipo. De esta forma su estimación previa del trabajo que se puede realizar en el sprint será bastante ajustada.
- El equipo tiene un conocimiento de las tecnologías empleadas, y del negocio del producto suficiente para realizar estimaciones basadas en "juicio de expertos", y para comprender los conceptos del negocio que expone el propietario del producto.

Entradas

- La pila del producto.
- El producto desarrollado hasta la fecha a través de los sucesivos incrementos (excepto si se trata del primer sprint)
- Circunstancias de las condiciones de negocio del cliente y del escenario tecnológico empleado.

Ilustración 64 - Reunión de planificación del sprint

Resultados

- Pila del sprint.
- Duración del sprint y fecha de la reunión de revisión.
- Objetivo del sprint.

Es una reunión conducida por el responsable del funcionamiento de Scrum (Scrum Manager, o un miembro del equipo en equipos ya expertos en trabajo con Scrum) a la que deben asistir el propietario del producto y el equipo completo, y a

la que también pueden asistir otros implicados en el proyecto.
La reunión comienza con la presentación del propietario de la pila de producto (product backlog), en la que expone los resultados que por orden de prioridad necesita; especialmente los que prevé, se podrán desarrollar en el siguiente sprint.
Si la pila del producto ha tenido cambios significativos desde la anterior reunión; explica las causas que los han ocasionado.
El objetivo es que todo el equipo conozca las razones y los detalles con el nivel necesario para estimar el trabajo necesario.

Formato de la reunión

Esta reunión marca el inicio de cada sprint. Una persona con la responsabilidad de procesos en la organización[5] es el responsable de su organización y gestión.
Duración máxima: un día.
Deben asistir: el propietario del producto, el equipo y el Scrum Manager (o responsable de este rol)
Pueden asistir: es una reunión abierta a todos los que puedan aportar información útil.
Consta de dos partes separadas por una pausa de café o comida, según la duración.

Primera parte:

Duración de 1 a 4 horas.
Propietario del producto:

- Presenta las funcionalidades de la pila del producto que tienen mayor prioridad y que estima se pueden realizar en el sprint.
- La presentación se hace con un nivel de detalle suficiente para transmitir al equipo toda la información necesaria para construir el incremento.

El equipo

- Realiza las preguntas y solicita las aclaraciones necesarias.
- Propone sugerencias, modificaciones y soluciones alternativas.

Las aportaciones del equipo pueden suponer modificaciones en la pila. De hecho no es que "puedan" es que "deben" suponerlas.
Esta reunión es un punto caliente del protocolo de Scrum para favorecer la fertilización cruzada de ideas en equipo y añadir valor a la visión del producto.

[5] En las organizaciones en fase de implantación es recomendable la figura de un "Scrum Manager" que centraliza todas las responsabilidades para garantizar el funcionamiento de Scrum en la organización.

Tras reordenar y replantear las funcionalidades de la pila del producto, el equipo define el "objetivo del sprint" o frase que sintetiza cuál es el valor que se le va a entregar al cliente.

Exceptuando sprints dedicados exclusivamente a re-factorización o a colecciones de tareas desordenadas (que deberían ser los menos), la elaboración de este lema de forma conjunta en la reunión es una garantía de que todo el equipo comprende y comparte la finalidad del trabajo; y durante el sprint sirve de criterio de referencia en las decisiones que auto-gestiona el equipo.

Ilustración 65 - Formato de la reunión de planificación del sprint

Segunda parte:

En la segunda parte, que puede alargarse hasta el final de la jornada:
El equipo desglosa cada funcionalidad en tareas, y estima el tiempo para cada una de ellas, determinando de esta forma las tareas de la pila del sprint.
En este desglose el equipo tiene en cuenta los elementos de diseño y arquitectura que deberá incorporar el sistema.
Los miembros del equipo se auto-asignan las diferentes tareas tomando como criterios sus conocimientos, intereses y distribución homogénea del trabajo.
Esta segunda parte debe considerarse como una "reunión del equipo", en la que deben estar todos sus miembros y ser ellos quienes descomponen, estiman y asignan el trabajo.

El papel del propietario del producto es atender a dudas y comprobar que el equipo comprende y comparte su objetivo.
El Scrum Manager[1] actúa de moderador de la reunión.

Funciones del rol de Scrum Manager[1]

El Scrum Manager es responsable y garante de:

1.- Se realiza esta reunión antes de cada sprint.

2.-Antes de la reunión el propietario del producto dispone de una pila adecuada y suficiente para realizar el sprint.
3.- El diálogo principal de la reunión se realiza entre el propietario del producto y el equipo. Otros asistentes pueden participar, pero su colaboración no puede implicar toma de decisiones ni limitar el diálogo principal.
4.- La reunión es un trabajo de colaboración activa entre los dos protagonistas: cliente y equipo, y concluyen con un acuerdo sobre el incremento de producto que van a realizar en el sprint.
5.- El equipo comprende la visión y necesidades de negocio del cliente.
6.- El equipo ha realizado una descomposición y estimación del trabajo realistas, y ha considerado las posibles tareas necesarias de análisis, investigación o apoyo.
7.- Al final de la reunión están objetivamente determinados:

- Los elementos de la pila del producto que se van a ejecutar.
- El objetivo del sprint.
- La pila del sprint con todas las tareas estimadas y asignadas.
- La duración del sprint y la fecha de la reunión de revisión.

El Scrum Manager modera la reunión para que no dure más de un día. Debe evitar que el equipo comience a profundizar en trabajos de análisis o arquitectura que son propios del sprint.

Pizarra de trabajo

Es recomendable, que el propietario del producto emplee una hoja de cálculo, alguna herramienta similar, o el soporte de una intranet, para guardar en formato digital la pila del producto
Pero no es aconsejable emplearla como base para trabajar sobre ella en la reunión, proyectándola sobre la pantalla de la sala.
Es mucho mejor trabajar y manipular elementos físicos; y usar una pizarra y fichas removibles (adhesivas, chinchetas, magnéticas).

Ilustración 66 - Ejemplo de pizarra de trabajo para la reunión de planificación del sprint

Un ejemplo de pizarra

La pizarra facilita la comunicación y el trabajo de la reunión.
Al final de la reunión el propietario del producto registrará en la hoja de cálculo, o en la herramienta que emplee, el estado y las modificaciones en la pila del producto.
El equipo hará lo mismo con la pila del sprint.
Según la distribución y espacio de la oficina, quizá se reutilice la pizarra o las notas para el seguimiento del sprint; o quizá no.

Algunos soportes que suelen emplearse:

- Pizarra blanca y fichas adhesivas tipo "Post-it"
- Pizarra de corcho laminado y chinchetas para sujetar las fichas.
- Pizarra de acero vitrificado y soportes magnéticos para sujetar las fichas.

Se puede conseguir una solución práctica y económica empleando fichas adhesivas ("Post-it") y usando como pizarra cartón pluma blanco de 5mm. fijado con puntas directamente sobre la pared.
El cartón pluma es un material ligero, de acabado satinado que puede adquirirse en tiendas de materiales para bellas artes y manualidades.

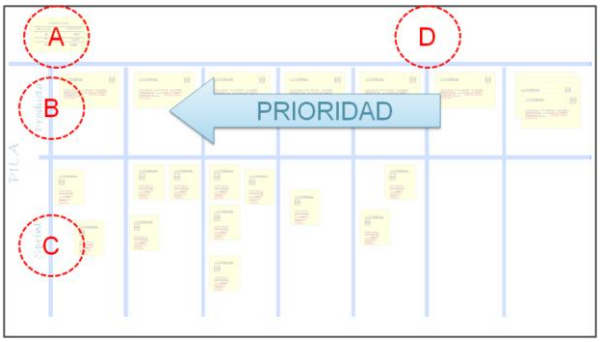


Ilustración 67 - Ejemplo de pizarra de trabajo para la reunión de planificación del sprint

Con cinta adhesiva removible se marcan líneas para delimitar:

- Un área superior donde el Scrum Manager coloca al principio de la reunión la capacidad real del sprint a 3, 4 y 5 semanas (A); y al final (D), las notas con: el objetivo establecido, duración del sprint, funcionalidades de la pila del producto comprometidas, hora fijada para las reuniones diarias y fecha prevista para la reunión de revisión del sprint.
- B.- Una franja para ordenar los elementos de la pila del producto de mayor a menor prioridad.
- C.- Una franja paralela para descomponer cada elemento de la pila del producto en las correspondientes tareas de la pila del sprint.

En cada ficha se refleja la información básica para las decisiones de la reunión: priorización, estimación, descomposición y asignación a los miembros del equipo.
Las siguientes imágenes muestran un ejemplo de uso:

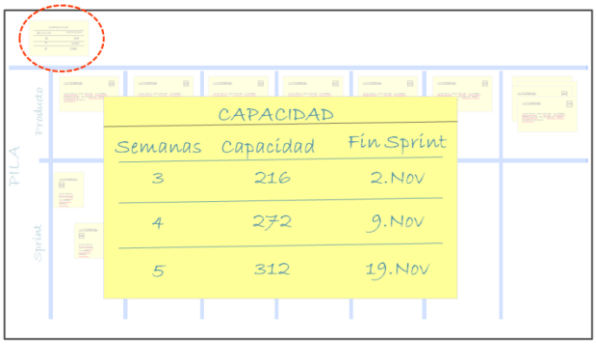


Ilustración 68 - Ejemplo de pizarra de trabajo

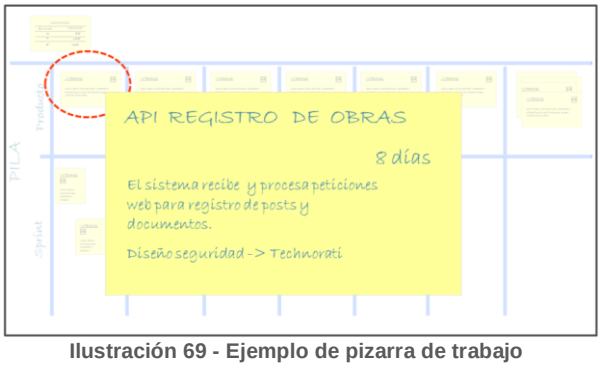


Ilustración 69 - Ejemplo de pizarra de trabajo

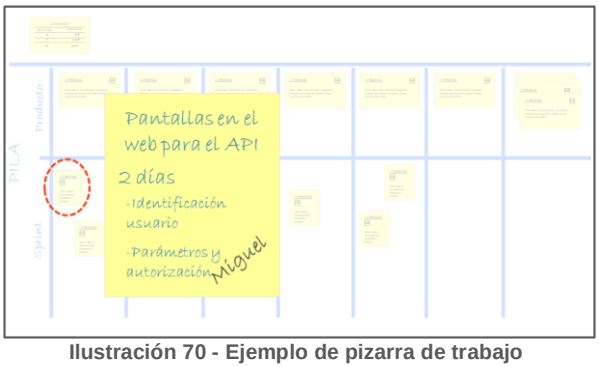


Ilustración 70 - Ejemplo de pizarra de trabajo

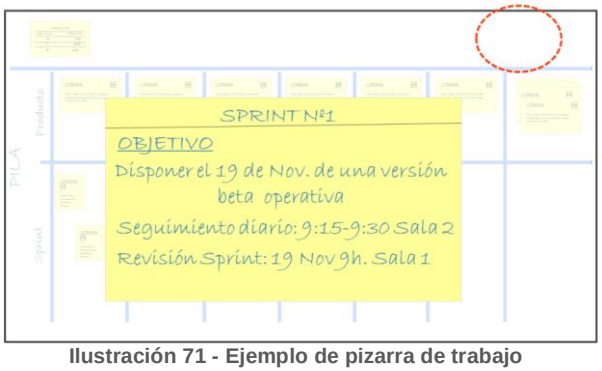


Ilustración 71 - Ejemplo de pizarra de trabajo

Algunas marcas comerciales, entre ellas Post-it comercializan tarjetas adhesivas, con fondo rayado, similares a fichas que resultan especialmente apropiadas, porque no se adhieren entre ellas, pero sí a las pizarras.

Seguimiento del sprint

Descripción

Reunión diaria breve, de no más de 15 minutos, en la que cada miembro del equipo dice las tareas en las que está trabajando, si se ha encontrado o prevé encontrarse con algún impedimento, y actualiza sobre la pila del sprint las ya terminadas, o los tiempos de trabajo que les quedan.

Entradas

Pila del sprint y gráfico de avance (burn-down) actualizados con la información de la reunión anterior.
Información de las tareas realizadas por cada componente del equipo

Resultados

Pila del sprint y gráfico de avance (burn-down) actualizados.

Identificación de necesidades e impedimentos.

Formato de la reunión

Se recomienda realizarla de pie y emplear un formato de pila de tareas en una pizarra, junto con el gráfico de avance del sprint, para que todo el equipo pueda ver y anotar.
En la reunión está presente todo el equipo, y pueden asistir también otras personas relacionadas con el proyecto o la organización, pero éstas no pueden intervenir.

Cada miembro del equipo expone estas tres cuestiones:

1.- Tarea en la que trabajó ayer.
2.- Tarea o tareas en las que trabajará hoy.
3.- Si va a necesitar alguna cosa especial o prevé algún impedimento para realizar su trabajo.

Y actualiza sobre el sprint backlog el tiempo de trabajo que queda pendiente en las tareas que tiene asignadas, o marca como finalizadas las ya completadas.

Al final de la reunión:

- Con las estimaciones actualizadas, el equipo refresca el gráfico de avance del sprint.
- El Scrum Manager (o responsable de la gestión de procesos de la organización) comienza la gestión de necesidades e impedimentos identificados.

Revisión del sprint

Descripción

Reunión realizada al final del sprint en la que, con una duración máxima de 4 horas, el equipo presenta al propietario del producto, clientes, usuarios, gestores… el incremento construido en el sprint.

Objetivos

- El propietario del producto obtiene información objetiva del progreso del sistema. Esta reunión marca a intervalos regulares, el ritmo de construcción del sistema y la trayectoria que va tomando la visión del producto.
- Al ver y probar el incremento, el propietario del producto, y el equipo en general obtienen feedback clave para evolucionar y dar más valor a la pila del producto.
- Otros ingenieros y programadores de la empresa también pueden asistir para conocer cómo trabaja la tecnología empleada.
- El Scrum Manager obtiene retro-información sobre buenas prácticas y problemas durante el sprint, necesaria para las prácticas de ingeniería de procesos y mejora continua de la implementación Scrum Management.

Pre-condiciones

- Se ha concluido el sprint.
- Asiste todo el equipo de desarrollo, el propietario del producto, el responsable de procesos de la empresa y todas las personas implicadas en el proyecto que lo deseen.

Entradas

- Incremento terminado.

Resultados

- Feedback para el propietario del producto: hito de seguimiento de la construcción del sistema, e información para mejorar el valor de la visión del producto.
- Feedback para el Scrum Manager (o responsable de la gestión de procesos de la organización): buenas prácticas y problemas durante el sprint.
- Convocatoria de la reunión del siguiente sprint.

Formato de la reunión

Es una reunión informal. El objetivo es ver el incremento y trabajar en el entorno del cliente. Están prohibidas las presentaciones gráficas y "powerpoints".
El equipo no debe invertir más de una hora en preparar la reunión, y lo que se muestra es el resultado final: terminado, probado y operando en el entorno del cliente (incremento).
Según las características del proyecto puede incluir también documentación de usuario, o técnica.

Es una reunión informativa. NO TIENE UNA MISIÓN ORIENTADA A TOMAR DECISIONES, NI A CRITICAR EL INCREMENTO. Con la información generada en la preparación del siguiente

sprint se expondrán y tratarán las posibles modificaciones sobre la visión del producto.

Un protocolo recomendado:
1.- El equipo expone el objetivo del sprint, la lista de funcionalidades que se incluían y las que se han desarrollado.
2.- El equipo hace una introducción general del sprint y demuestra el funcionamiento de las partes construidas.
3.- Se abre un turno de preguntas y sugerencias sobre lo visto. Esta parte genera información muy valiosa para que el propietario del producto, y el equipo en general, puedan mejorar el valor de la visión del producto.
4.- El Scrum Manager, de acuerdo con las agendas del propietario del producto y el equipo cierra la fecha para la reunión de preparación del siguiente sprint.

¿Retrospectiva?

Desde la premisa de flexibilidad de Scrum Manager, se puede considerar también la realización de reuniones retrospectivas al final de cada sprint, o de cada versión de producto o cada cierto periodo de tiempo.

Conviene señalar que una reunión retrospectiva no es lo mismo que la reunión de revisión del sprint.
Al igual que los modelos de procesos incorporan prácticas de "ingeniería de procesos" para conseguir una mejora continua de su capacidad, en agilidad también van surgiendo prácticas para lo que sería el equivalente de mejora continua de la agilidad de la organización; y en esta línea, las reuniones retrospectivas son un una "meta-práctica" ágil.

El hecho de que se realicen normalmente al final de cada sprint lleva a veces a confusión y a tomarlas como reuniones de "revisión de sprint", cuando suele ser aconsejable considerarlas como diferentes, porque sus objetivos son diferentes.

El objetivo de la revisión del sprint es analizar "QUÉ" se está construyendo, mientras que una reunión retrospectiva se centra en "CÓMO" lo estamos construyendo: "CÓMO" estamos trabajando, con el objetivo de analizar problemas y aspectos mejorables.

Resumen

La gestión y evolución de un proyecto con Scrum se determina en tres reuniones:
- Planificación del sprint.
- Seguimiento del sprint.
- Revisión del sprint.

Planificación del sprint
- Duración máxima 1 día.
- Se determinan las funcionalidades que se desarrollarán en el sprint.
- Cada funcionalidad se desglosa en tareas
- Cada tarea se estima y se asigna a una persona del equipo.
- El resultado es la pila del sprint.

Seguimiento del sprint
- Breve reunión diaria en la que el equipo revisa la evolución del sprint.
- Cada uno expone la tarea en la que ha estado trabajando, en cuál va a trabajar y si necesita algo para poderla realizar.
- Cada miembro actualiza la estimación de tiempo pendiente de sus tareas.

Revisión del sprint
- Duración máxima 4 horas.
- Muestra el incremento desarrollado a todas las personas implicadas en el proyecto.
- No debe confundirse con una reunión retrospectiva para mejora de las prácticas ágiles. El foco es el producto, no las prácticas ágiles de la organización.

Medición: consideraciones

Introducción

¿Por qué medir?

La información es la materia prima de la toma de decisiones, y la que puede ser objetivamente cuantificada proporciona criterios objetivos de gestión y seguimiento.

Desde los niveles concretos de la programación, hasta los más amplios de la gestión global de la compañía, Scrum Management considera tres fondos de escala, o de zoom con los que se puede medir el trabajo

- Desarrollo y gestión de la solución técnica.
- Gestión de proyecto.
- Gestión de la organización.

En el primero se puede medir, por ejemplo, la proporción de polimorfismo del código de un programa, en el segundo, el porcentaje de trabajo realizado, y en el tercero, también por ejemplo, el nivel de satisfacción laboral.

Este curso cubre el área de gestión de proyectos, desde una perspectiva ágil; pero en esta introducción se exponen consideraciones generales, comunes a los tres.

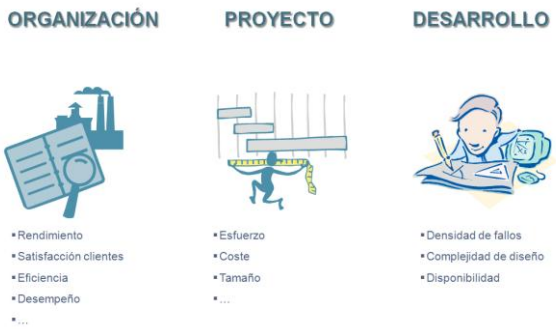


Ilustración 72 - Ámbitos de aplicación de métricas

Flexibilidad y sentido común

Medir es costoso

Antes de incorporar un procedimiento de medición, se debe cuestionar su conveniencia, y la forma en la que se aplicará.

Medir no es un fin en sí mismo

No se deben implantar procesos de medición tan sólo porque sí.
Tomar una lista más o menos "prestigiosa" de métricas e incorporarla a los procedimientos de la empresa, puede tentar por la imagen de profesionalidad que transmitirá un escenario de trabajo monitorizado con indicadores y complejas mediciones, pero no dice mucho a favor de cómo se han analizado y adaptado esas métricas a la realidad de los proyectos y la empresa.

Criterios para el diseño y aplicación de métricas

Cuantas menos, mejor

- Medir es costoso
- Medir añade burocracia
- El objetivo de "Scrum Management" es trabajar con la mejor relación valor / simplicidad.

Cuestiones para cada indicador:

¿Por qué se va a usar?
¿Cuál es el valor por incorporarlo?
¿Cuál por omitirlo?
¿Se pueden tomar decisiones de gestión sin esa información?

La cita "No se puede gestionar lo que no se puede medir", por la redondez de su forma, tienta a considerarla incuestionable. Se desarrollan así patrones de gestión que renuncian a la experiencia, capacidad y sentido común del gestor, incluso en las ocasiones en las que éste puede ser suficiente; y se termina reclamando métricas, produciendo gestores mecanicistas.

El sentido común puede bastar, por ejemplo, para tomar decisiones de gestión de personal en una empresa de ingeniería de 10 empleados, sin necesidad de procedimientos de medición del clima laboral; pero sin embargo éstos son necesarios en empresas con cientos de personas en sedes y departamentos diferentes.

El objetivo de la gestión Scrum es el valor, y la cuestión clave para la incorporación de indicadores en la gestión de proyectos es:

¿Cómo contribuye el uso de este indicador en el valor que el proyecto va a aportar al cliente?

VALOR APORTADO VALOR POR OMITIRLA

Ilustración 73 - criterios para la aplicación de una métrica

¿El indicador es apropiado para el fin que se debe conseguir?

Medir no es, o no debería ser, un fin en sí mismo.

¿Cuál es el fin?
¿Cumplir agendas, mejorar la eficiencia del equipo, las previsiones…?

Sea crítico. El que lo haga, o diga que lo hace la mayoría, no es una razón. Si después de analizarlo no le convence, si prefiere no incorporar esa métrica, o modificarla: usted es el gestor.

Determinar qué medir es la parte más difícil. En el mejor de los casos, las decisiones erróneas sólo supondrán un coste de gestión evitable; pero muchas veces empeorarán lo que se intentaba mejorar.

Medición de la eficiencia en la empresa

La organización XYZ, dedicada al desarrollo de software, está integrando un sistema de calidad y mejora integral para toda la empresa, que incluye métricas para conocer el grado de eficiencia en cada departamento.

Para el de RR.HH. se ha diseñado e implantado un indicador habitual para estos casos, que determina la eficiencia en los procesos de selección de personal.
Mide el coste de cada proceso de selección (anuncios, selección de currículos, entrevistas…) y lo divide por el número de puestos cubiertos.

Como no sólo es importante la eficiencia, sino también la satisfacción del cliente (interno en este caso, que será el departamento que solicita personal) esta métrica se combina con otra para determinarlo: tiempo de respuesta.
Cuánto tiempo se tarda en cubrir las vacantes.

La implantación del indicador ha dado buenos resultados: desde su puesta en marcha, el departamento de RR.HH. ha comenzado a ser más eficiente y conseguir mayor satisfacción de su cliente:
1.- Va reduciendo los costes que suponen la incorporación de nuevas personas a la empresa.
2.- Va reduciendo los tiempos de respuesta a los departamentos que solicitan nuevo personal.

Medición del avance del proyecto

La organización XYZ ha diseñado un cuadro de información para mostrar el grado de avance de cada proyecto.

Los indicadores de progreso de los proyectos, y de las tareas en las que se descomponen, se elaboran con el sistema clásico de la gestión de proyectos predictiva:

1.- Se realiza la estimación del tiempo de trabajo necesario para cada tarea.
2.- Diariamente se actualizan los tiempos de trabajo invertidos.
3.- La diferencia muestra el porcentaje ejecutado de las tareas, y por tanto, el de cada proyecto.

Medición de la eficiencia de los trabajos de programación

La organización XYZ ha adoptado métricas estándar de eficiencia de Ingeniería del Software:
LOC/Hour: Número total de líneas de código nuevas o modificadas en cada hora.

Además para motivar la productividad, ha vinculado los resultados de esta métrica a la retribución por desempeño de los programadores. El resultado ha sido producir más líneas de código sin incrementar la plantilla.

Para evitar que se trate de un incremento “hueco” de líneas de código, o que conlleve un aumento de los errores por programar más deprisa, se ha dotado de mayor “rigor” al sistema de métrica, incorporando al poco tiempo otras métricas para complementar y mejorar el sistema de calidad:

Test Defects/KLOC, Compile Defects/KLOC y Total Defects/KLOC, para controlar que no aumenten el número de errores deslizados en el código.
También se incorporaron indicadores “appraisal time” para medir tiempo y costes del diseño y la ejecución de las revisiones de código.

Y por el temor a que el sistema de medición pueda resultar excesivamente costoso se incorporan también indicadores de coste de calidad (COQ) que miden los tiempos de revisión y los contrastan con las mejoras en los tiempos eliminados por reducción de fallos.

¿Lo que vamos a medir es un indicador válido de lo que queremos conocer?

Hay tareas de programación relativamente mecánicas, orientadas más a la integración y configuración que en al desarrollo de nuevos sistemas.
Para aquellas puede resultar medianamente acertado considerar la eficiencia como volumen de trabajo realizado por unidad de tiempo.

Para las segundas sin embargo, es más apropiado pensar en la cantidad de valor integrado por unidad de desarrollo; expresadas éstas en horas, iteraciones o puntos de función.

¿Qué queremos conocer: la cantidad de líneas de programa, o el valor que entregamos al cliente?
¿Está relacionado lo uno con lo otro?
¿Se puede medir objetivamente el valor entregado al cliente?

En nuestro trabajo son muchos los parámetros que se pueden medir con criterios objetivos y cuantificables: el tiempo de tarea, los tiempos delta, y los de las interrupciones, el nº de puntos de función, la inestabilidad de los requisitos, la proporción de acoplamiento, el nº de errores por línea de código…

¿No estaremos muchas veces midiendo esto, simplemente porque es cuantificable?

¿No estaremos midiendo el nº de líneas que desarrollan las personas cuando en realidad queremos saber el valor de su trabajo?

¿No nos estará pasando lo mismo cuando pretendemos medir: la facilidad de uso, la facilidad de mantenimiento, la flexibilidad, la transportabillidad, la complejidad, etc.?

Resumen

Las métricas se pueden aplicar en el nivel de gestión de la organización, de gestión de los proyectos o de construcción de la solución técnica.

En el diseño e implantación de métricas se debe considerar:

Usar el menor número de métricas posible.

¿El indicador es apropiado para el fin que se debe conseguir?

¿Lo que vamos a medir es un indicador válido de lo que queremos conocer?

Medición: Las Unidades

Velocidad, trabajo y tiempo

Velocidad, trabajo y tiempo son las tres magnitudes que mide la gestión de proyectos ágil, y componen la fórmula de la velocidad, definiéndola como: cantidad de trabajo realizada en por unidad de tiempo.

Velocidad = Trabajo / Tiempo

Tiempo

El desarrollo ágil emplea la técnica "timeboxing"[6] para gestión de tiempo. En el caso de Scrum, la unidad de tiempo para cada incremento de producto es el Sprint.

Tiempo real y tiempo ideal

Antes de continuar, una observación: la diferencia, al hablar de tiempo, entre tiempo "real" y tiempo "ideal".

Ilustración 74 - Tiempo ideal y tiempo real

Tiempo real, es el efectivo de trabajo. Es equivalente a la jornada laboral.
Para un equipo de cuatro personas con jornada laboral de ocho horas el tiempo real en una semana (cinco días laborables) es:

4 * 8 * 5 = 160 horas

El tiempo real de ese equipo en un sprint de 12 días de trabajo es:

4 * 8 * 12 = 384 horas

Sin embargo el término "tiempo ideal" se refiere al tiempo de trabajo necesario, en "condiciones ideales", esto es, sin ninguna interrupción, pausa, distracción o atención a tareas ajenas a la tarea del sprint que se tiene asignada.
Es el concepto similar al que PSP[7] denomina "Delta Time".

Trabajo

Medir el trabajo puede ser necesario por dos razones: para registrar el ya hecho, o para estimar anticipadamente, el que hay que realizar.
En ambos casos se necesita una unidad, y un criterio objetivo de cómo se cuantifica.

Ilustración 75 - Velocidad

Trabajo ya realizado

Medir el trabajo ya realizado no entraña especial dificultad.
Se puede hacer con unidades relativas al producto (p. ej. líneas de código) o a los recursos empleados (coste, tiempo de trabajo…)

Para medirlo basta contabilizar lo ya realizado, empleando las unidades con las que se opere: líneas de código, horas trabajadas (reales o teóricas)…

Medir el trabajo realizado NO es una métrica Ágil.

> ***LA GESTIÓN ÁGIL NO DETERMINA EL GRADO DE AVANCE DEL PROYECTO POR EL TRABAJO YA REALIZADO, SINO POR EL PENDIENTE DE REALIZAR***

Es posible que otros procesos de la organización necesiten registrarlo y medirlo, pero no la gestión ágil de proyectos.

Trabajo pendiente de realizar

Scrum mide el trabajo pendiente para:

- Estimar el esfuerzo y la duración prevista para completar el trabajo (tareas, historias de usuario o epics).
- Determinar el avance del proyecto, y en especial en cada sprint.

[6] Timeboxing, se suele traducir por "tiempo limitado" es un aspecto común de las metodologías ágiles, que marcan iteraciones para cada incremento, determinando qué tareas se van a realizar, y en qué tiempo.

[7] Personal Software Process

Ilustración 76 - Para qué medir

Para Scrum Management, estimar con precisión, de forma cuantitativa y objetiva el trabajo que necesitará la construcción de un requisito, es un empeño más que cuestionable.

El trabajo de un requisito no se puede cuantificar de forma absoluta, porque las funcionalidades no son realidades de solución única.

La "cantidad de trabajo" que consumirá cada funcionalidad o cada historia de usuario no se puede calcular de forma absoluta y objetiva; y en el caso de que se pudiera, la complejidad de la medición haría que la métrica resultante fuera demasiado pesada para la gestión ágil.

Y si no resulta posible estimar con precisión la cantidad de trabajo que hay en un requisito, tampoco se puede saber cuánto tiempo costará, porque además de la incertidumbre del trabajo, se suman las inherentes al "tiempo":

- No es realista hablar de la cantidad o de la calidad del trabajo que realiza una persona por día, o por hora, porque hay diferencias muy grandes de estos resultados, según las personas.
- Una misma tarea, realizada por las mismas personas consumirá diferentes tiempos reales en entornos de trabajo diferentes.

Sobre estas premisas:

- No es posible estimar con precisión, ni el trabajo de un requisito, ni el tiempo necesario para desarrollarlo.
- La complejidad de las técnicas de estimación crece exponencialmente en la medida que:
  - Intentan incrementar la fiabilidad y precisión de los resultados.
  - Aumenta el tamaño de la tarea estimada.

La estrategia empleada por la gestión ágil es:

- No empeñarse en estimaciones precisas.
- Estimar con la técnica "juicio de expertos"
- Descomponer las tareas de los sprints en sub-tareas más pequeñas, si las estimaciones superan los rangos de las 16-20 horas de de trabajo.

Ilustración 77 - Características de las métricas ágiles

Unidades de trabajo

Las unidades para medir el trabajo pueden estar relacionadas directamente con el producto, como los tradicionales puntos de función de COCOMO; o indirectamente, a través del tiempo necesario para realizarlo.

Ilustración 78 - Unidades de trabajo

La gestión ágil suele llamar a las unidades que emplea para medir el trabajo "puntos", "puntos de funcionalidad" "puntos de historia"…
Así por ejemplo la unidad de medida "Story Point" de eXtreme Programming define: la cantidad de trabajo que se realiza en un día de trabajo ideal.

Cada organización, según sus circunstancias y su criterio institucionaliza su métrica de trabajo definiendo el nombre y las unidades.

Puede basarse en

- Estimación del tamaño relativo y emplear puntos
- Estimación del tiempo ideal necesario para realizar la tarea que se mide.

Lo importante es que la métrica empleada, su significado y la forma de aplicación sea consistente en todas las mediciones, en todos los

proyectos de la organización y conocida por todas las personas:

Que se trate de un procedimiento de trabajo institucionalizado.

Velocidad

Velocidad es la magnitud que viene determinada por la cantidad de trabajo realizada en un periodo de tiempo (Timebox) Los equipos que miden el trabajo con tiempo ideal, hablan de "Velocidad".

Decir, por ejemplo, que la velocidad del equipo en el sprint ha sido de 23, se refiere a que han completado tareas que medían en toral 23 story points.

Si en el sprint siguiente consiguen una velocidad mayor, querrá decir que han logrado programar más story points en el mismo tiempo, o lo que es lo mismo que han conseguido aumentar el porcentaje de tiempo ideal en la jornada de trabajo: han reducido los tiempos de interrupciones, distracciones o dedicados a otras tareas.

Se le puede llamar velocidad o eficiencia, lo importante no son los nombres, ni merece la pena entrar en cuestiones bizantinas.

Resumen

Tiempo real: tiempo total de trabajo, equivale a la jornada laboral

Tiempo ideal: Tiempo de trabajo en "condiciones ideales", esto es: sin ninguna interrupción, pausa, distracción, o atención a tareas ajenas a la que se tiene asignada en el sprint.

Para determinar el grado de avance de un proyecto, la gestión ágil no mide el trabajo ya realizado, sino el pendiente de realizar.

Tomando como premisas

- No es posible estimar con precisión, ni el trabajo de un requisito, ni el tiempo necesario para desarrollarlo.
- La complejidad de las técnicas de estimación crece exponencialmente en la medida que:
  - Intentan incrementar la fiabilidad y precisión de los resultados.
  - Aumenta el tamaño de la tarea estimada.

La estrategia empleada por la gestión ágil es:

- No profundiza en estimaciones precisas.
- Emplea la técnica de "juicio de expertos"
- Descompone las tareas de los sprints en sub-tareas más pequeñas, si las estimaciones superan los rangos de las 16-20 horas de de trabajo.

Velocidad y eficiencia son términos similares. Se suele preferir "velocidad" cuando las mediciones se basan en tiempo teóricos, y "eficiencia" cuando lo hacen en tiempo real.

Medición: Usos y herramientas

Gráfico de producto:

En inglés: gráfico “burn-up”.
Es una herramienta de planificación y seguimiento del propietario del producto, que muestra de un vistazo, en un gráfico muy simple el plan general de desarrollo del producto, y la traza de su evolución.

Se confecciona con:

- La estimación del esfuerzo prevista en la pila del producto.
- La velocidad del equipo.

Es un diagrama cartesiano que representa en el eje de ordenadas el trabajo estimado para desarrollar el producto, y en el de abcisas las fechas, medidas según las duraciones previstas para los sprints.

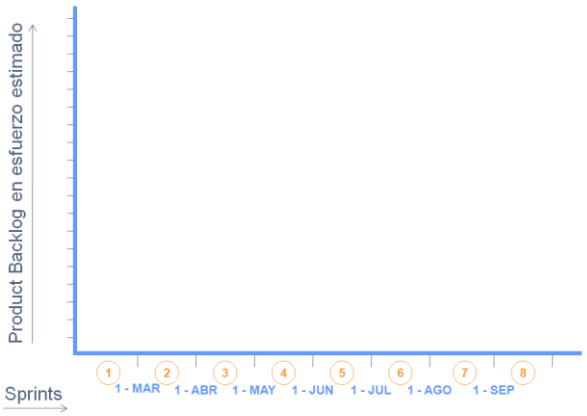


Ilustración 79 - Gráfico para planificación de producto (burn-up)

Ejemplo:

Representación del plan del producto, a partir de los temas previstos en la pila de producto (product backlog).

Convenciones empleadas por el equipo:

- Unidad para medición de trabajo: tiempo ideal
- Tiene previsto realizar sprints de duración fija mensual.
- El equipo está formado por 4 personas, y viene desarrollando una velocidad de 300 puntos por sprint.

La figura siguiente representa la situación actual del pila del producto (product backlog:[8]) el propietario del producto tiene previsto cerrar la versión 1.0 cuando disponga de los cuatro primeros temas, y su estimación inicial de trabajo para llevarlos a cabo es de 950 puntos.

Id	Temas	Trabajo	Criterio de validación	
1	Tema A 1.0	150	Lorem ipsum dolor sit amet	
2	Tema B 1.0	250	consectetuer adipiscing elit	Estimación: 950 PUNTOS
3	Tema C 1.0	250	Aliquam vehicula accumsan tortor	
4	Tema D 1.0	300	Pellentesque turpis	Versión 1.0
5	Tema A 1.1	250	Phasellus purus orci	
6	Tema D 1.1	350	penatibus et magnis dis partur	1.700 PUNTOS
7	Tema E 1.0	150	Quisque volutpat ante sit amet velit	Versión 1.1
8	Tema B 1.1	500	Cras iaculis pede eu tellus	
9	Tema C 1.1	150	Vestibulum vel diam sed pede	2.550 PUNTOS
10	Tema E 1.1	200	Suspendisse aliquam felis et turpis	Versión 1.2
11	Tema F 1.0	TBD	Nullam imperdiet lorem vitae justo	
12	Tema A 1.2	TBD	Suspendisse potenti. In nec nunc	
13	Tema B 1.2	TBD	Nam eros tellus, facilisis sed, pretium	
14	Tema F 1.1	TBD	Morbi arcu tellus, condimentum	

Ilustración 80 –Plan de versiones del producto

La versión 1.1 incluirá 3 temas más que, según la estimación inicial, supondrán unos 750 puntos de trabajo.

Y están también trazados los temas con los que piensa cerrar la versión 1.2, que se prevén con 850 puntos más de trabajo.

Para representar el plan del producto con un “Burn-Up”, se representan, con los fondos de escala apropiados:

Eje X = Fechas de los sprints previstos
Eje Y = Puntos de trabajo

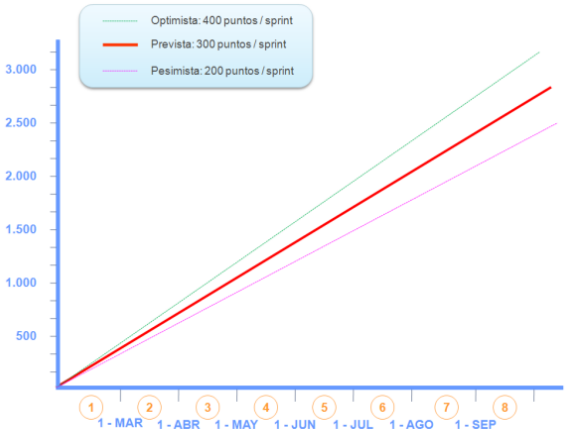


Ilustración 81 - Gráfico para planificación de producto (burn-up)

A continuación se traza en el gráfico la línea de velocidad prevista.

Siguiendo con el ejemplo, la línea roja de la imagen representa la velocidad de 300 puntos por sprint.

También puede resultar útil esbozar una estimación optimista y otra pesimista para tener la visión de una holgura de fechas aceptable.

[8] Las listas de producto y de versión evolucionan de forma continua durante la vida del producto.

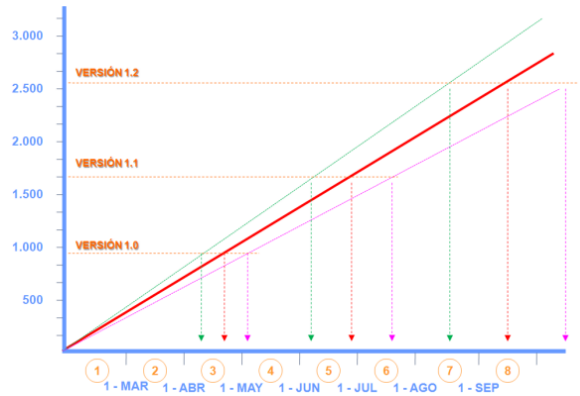


Ilustración 82 - Gráfico para planificación de producto (burn-up)

La línea de velocidad proyecta sobre el eje X la fecha o sprint en el que previsiblemente se completarán las versiones representadas en el eje Y.

Gráfico de avance: monitorización del sprint

También se suele emplear la denominación inglesa: gráfico "burn-down".

Es el gráfico que actualiza el equipo en las reuniones de seguimiento del sprint, para comprobar el ritmo de avance, y detectar desde el primer momento si es el previsto, o se puede ver comprometida la entrega prevista al final de sprint.

La estrategia ágil para el seguimiento de los proyectos se basa en:

- Medir el esfuerzo que falta, no el realizado.
- Seguimiento muy cercano (diario de ser posible).

Y en este gráfico toman forma los dos principios:

- En el eje Y se registra el trabajo que aún falta por realizar.
- Se actualiza a diario.

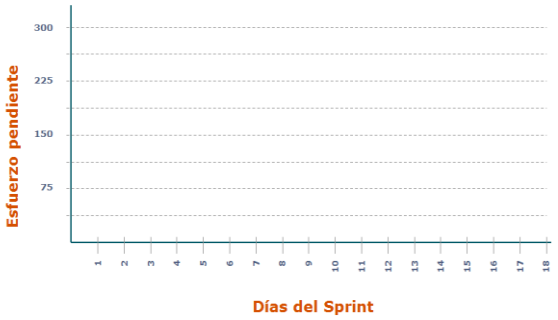


Ilustración 83 - Gráfico de avance del sprint (burn-down)

El equipo dispone en la pila del sprint, de la lista de tareas que va a realizar, y en cada uno figura el esfuerzo pendiente.
Esto es: el primer día, en la pila de tareas figura para cada tarea el esfuerzo que se ha estimado, puesto que aún no se ha trabajado en ninguna de ellas.

Día a día, cada miembro del equipo actualiza en la pila del sprint el tiempo que le queda a las tareas que va desarrollando, hasta que se terminan y van quedando en 0 los tiempos pendientes.
La figura siguiente muestra un ejemplo de pila en el sexto día del sprint: las tareas terminadas ya no tienen esfuerzo pendiente, y del esfuerzo total previsto para el sprint: 276 puntos (A), en el momento actual quedan 110 (B).

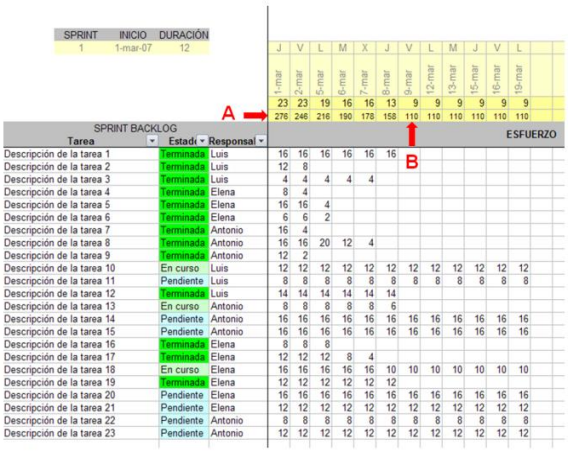


Ilustración 84 - Pila del sprint

Con esta información de la pila del sprint se actualiza el gráfico poniendo cada día el esfuerzo pendiente total de todas las tareas que aún no se han terminado.

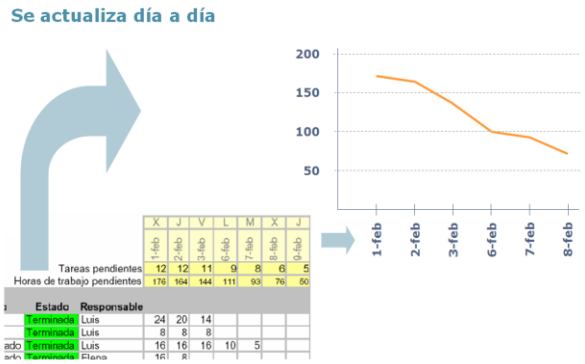


Ilustración 85 - Representación de los tiempos pendientes en el gráfico burn-down

El avance ideal de un sprint estaría representado por la diagonal que reduce el esfuerzo pendiente de forma continua y gradual hasta terminarlo en el último día del sprint:

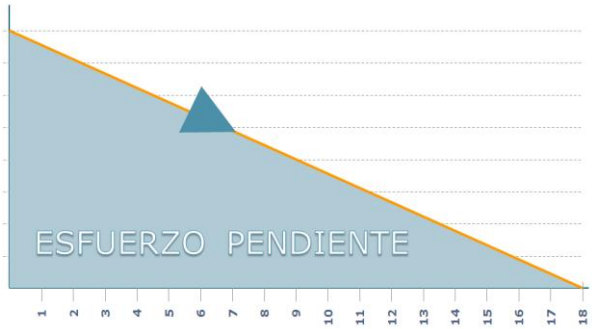


Ilustración 86 - El gráfico burn-down muestra el esfuerzo pendiente

Las gráficas de diagonal perfecta no son lo habitual, y la siguiente imagen es un ejemplo de un patrón de avance más normal.

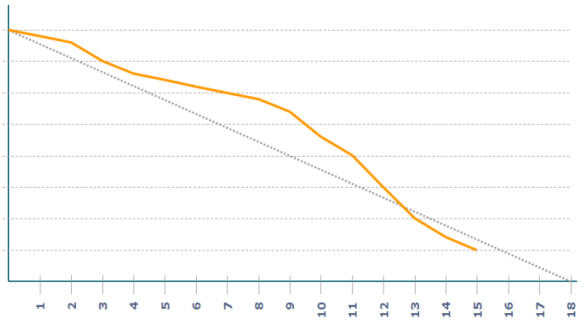


Ilustración 87 - Ejemplo de gráfico burn-down

El siguiente sería el aspecto de la gráfica en un "sprint sub-estimado"

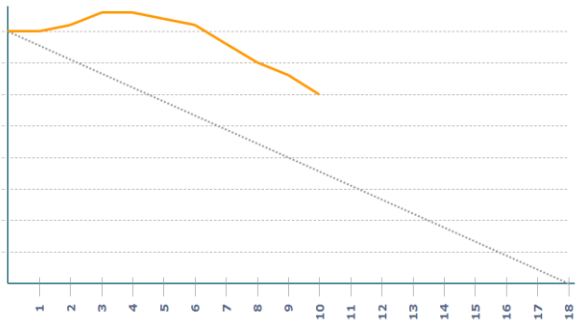


Ilustración 88 - Gráfico burn-down de un sprint subestimado

La estimación que realizó el equipo en la reunión de inicio del sprint es inferior al esfuerzo real que están requiriendo las tareas.

Y el siguiente sería el patrón de gráfica de un "sprint sobre-estimado".

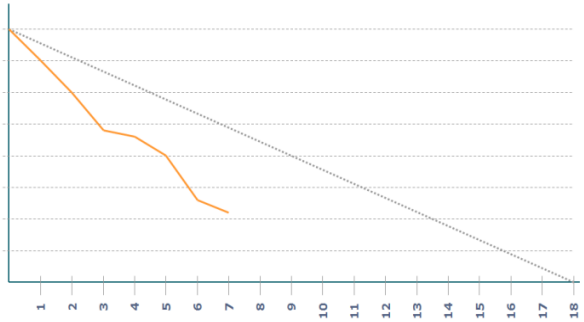


Ilustración 89 - Gráfico burn-down de un sprint sobre-estimado

Estimación de póquer

Es una práctica ágil, útil para conducir las reuniones en las que se estima el esfuerzo y la duración de tareas.

Para evitar en estos casos las discusiones dilatadas que no terminan de dar conclusiones concretas, James Grening ideó este juego de planificación como ayuda para conducir la reunión.

El modelo inicial de Grening consta de 8 cartas, con los números representados en siguiente figura, porque James lo ideó para las estimaciones de versión en eXtreme Programming, con equipos que emplean como unidad de esfuerzo: días de trabajo de cada par de programadores[9] y trabajan con tareas de tamaño máximo de 10 días.[10]

[9] eXtreme Programming trabaja con programación por parejas.
[10] Las tareas de mayor tamaño se descomponen en subtareas hasta que éstas tienen estimaciones máximas de 10 días.

Ilustración 90 - Ejemplo de cartas para estimación de póquer

El funcionamiento es muy simple: cada participante dispone de un juego de cartas, y en la estimación de cada tarea, todos vuelven boca arriba la combinación que suma el esfuerzo estimado.

Cuando se considera que éste es mayor de 10 horas (o del tamaño máximo considerado por el equipo para una tarea), se levanta la carta "∞"

Cada equipo u organización puede utilizar un juego de cartas con las numeraciones adecuadas a la unidad de esfuerzo con la que trabajan, y el tamaño máximo de tarea que se va a estimar.

Lo relevante no es el número de cartas, la unidad de medida empleada, o si el tamaño máximo de tarea debe ser 5, 7 ó 10 días, sino trabajar con el modelo que el equipo considera más apropiado, respetando los principios siguientes:

- No definir tareas demasiado grandes, porque entorpece la precisión de las estimaciones y la identificación de riesgos.
- No definir tareas cuya duración (en puntos o tiempo) resulte inferior a medio día ideal de trabajo.
- Utilizar al misma unidad de medida (story points, días, horas…) en todos los gráficos y pilas.

Variante: sucesión de Fibonacci

Basado en el hecho de que al aumentar el tamaño de las tareas, aumenta también el margen de error, se ha desarrollado una variante que consiste en emplear sólo números de la sucesión de Fibonacci para realizar las estimaciones, de forma que:

- El juego de cartas está compuesto por números de la sucesión de Fibonacci.
- La estimación no se realiza levantando varias cartas para componer la cifra exacta, sino poniendo boca arriba la carta con la cifra más aproximada a la estimación.

Para estimar tareas puede ser válido un juego de cartas como éste:

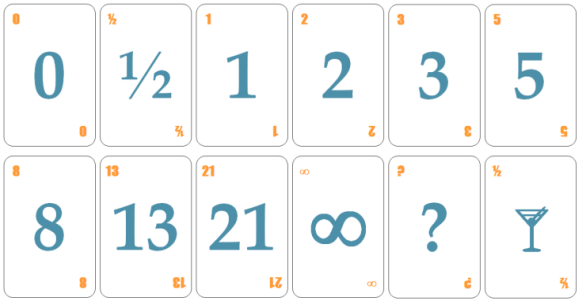


Ilustración 91 - Ejemplo de cartas para estimación de póquer con secuencia Fibonacci

Si se quiere emplear la planificación de póquer para estimar requisitos a nivel de producto o de versión (funcionalidades, temas…) además de usarlo al nivel de tareas de sprint, se pueden añadir cartas al juego para permitir estimaciones de mayor tamaño (34, 55, 89, 144…)

Es frecuente emplear una carta con un símbolo de duda o interrogación para indicar que, por las razones que sean, no se puede precisar una estimación.
También es posible incluir otra carta con alguna imagen alusiva, para indicar que se necesita un descanso.

Funcionamiento

- Cada participante de la reunión tiene un juego de cartas.
- Para cada tarea (historia de usuario o funcionalidad, según sea el nivel de requisitos que se va a estimar) el cliente, moderador o propietario del producto expone la descripción empleando un tiempo máximo.
- Hay establecido otro tiempo para que el cliente o propietario del producto atienda a las posibles preguntas del equipo.
- Cada participarte selecciona la carta, o cartas que representan su estimación, y las separa del resto, boca abajo.
- Cuando todos han hecho su selección, se muestran boca arriba.
- Si la estimación resulta "infinito", por sobrepasar el límite máximo establecido, la tarea debe dividirse en sub-tareas de menor tamaño.
- Si las estimaciones resultan muy dispares, quien asume la responsabilidad de gestionar

la reunión, con su criterio de gestión, y basándose en las características del proyecto, equipo, reunión, nº de elementos pendientes de evaluar, puede optar por:

- Preguntar a las personas de las estimaciones extremas: ¿Por qué crees que es necesario tanto tiempo?, y ¿por qué crees que es necesario tan poco tiempo? Tras escuchar las razones, repetir la estimación.
- Dejar a un lado la estimación de esa tarea y retomar al final o en otro momento aquellas que hayan quedado pendientes.
- Pedir al cliente o propietario del producto que descomponga la funcionalidad y valorar cada una de las funcionalidades resultantes.
- Tomar la estimación menor, mayor, o la media.

Este protocolo de reunión evita los atascos de análisis circulares en ping-pong entre diversas opciones de implementación, hace participar a todos los asistentes, reduce el cuarto de hora o la media hora de tiempo de estimación de una funcionalidad, a escasos minutos, consigue alcanzar consensos sin discusiones, y además resulta divertido y dinamiza la reunión.

Resumen

El **gráfico de producto o burn-up,** muestra en un vistazo el plan general de desarrollo del producto.

A partir de la velocidad del equipo y las estimaciones de trabajo previstas en la pila del producto, representa las fechas o sprints en los que previsiblemente se irán terminando las diferentes versiones.

El **gráfico de avance o burn-down** es una herramienta ágil que monitoriza el ritmo de trabajo (normalmente de un sprint).

En el eje vertical de un diagrama cartesiano representa el trabajo pendiente a lo largo del tiempo del sprint (eje horizontal).

Las desviaciones sobre, o bajo la línea diagonal que representaría el avance ideal del sprint alertan de forma temprana de desviaciones sobre el ritmo de desarrollo previsto.

La **estimación de póquer** es una práctica ágil para reducir las dificultades habituales en las reuniones de trabajo para planificación por juicio de expertos.

En ella los participantes emplean un juego de cartas para concretar las unidades de esfuerzo que estiman para cada tarea.

Hay dos variantes:
Natural: los participantes pueden estimar el esfuerzo con la cifra exacta que crean más adecuada.
Fibonacci: las estimaciones solo se pueden realizar empleando números de la sucesión de Fibonacci.

En cada caso, el juego de cartas empleado tiene la numeración apropiada.

Kanban

Introducción

¿Gestión visual?

En una calle la organización es compleja: circulación de coches, peatones, zonas de aparcamiento, carriles para autobuses, etc. pero la señalética nos indica todo. Estamos acostumbrados a la "gestión visual."

La industria de la automoción es posiblemente la primera que incorporó la "gestión visual" en el propio escenario de producción: todo está a la vista, la identificación de los sectores, los productos son reconocibles y están etiquetados indicando su fase dentro del proceso de la cadena.
En todo momento se sabe qué se está produciendo. La información está en el mismo entorno de trabajo, visible, expresada con simplicidad, y no en ordenadores o libros de registro de procesos.

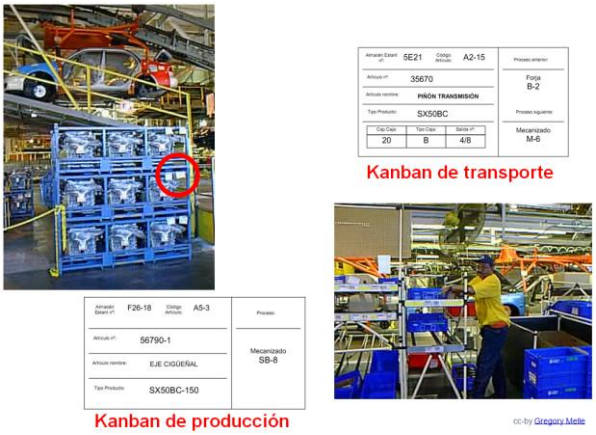


Ilustración 92 - La industria del automóvil fue la primera en incorporar gestión visual

Kanban

Kanban es una técnica de control visual. Es un sistema de señalización para desencadenar acciones.

El término japonés Kanban se puede traducir por tablero, o tarjeta de señalización, haciendo referencia a un sistema de señalización empleado en los procesos de producción para coordinar que cada parte se entrega al departamento o sección correspondiente sólo cuando se necesita, reduciendo al mismo tiempo la sobreproducción o almacenamientos innecesarios.

El origen de este sistema de control visual se remonta a finales de los cuarenta o principios de los cincuenta, cuando su autor, Taiichi Onho desarrollaba señales para implementar con gestión visual métodos de producción "Just In Time" (JIT) en los centros de producción de Toyota de Japón.

Los dos pilares del sistema de producción de Toyota son "just-in-time" y "automatización humanizada" o autonomatización (autonomation). La herramienta que utilizamos para operar con el sistema es kanban.

Taiichi Ohno's

Ilustración 93 - Origen de kanban como sistema de control visual

Estas prácticas no tuvieron demasiada difusión, más allá de los centros en los que se aplicaban, hasta las fechas de recesión en los 70, cuando se empezó a popularizar el uso de "kanban's", para minimizar el trabajo en el proceso (WIP)[11] y reducir los costes asociados al almacenamiento de inventarios.

> ***Kanban es un sistema de señalización para comunicar información relativa y necesaria en la ejecución o monitorización de un trabajo.***

Originalmente Toyota empezó a usar kanban para reducir costos y gestionar el uso de las máquinas, pero hoy Toyota continúa usando el sistema no sólo para gestionar costes y flujos, sino también para identificar impedimentos en el flujo de producción y analizar los puntos de mejora de forma continua.

Kanban también es herramienta útil para la gestión ágil.

El desarrollo ágil de software emplea prácticas de gestión visual, por ser las que mejor sirven a los principios de comunicación directa y simplicidad en la documentación y gestión.

[11] "Work in process" (trabajo en el proceso) también "in-process inventory" (inventario en el proceso) o elementos del producto, que aún no están terminados y se encuentran en el proceso de producción. No confundir con Work in progress (trabajo en progreso) término que designa un trabajo que ha comenzado pero aún no está terminado.

Desde 2005 es cada vez más frecuente reemplazar los formatos de lista para las pilas de producto y de sprint, por notas adhesivas, porque que resultan muy versátiles al poder cambiar su posición: para re-ordenar las prioridades de las historias de una pila de producto, e indicar sólo con su posición qué tareas o historias se están programando, o probando, o ya se han terminado, etc.

La generalización del uso de kanban como herramienta en equipos de programación, puede inducir el error de considerar a kanban como método, práctica o modelo de gestión ágil surgido en equipos de programación.

Antes de ver algunos ejemplos de las posibilidades de kanban, conviene resaltar que no se trata de un modelo ágil, sino de una herramienta, o técnica para comunicar información relativa y necesaria en la ejecución o control de un trabajo. Taiichi Onho usó información kanban como herramienta para producción. Pero el modelo de producción era just-in-time" + autonomatización, no kanban.

Emplear etiquetas para registrar epics, historias o tareas, y diferentes marcas o su posición en el tablero para representar su estado de desarrollo.

Ilustración 94 - Tablero kanban

Habría dos suposiciones erróneas, al considerar a kanban como un modelo de gestión ágil, que despistarían y limitarían sus posibilidades:

- ¿Cómo es el modelo de gestión Kanban?
- ¿Cuál es el formato que debe tener un tablero Kanban?

Kanban no es un modelo o marco de gestión, sino una herramienta de señalización.
No hay por tanto un formato cerrado de tarjetas o tableros kanban.

Intentar acotar la forma de usar tarjetas con información kanban, o cómo gestionar las historias o tareas si se emplean tarjetas kanban, restringe las posibilidades de la herramienta, y va en contra de un criterio de gestión flexible para adaptar las técnicas a las características de la organización.

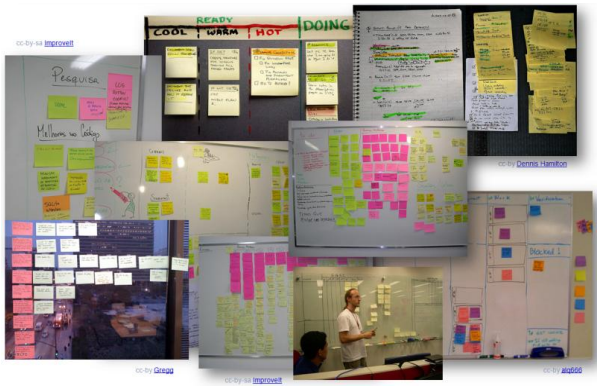


Ilustración 95 - Ejemplos de tableros kanban

Fortalezas de Kanban

Un sistema visual kanban puede emplearse para facilitar información, trazar la gestión de un procedimiento, controlar su ejecución, o una combinación de ellas.

INFORMACIÓN – CONTROL – GESTIÓN VISUAL

Ilustración 96 - Información y gestión visual

Información

Los principios ágiles a los que se puede dar un soporte adecuado empleando kanban en su dimensión de medio de información son:

Favorece la comunicación directa

Facilita la comunicación directa del equipo al actualizar la información en reuniones enfrente de un tablero kanban.

Comparte la visibilidad de la evolución del proyecto con todos los implicados.

Detección temprana de problemas

La información kanban ofrece un seguimiento próximo del proyecto: la actualización de la información just-in-time, ayuda a identificar en un primer momento los posibles impedimentos problemas y riesgos, que de otra forma suelen pasar desapercibidos hasta que empiezan a producir retrasos o repercusiones ya inevitables.

Favorece una cultura de colaboración y resolución.

Es un medio de comunicación abierto y transparente para el equipo y todos los participantes.

Gestión / Control

Kanban como herramienta para la gestión del proyecto ofreciendo, además de información, pautas de flujo y control aporta:

Monitorización y regulación del flujo, y la carga de trabajo

La posición de cada tarjeta sobre el tablero refleja el estado en el que se encuentra la correspondiente historia de usuario.
Los estados básicos que se suelen representar en un tablero kanban son “pendiente”, “en curso” y “terminada”.

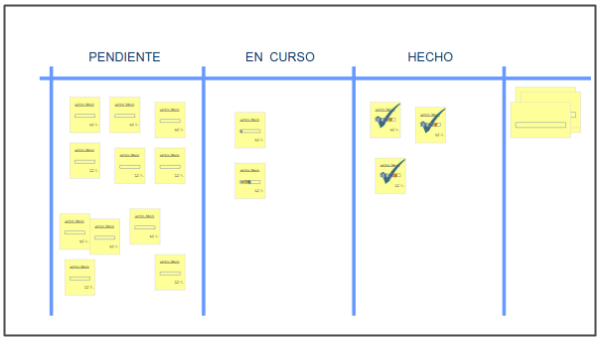


Ilustración 97 - Tablero kanban típico: pendiente - en curso - hecho

En algunos casos puede resultar conveniente marcar sub-estados (testing, validada…).

La ordenación de las tareas desde el área “pendiente”, indica ya en el inicio la secuencia de las tareas según sus prioridades.

Al emplear kanban como herramienta de gestión o control, se suele establecer el parámetro “WIP” (Work In Process”) para delimitar el número máximo de tareas o de historias que pueden estar de forma simultánea en el estado “en curso”.

Genera un flujo de trabajo que lleva los problemas a la superficie

Los conflictos en la priorización o volumen de trabajo, las incidencias o impedimentos en el desarrollo se ponen de manifiesto de forma inmediata en el seguimiento diario del flujo de la pizarra.

Produce un ritmo sostenido y evita la ley de Parkinson

Genera un flujo continuo de trabajo cuyo ritmo no está marcado por una planificación temporal: Gantt o Sprint.
Los ritmos temporales tienden a generar ciclos arritmicos de trabajo favoreciendo la procrastinación al principio del plan o ciclo y presión al final del plazo.

> ***El trabajo se expande hasta llenar el tiempo disponible para que se termine.***
>
> ***Ley de Parkinson***

> ***Los procesos Ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios debemos ser capaces de mantener un ritmo constante de forma indefinida***
>
> ***Manifiesto Ágil***

Produce desarrollo incremental

Y por supuesto responde al principio ágil imprescindible: desarrollo incremental.

Tareas que se completan con un ritmo sostenido y de forma continua, y con ellas historias de usuario que se van sumando al producto, proporcionándole un incremento continuo de valor según las prioridades de negocio marcadas por el propietario del producto.

Ejemplos de uso

Información:

Dos ejemplos que muestran posible usos de tableros kanban para monitorizar el progreso del trabajo, pero sólo mostrando la información de forma simple, visual y abierta, sin aportar pautas o criterios de gestión.

No emplean el formato más habitual de distribución horizontal en orden: "Pendiente" -> "En curso" -> "Terminado"; sino que agrupan en vertical la información del desarrollo de cada epic o historia de usuario.
Pueden ser útiles como tableros de información para gestores de producto.

A nivel de epics / historias

Este ejemplo representa un posible tablero kanban en el puesto de trabajo del o de los responsables del producto, que podría ser el product manager, o el departamento de marketing, la oficina de proyectos, etc. según la organización de la empresa y de productos que desarrolle.

El tablero informa del estado de desarrollo del producto a nivel de epics[12] y las correspondientes historias de usuario.

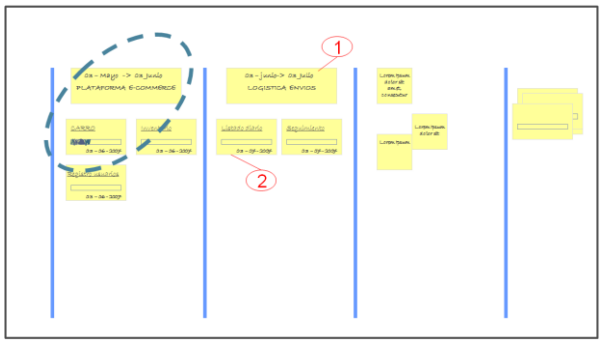


Ilustración 98 - Ejemplo de tablero kanban para seguimiento de producto a nivel de epics e historias

Cada columna representa un "epic" o gran historia del producto, que está descrita en la tarjeta de la parte superior (1)

Este tablero puede ser útil para un gestor de producto, que podría indicar en la tarjeta del epic información adicional como la versión o *release* de producto en la que se integrará, o fecha en la en la que se espera incluir, o esfuerzo estimado para su desarrollo, etc.

Debajo de la tarjeta con el epic, de cada columna se despliegan las historias de usuario (2) que lo componen.

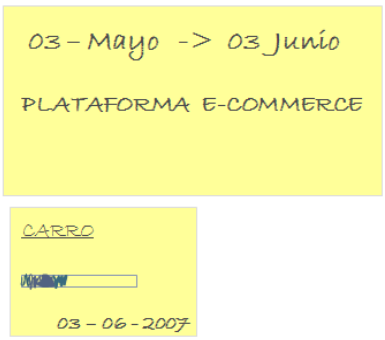


Ilustración 99 - Ejemplo de tarjetas kanban

En este ejemplo la tarjeta de cada historia de usuario refleja el avance que la misma está teniendo en la pizarra del equipo, a través de una barra de una barra de progreso que actualiza a diario el propietario de producto, pintando sobre la tarjeta de la historia.

A nivel de historias / tareas

Con un esquema similar al anterior, el tablero de este ejemplo muestra información visual de las historias de usuario que se están desarrollando, las tareas que componen cada una, y el grado de avance.

[12] Grandes historias o funcionalidades generales.

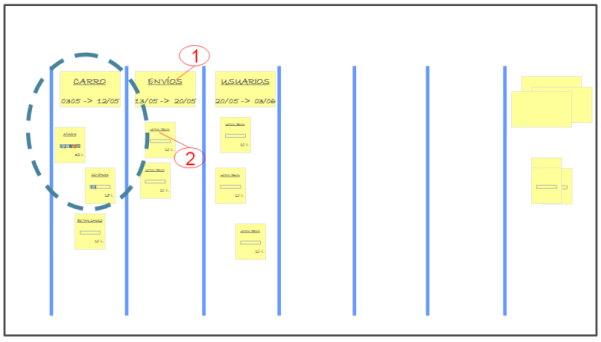


Ilustración 100 - Ejemplo de tablero kanban para seguimiento de producto a nivel de historias y tareas

Las historias de usuario y su descomposición en tareas, están presentados en columnas.
En la cabecera (1) está situada la tarjeta con la descripción e información de estimación o fechas previstas según la velocidad del equipo, y a lo largo de la columna, las diferentes historias.

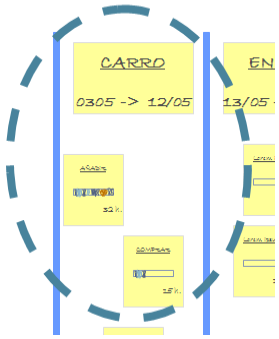


Ilustración 101 - Ejemplo de tablero para seguimiento de producto: detalle

En este ejemplo, las tarjetas que representan las tareas, incluyen una "barra de estado" que representa cuánto falta para terminar.

Información + autogestión

Formato básico

Este es el formato más habitual para ayudar a los equipos a auto-gestionar un flujo de trabajo continuo a través del avance de las tareas por tres estados básicos: pendiente, en curso y terminado.

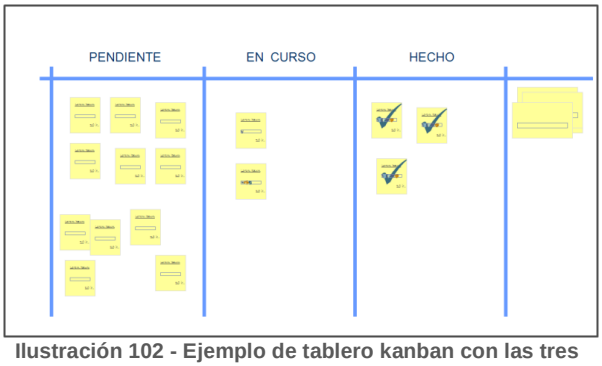


Ilustración 102 - Ejemplo de tablero kanban con las tres áreas típicas: pendiente - en curso - hecho

Según las necesidades o preferencias de cada equipo, se suelen ampliar o detallar estos estados básicos, con indicaciones relativas a si se ha preparado ya la prueba unitaria, si se ha realizado, o si la tarea pendiente está en la pila o ya está preparada y estimada, etc.

WIP -Work In Process (Trabajo en el proceso)

Al emplear un sistema de monitorización kanban como herramienta de gestión o control del ritmo de avance desaparece el concepto de sprint.

El **incremento** ya no es el resultado de un sprint, sino cada historia que se termina. Par lograr un flujo continuo de funcionalidades que, una a una van aportando incrementos de forma sostenida, es necesario limitar la cantidad de trabajo que hay en proceso. Limitar el número de tareas que pueden estar de forma simultánea en los diferentes estados de desarrollo.
De esta forma se evita que se amontonen tareas obligando a los miembros del equipo a trabajar en varias a la vez, perdiendo el foco, el ritmo y la eficiencia.

Al parámetro que indica el número máximo de tareas en un área del tablero kanban se le denomina WIP: Work In Process.

Un valor "WIP" demasiado bajo produce tiempos muertos, y demasiado alto, cuellos de botella.

La experiencia ayuda al equipo a ir ajustándolo para lograr un flujo continuo, o lo más continuo posible.

Si no se cuenta con experiencia previa, y considerando que las tareas no deberían tener tamaños mayores de 4 horas ideales, el equipo debe establecer un criterio de inicio, y a partir de él ir ajustando.

En este sentido una recomendación generalmente útil es empezar con un WIP igual al nº de miembros del equipo x 1.5, redondeando el resultado por exceso, o x2.

En ningún caso es aconsejable trabajar con tareas de tamaño que se prevea superior a un día de trabajo, y si esto ocurre lo aconsejable es dividirlas en otras de menor tamaño.

Formato básico con control del trabajo en el poceso (WIP)

La figura siguiente presenta una implementación kanban con límite de trabajo en los estados "Producto analizado" y "En curso".

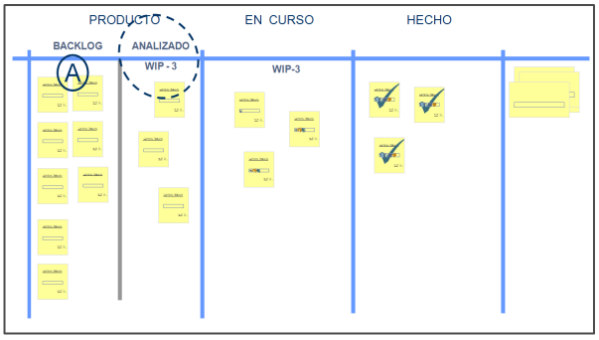


Ilustración 103 - Ejemplo de tablero kanban con indicación del WIP de algunas áreas

En este ejemplo, el propietario de producto tiene una zona para registrar y ordenar de forma priorizada el backlog. (A).
Es el área en la que el responsable de producto añade, modifica, y reordena la prioridad de cada historia de forma continua.

Pero sólo son 3 las historias que pueden estar en estado "decididas" para pasar a producción. 3 con la que ya está previsto analizar y revisar la estimación con el equipo.

Ilustración 104 - Ejemplo de tablero kanban con indicación del WIP de algunas áreas: detalle

De igual forma, el área "en curso" tiene un límite de 3 historias.

Hasta que una no pasa a "HECHO", no puede entrar ninguna a producción, y de igual forma mientras haya 3 en la zona "ANALIZADO" no se decide cuál será la próxima historia del backlog.

De esta forma se crea un flujo de trabajo sin cuellos de botella, continuo y enfocado.

Kanban Box

Una práctica diseñada para responder a las dificultades frecuentes al gestionar tareas de varios proyectos en el mismo departamento de producción es una implementación Kanban diseñada en Scrum Manager que hemos denominado Kanban Box.

La configuración es la siguiente:

La organización mantiene una "pila de producción" o lista de historias de usuario, pendientes, estimadas y priorizadas.

Si la organización trabaja en un único producto, la "pila de producción" es en definitiva la pila del producto o product backlog.

Si lleva a cabo el desarrollo o mantenimiento simultáneo de varios sistemas, la pila de producción es gestionada por los propietarios de producto, o la oficina de proyectos... en definitiva quienes sean responsables según la estructura de la organización.

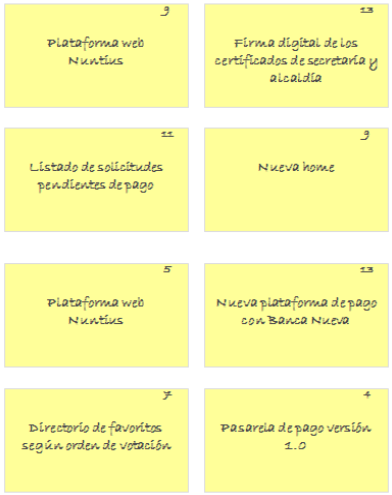


Ilustración 105 - Tarjetas con historias de usuario

En la pila de producción las tareas están estimadas y ordenadas según los criterios de prioridad compartidos entre los intereses de los diferentes proyectos y de la organización en conjunto.

El equipo que va a hacerse cargo de una historia, la descompone en tareas que representa en una "caja kanban":

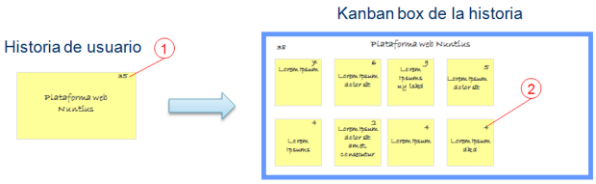


Ilustración 106 - Descomposición de una historia de usuario en una kanban box

(1) Estimación temprana de la historia, realizada por "juicio de experto" por el propietario del producto.
(2) Estimación de la tarea realizada por el equipo al descomponer la historia en tareas.

En cada caja se representa:

(1) Puntos totales de esfuerzo estimados
(2) Tarjetas con las tareas
(3) Un fondo de escala graduado con los puntos totales de esfuerzo estimados
(4) Barra dibujada con rotulador "borrable" que representa la velocidad prevista
(5) Barra de velocidad real
(6) Una descripción de la historia de usuario.

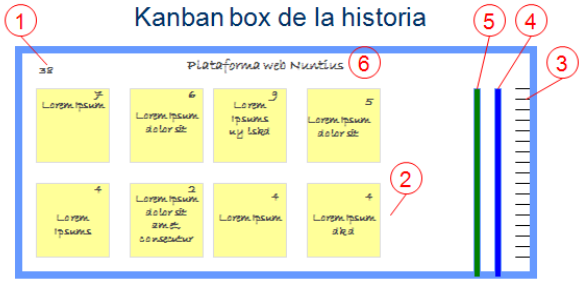


Ilustración 107 - Ejemplo de kanban box

De esta forma se van "encajando" las historias de usuario, o preparando para pasar a producción.

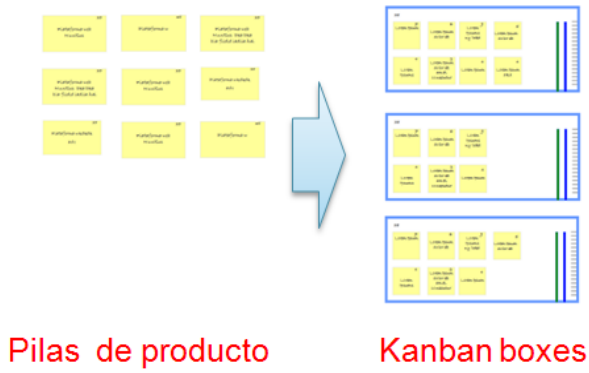


Ilustración 108 - Descomposición de historias de usuario en kanban boxes

Las cajas preparadas van entrando en los "slots" disponibles en la columna "pendiente" del tablero general de la organización.

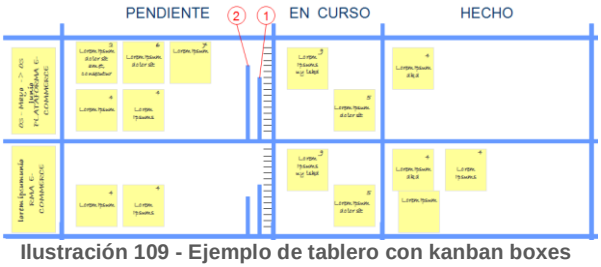


Ilustración 109 - Ejemplo de tablero con kanban boxes

A diario, cada equipo realiza la reunión de seguimiento o reunión de pie frente al tablero, actualizando el estado de cada tarea (pendiente -> en curso -> hecho), y la actualización de las barras de velocidad:

La barra de velocidad prevista (1) se actualiza todos los días con el ritmo de velocidad previsto según la velocidad media de la organización, y el nº de miembros del equipo. Si por ejemplo se trata de un equipo de 3 personas, y la velocidad media es de 3 puntos por persona/día, cada día la barra de velocidad prevista disminuye e 9 puntos.

La barra de velocidad real (2) representa la suma del esfuerzo de las tareas que aún se encuentran en estado "pendiente" y "en curso".

La diferencia de altura entre las barras de velocidad muestra desviaciones del esfuerzo previsto, en uno u otro sentido, de forma visual y rápida., de forma similar a un gráfico burn-down.

El tablero general de una organización o departamento de producción con cuatro equipos presentaría un aspecto similar a la figura siguiente.

La columna “Pendiente” contiene cuatro “slots” en los que van entrando las cajas kanban, a medida que se van terminando las anteriores.

Ilustración 110 - Ejemplo de tablero con kanban boxes

Resumen

La gestión visual introduce en el propio escenario de trabajo información con instrucciones para su ejecución o información sobre su estado.

Kanban es un sistema de señalización para comunicar información relativa y necesaria en la ejecución o monitorización de un trabajo.

El origen del sistema de señalización Kanban se remonta a finales de los cuarenta, en los centros de producción de Toyota de Japón.

A partir de 2005 surgen prácticas de programación ágil que muestran la información de control o monitorización de los proyectos con técnica Kanban, empleando tarjetas adhesivas sobre tableros.

Kanban no es un modelo o marco de gestión, sino una herramienta de señalización. No hay por tanto un formato cerrado de tarjetas o tableros kanban.

La señalización kanban se puede emplear para:

La gestión de un proyecto puede utilizar señalización kanban para:

- Ofrecer información de seguimiento del proyecto.
- Herramienta de gestión y control del ritmo del proyecto.

Información

Como herramienta de información, la señalización kanban da soporte a los siguientes principios ágiles:

- Comunicación directa.
- Detección temprana de problemas.
- Cultura de colaboración y resolución.

Gestión / control

Como herramienta de gestión aporta:

- Monitorización y regulación del flujo y carga de trabajo.
- Genera un flujo de trabajo que lleva los problemas a la superficie
- Produce un ritmo de trabajo sostenido y evita la ley de Parkinson.

WIP – Work In Process (Trabajo en el proceso)

WIP es el parámetro que determina el número máximo de tareas que se pueden encontrar en el estado “en curso” de forma simultánea.

Un valor WIP demasiado bajo produce tiempos muertos, y demasiado alto, cuellos de botella.

Lista de ilustraciones

Lista de ilustraciones

Trabajos citados

Trabajos citados

Ambler, S. W. (s.f.). *The Agile Unified Process (AUP)*. Obtenido de http://www.ambysoft.com/unifiedprocess/agileUP.html

Bauer, F., Bolliet, L., & Helms, H. (1968). Software Engineering. *Nato Software Engineering Conference 1968* (pág. 136). Garmisch: Peter Naur and Brian Randell.

Bell, T., & Thayer, T. (1976). Software requirements: Are they really a problem? *ICSE '76 Proceedings of the 2nd international conference on Software engineering* .

Boehm, B., & Turner, R. (2003). *Balancing Agility and Discipline.* Boston: Addison-Wesley.

Böhm, C., & Jacopini, G. (1966). Flow Diagrams, Turing Machines and Languages with Only Two Formation Rules. *Communications of the ACM* , 366-371.

Cockburn, A. (2004). *Crystal Clear : A Human-Powered Methodology for Small Teams.* Addison-Wesley Professional.

Dijkstra, E. (1968). Go To Statement Considered Harmful. *Communications of the ACM* , 147-148.

Gilb, T. (1976). *Software Metrics.* Bromley.

Highsmith, J. (2000). *Adaptive Software Development: A Collaborative Approach to Managing Complex Systems.* New York: Dorset House.

Imai, K., & Takeuchi, H. (1985). Managing the New Product Development Process: How Japanese Companies Learn and Unlearn. *The Uneasy Alliance: Managing the Productivility-Technology Dilema* .

Iverson, K. E. (1962). *A Programming Language.* New York: John Wiley and Sons.

Juran, J. (1951). *Quality Control Handbook.* New York: McGraw-Hill.

Laporte, C., & April, A. (2005). Applying Software Engineering Standards in Small Settings: Recent Historical Perspectives and Initial Achievements. *CMU/SEI-2006-Special Report-001, January 2006* , 39, 51.

Nonaka, I., & Takeuchi, I. (2004). *Hitotsubashi on knowledge management.* Japan: John Wiley & Sons.

Norden, P. (1958). Curve Fitting for a Model of Applied Research and Development Scheduling. *http://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=5288520* , 232-248.

Orr., K. (2003). CMM versus Agile Development: Religious wars and software development. *Cutter Consortium, Executive Reports 3(7)* .

Paulk, M., B., C., Chrissis, M., & Weber, C. (1996). *Capability Maturity Model, version 1.1.* Pittsburg, PA: Software Eng. Inst., Carnegie Mellon Univ.

Schwaber, K. (1995). *SCRUM Development Process.* Burlington: OOPSLA 95.

Software Engineering Coordinating Comittee. (2000). *Guide to the Software Engineering Body of Knowledge.* Montreal: IEEE Computer Society.

Stevens, W., Myers, G., & Constantine, L. (1974). Structured Design. *IBM Systems Journal* , 115-139.

Sutherland, J., Viktorov, A., & Blount, J. (2006). Adaptive Engineering of Large Software Projects with Distributed/Outsourced Teams. *International Conference on Complex Systems* .

Takeuchi, H., & Nonaka. (1986). The new new product development game. *Harvard Business* , 137-146.

Turner, R., & Jain, A. (2002). Agile Meets CMMI: Culture Clash or Common Cause? *XP/Agile Universe 2002* , 153-165.

Wujec, T., & Muscat, S. (2002). *Return on Imagination: Realizing the Power of Ideas.* Return on Imagination: Realizing the Power of Ideas.

Índice

Índice