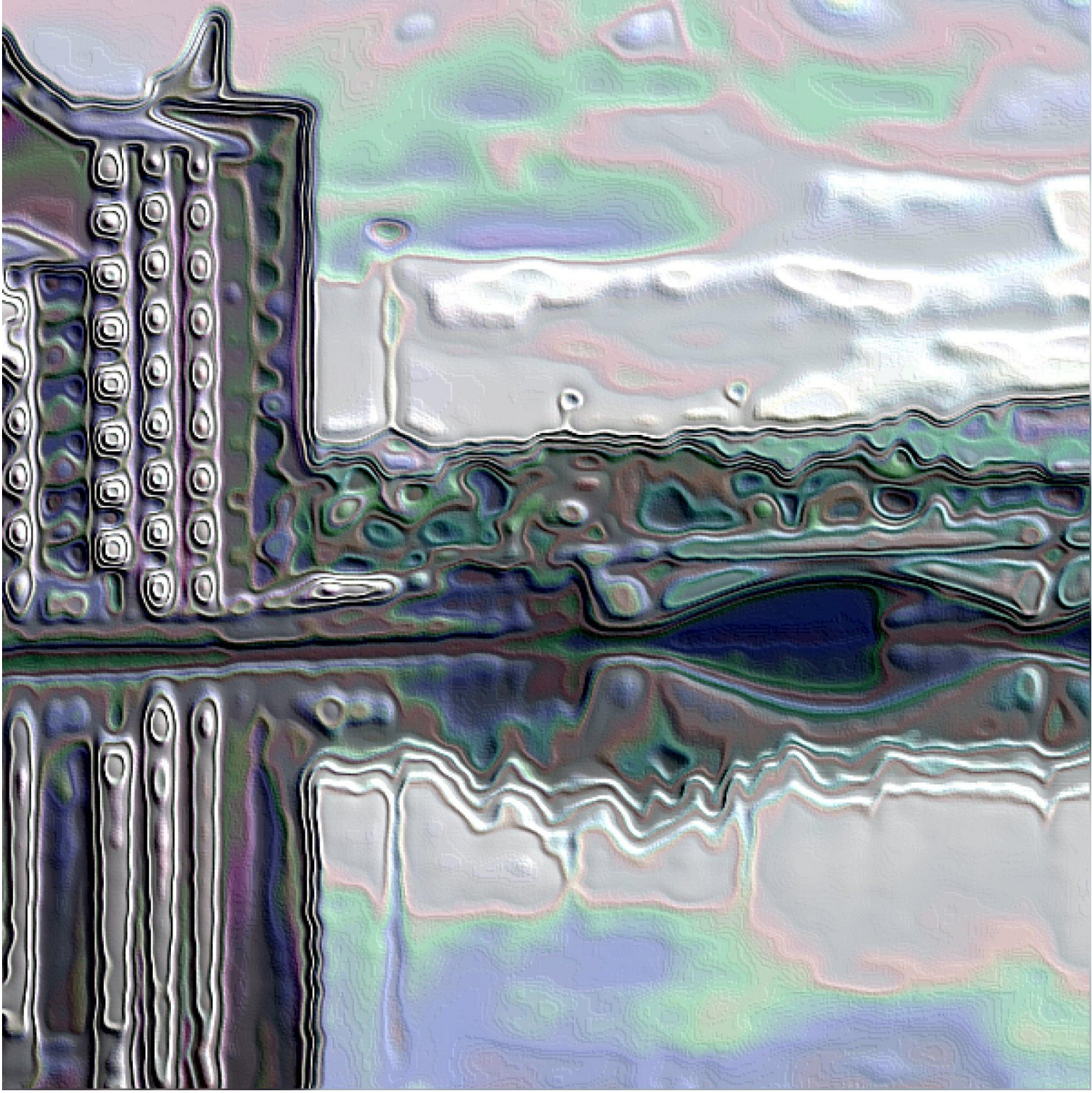


Scrum Manager II

Para avanzar en scrum

v. 2.5

Scrum Manager®

Scrum Manager II

Para avanzar en scrum

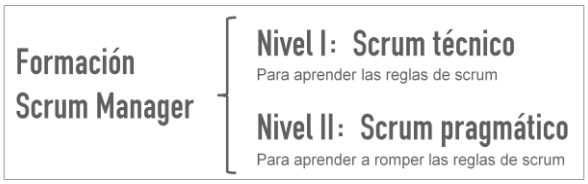
Versión. 2.5 – Abril 2014

Diseño de cubierta: Scrum Manager.
Imagen derivada de la original: The Albert Bridge 04 – Belfast de William Murphy (http://www.streetsofdublin.com/)

Contenido

Formación Scrum Manager

Este libro cubre el nivel II del conocimiento troncal de formación Scrum Manager®



Los contenidos de formación se mantienen regularmente actualizados. Puede descargar la última versión, o consultarla en línea en la dirección: http://www.scrummanager.net/bok

Este documento es un ***recurso educativo abierto (OER)*** y puede emplearse gratuitamente para consulta y autoformación.

No está permitido su uso para actividades comerciales.

Los recursos educativos abiertos de Scrum Manager son posibles gracias al soporte de los:

Servicios de formación y asesoría Scrum Manager

- ***Cursos*** en convocatorias presenciales, o a medida para empresas o grupos de estudiantes.
  - Puede consultar el calendario de convocatorias en diferentes ciudades en la página de cursos de http://www.scrummanager.net:
  - Para información de cursos a medida: contacte con un Centro Oficial de Formación Scrum Manager (http://scrummanager.net/centros) o en la dirección admin@scrummanager.net

- ***Exámenes presenciales de certificación.*** con supervisión de la ejecución de la prueba, e identificación del alumno
  - Incluidos en los cursos presenciales.
  - Disponibles en centros de formación autorizados.

Puede localizar centros certificados 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 admin@scrummanager.net

Más información:

http://www.scrummanager.net – (preguntas frecuentes)
http://www.scrummanager.net/oks
admin@scrummanager.net

Mejora continua y control de calidad Scrum Manager

Gracias por elegir los servicios de formación de Scrum Manager

Su valoración es el criterio del control de calidad de Scrum Manager, y decide la validez o no de los servicios de formación, y en su caso la continuidad de los cursos, centros y profesores.

Si ha participado en una actividad de formación auditada por Scrum Manager, le rogamos y agradecemos que valore la calidad del material, profesor, temario, etc. así como tus comentarios y sugerencias.

Scrum Manager anonimiza la información recibida, de forma que comparte con los profesores, y centros autorizados las valoraciones y aspectos de mejora, pero en ningún caso los nombres de los alumnos que las han realizado.

Puede realizar la valoración en la página accesible a miembros de Scrum Manager: http://scrummanager.net/qa.

1.- Conocimiento en continua evolución

Los marcos de prácticas ágiles no llegan a los proyectos TIC como “tesis” de conocimiento, sino como “antítesis” al que la Ingeniería del Software venía desarrollando.

Comenzaremos viendo qué significa esto, y así tomar la distancia necesaria para ver con perspectiva las razones por las que los proyectos TIC abrazaron la agilidad a finales del siglo pasado, y sus diferencias con la ingeniería de procesos; no desde las prácticas concretas, sino desde los principios en los que se basan, y con ello comprender las fortalezas y debilidades de la agilidad.

Alcanzar una visión de las razones y los principios de cada metodología, más allá de la concreción de un modelo es clave para dar el salto de gestión técnica a gestión experta. Esto es, de gestión basada en la aplicación de prácticas a gestión basada en la aplicación del propio conocimiento.

Gestión técnica: Gestión basada en la aplicación de modelos de prácticas y procesos.

Gestión experta: Gestión basada en el conocimiento tácito del gestor

El patrón dialéctico

Al cuestionar el conocimiento, se inicia su evolución que sigue un patrón dialéctico de: tesis, antítesis y síntesis.

De manera esquemática el patrón dialéctico puede definirse como el ritmo de avance que contrapone una ***antítesis*** a una concepción previa, entendida como ***tesis***. La antítesis muestra los problemas y contradicciones de la tesis, y de la confrontación surge un tercer momento llamado ***síntesis***, una resolución o una nueva comprensión del problema.

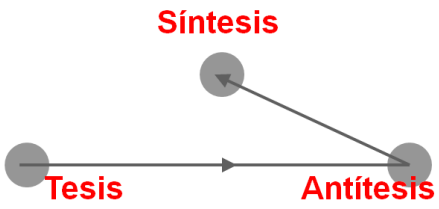


Ilustración 1: Patrón dialéctico

De esta forma la estrategia de abordar con ingeniería de procesos los retos de los proyectos de software, supuso la primera tesis para dar respuesta a la “crisis del software”, y sus problemas y contradicciones han sido puestos de manifiesto por su antítesis: la agilidad.

En 1968, en la primera conferencia sobre desarrollo de software celebrada por la organización OTAN, se analizaron los problemas de la programación del software, y en ella se acuñó el término “crisis del software” para referirse a ellos.

La conclusión de la conferencia (Bauer, Bolliet, & Helms, 1969) fue la necesidad de crear una disciplina científica que, como ocurría en otras áreas, permitiera aplicar un enfoque sistemático disciplinado y cuantificable al desarrollo, operación y mantenimiento de los sistemas del software, es decir, la aplicación de la ingeniería de procesos al software. Fue el nacimiento de la Ingeniería del Software.

La primera estrategia de la Ingeniería del software (tesis) se ha basado en dos pilares:

- Ingeniería de procesos:
- Gestión predictiva:

El primero para aplicar el principio básico de calidad contrastado con éxito en los entornos de producción industrial: “la calidad del resultado depende de la calidad de los procesos empleados”.

El segundo para garantizar el cumplimiento de agendas y presupuestos.

Mientras esta disciplina evolucionaba y se perfeccionaba a través de diferentes modelos de procesos y cuerpos de conocimiento para gestión de proyectos (MIL-Q9858, ISO9000, ISO9000-3, ISO 12207, SPICE, SW-CMM…) en la industria del software surgían dudas y se cuestionaba esta estrategia.

¿La planificación predictiva es apropiada para cualquier proyecto? ¿Los criterios de éxito son siempre el cumplimiento de fechas, costes y funcionalidades preestablecidas?

Empiezan a surgir proyectos cuya finalidad no es construir un sistema previamente definido y planificado en su totalidad, y para los que no es realista trazar un plan cerrado desde el inicio. Proyectos en los que no interesa saber si el sistema final tendrá 20 o 200 funcionalidades, ni conocer cómo serán éstas en detalle: Su interés es poner una novedad en el mercado lo antes posible, y desde ese momento evolucionar la visión y valor de forma continua.

Por otra parte también se cuestiona si el software se puede producir con patrones de procesos industriales, y se empieza a aceptar que en la calidad del resultado puede ser más importante el conocimiento tácito de la persona que lo realiza que el *know-how* aportado a través del proceso y la tecnología empleada.

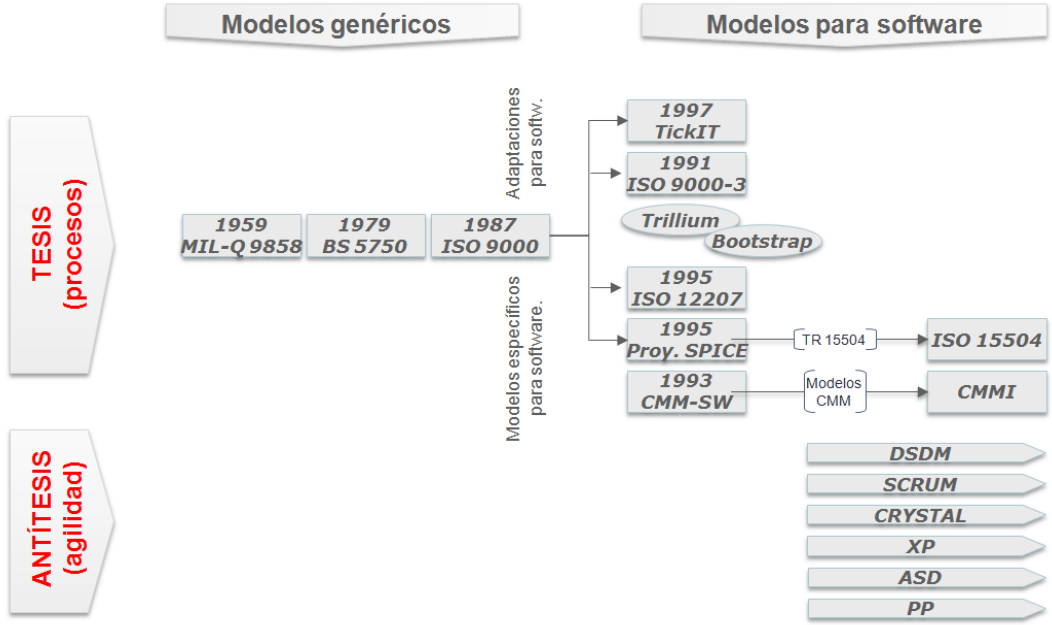


Ilustración 2: Evolución de los primeros modelos de mejora

Desde los orígenes de la agilidad, a mediados de los 90, hasta 2005-2010 han sido habituales las posturas radicales entre los defensores de los modelos de procesos y de los marcos ágiles, posiblemente más enfocados en descalificar al otro que en revisar y depurar los propios métodos.

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 de 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)

Espiral dialéctica del conocimiento.

El conocimiento profesional evoluciona de forma continua porque la realidad en la que se aplica está en permanente movimiento, y también porque la mejora siempre es posible.

La puesta en funcionamiento de nuevas técnicas, procesos o modelos genera sus propias antítesis al revelarse las debilidades, contradicciones y puntos de mejora, y el enfrentamiento de ambos conduce hacia una síntesis, que pasará a ser una nueva tesis, cuyo posterior uso producirá su antítesis, desarrollando a través de este patrón dialéctico una espiral de evolución continua del conocimiento (Nonaka & Takeuchi, 2004)

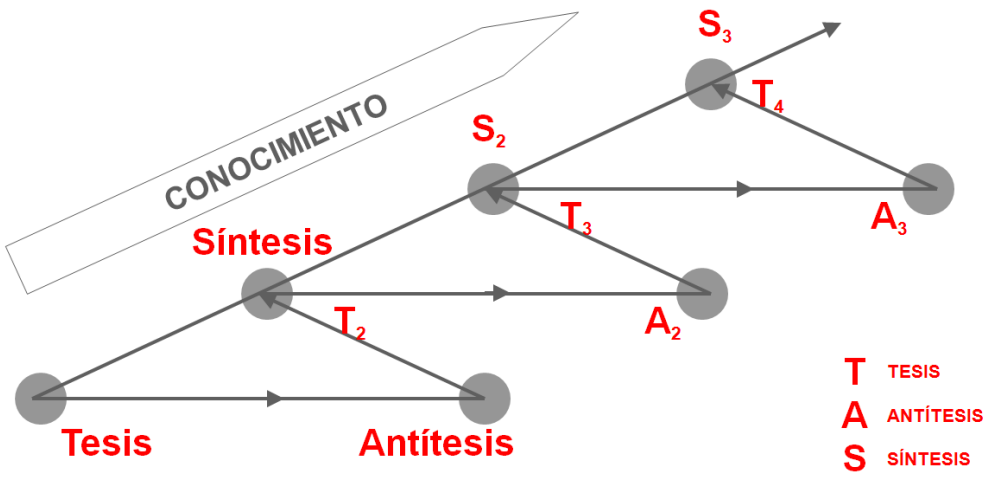


Ilustración 3: Espiral dialéctica del conocimiento

En disciplinas no técnicas y en generaciones anteriores el ritmo de avance sobre esta espiral dialéctica permitía a los profesionales desempeñarse con los conocimientos adquiridos en su licenciatura durante toda su carrera profesional. Sin embargo hoy esto no es posible, en especial, en el sector TIC

No hay métodos, prácticas o modelos de trabajo que nos ayuden con solvencia durante mucho tiempo, sino conocimiento en evolución. Esta es una consideración clave en el marco de Scrum Manager y la razón por la que no define un modelo fijo, sino un conocimiento actualizado como base para una gestión más experta que técnica. Más basada en el criterio documentado y experto del gestor que en la aplicación de prácticas o procesos.

2.- Empresa como sistema

Las empresas no están formadas por departamentos o áreas más o menos independientes. Son realidades sistémicas, y su eficiencia es proporcional a la armonía de los modos de trabajo de los diferentes departamentos. La consideración de scrum como el marco de reglas de trabajo propias del ámbito de gestión de proyectos, cuyas prácticas se pueden adoptar sin implicaciones en el resto de la organización produce resultados limitados e incluso contraproducentes.

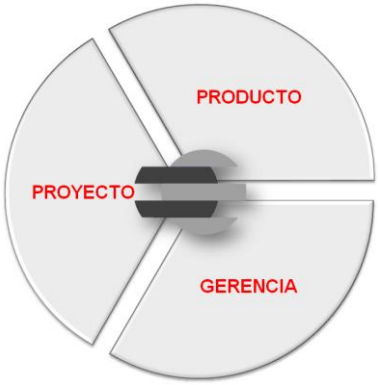


Ilustración 4: La empresa como sistema

Por ejemplo, en una organización cuya gerencia dirige con orientación a modelos de producción industrial, y el área de ingeniería en consecuencia trabaja con modelos basados en procesos con ciclos de vida secuenciales o de cascada, la adopción de prácticas ágiles en el área de gestión de proyectos generará problemas de funcionamiento.

3.- Flexibilidad

El objetivo no es implantar un marco de scrum basado en reglas. El objetivo es alcanzar una organización ágil en su conjunto, capaz de “avanzar en scrum” en su concepción original. Capaz de responder en escenarios de trabajo que evolucionan rápidamente, o tienen dosis altas de incertidumbre por las que no cuentan con requisitos estables al concebir nuevos productos o servicios. Se trata de clientes que necesitan empezar a usar un producto lo antes posible y mejorarlo de forma continua. De productos en los que la innovación es un valor clave.

Un principio básico de la implantación pragmática de scrum es la flexibilidad, que consiste en que las prácticas de scrum se adapten a la organización y no al revés. Se trata en definitiva de realizar una gestión experta más que una gestión técnica. Una gestión dirigida desde el conocimiento, experiencia y criterio del gestor y no tanto una gestión orientada a la búsqueda e implantación del mejor modelo. Una gestión basada en la persona antes que en el modelo.

El conocimiento de las distintas técnicas y metodologías amplía el criterio y el fondo de recursos del gestor.

Para seguir la evolución del conocimiento profesional y para ampliar y mejorar de forma continua el criterio e inventario de recursos profesionales propios es aconsejable:

- Vencer la resistencia al cambio y evitar actitudes de adopción o defensa dogmática de un modelo.
- Espíritu crítico-constructivo: Cuestionar continuamente de forma “antitética” los modos actuales, con el conocimiento y criterio profesional adecuar el sistema de trabajo propio a las características del proyecto, equipo y organización.

Scrum pragmático

Scrum Pragmático

Adaptar las prácticas scrum a las circunstancias de la propia organización, permite emplear técnicas de incremento continuo o iterativo; tableros kanban con el formato más adecuado a cada proyecto, y en general las prácticas y reglas que mejor encajan en las circunstancias de cada caso.

De esta forma se van abandonando los renglones de guía de las reglas definidas, y aplicando directamente los valores de scrum.

Scrum técnico

Reglas

Marco de reglas para desarrollo de software

Autores: Ken Schwaber y Jeff Sutherland
"Scrum Development Process OOPSLA'95" 1995

Aplicación de reglas definidas

Roles

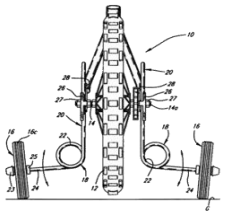
- Dueño de producto
- Equipo de desarrollo
- Scrum Master

Eventos

- El Sprint
- Reunión de planificación
- Scrum diario
- Revisión de sprint
- Retrospectiva de sprint

Artefactos

- Pila de product
- Pila de sprint
- Incremento



Aprender las reglas de Scrum

Scrum pragmático

Valores

Concepto original Scrum

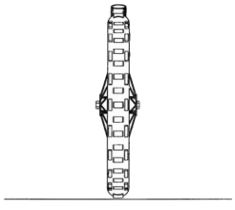
Autores: Hirotaka Takeuchi e Ikujiro Nonaka
"The New New Product Development Game" 1986

Aplicación de valores ágiles

- Personas > procesos
- Resultado > documentación
- Colaboración > negociación
- Cambio > planificación

… "Para avanzar en Scrum"

- Incertidumbre
- Autoorganización
- Fases de desarrollo solapadas
- "Multiaprendizaje"
- Control sutil
- Difusion del conocimiento



Aprender a avanzar en Scrum sin reglas

Responsabilidades

Al pasar del scrum técnico, basado en reglas, al scrum pragmático, para aplicar directamente principios de gestión ágil con en el conocimiento y experiencia de los equipos, y con una cultura, ya ágil en la organización, el ámbito de responsabilidades que se deben cubrir va más allá de los roles de proyecto:

La organización, como realidad sistémica debe dar respuesta de forma coordinada y alineada con su visión, a responsabilidades en tres áreas: Gerencia, procesos y producción.

Ilustración 5: Áreas de responsabilidad Scrum Manager

De gerencia

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

De procesos

- Configuración flexible de scrum
- Mejora continua
- Garantía de funcionamiento de scrum en cada proyecto ***(en scrum técnico asignada al rol de Scrum Master)***

De producción

- Producto***(en scrum técnico asignada al rol de Propietario del producto)***
- Auto-organización ***(en scrum técnico asignada al equipo)***
- Tecnología ágil ***(en scrum técnico asignada al equipo)***

El uso de prácticas y tecnologías ágiles, el trabajo en equipos autoorganizados, disponer de una visión de producto definida y gestionada durante todo el proyecto, y garantizar el funcionamiento de scrum durante la ejecución, son responsabilidades que pertenecen al 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, dispongan de medios para el diseño e implantación de una implantación ágil adecuada a la empresa, mejora continua del modelo y formación para las personas, son responsabilidades en el ámbito de la organización.

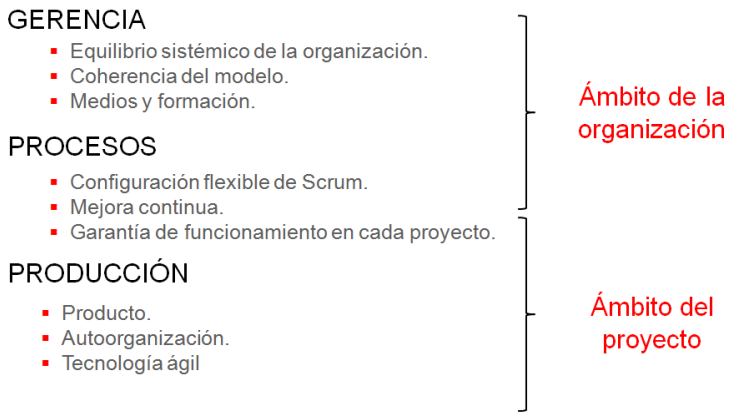


Ilustración 6: Ámbitos de responsabilidad Scrum Manager

En scrum técnico, las responsabilidades del ámbito del proyecto las asumen roles definidos:

- La responsabilidad de funcionamiento de scrum se asigna a un rol de gestor específico para el funcionamiento de scrum: Scrum Master.
- La responsabilidad de visión y gestión del producto al rol específico de propietario del producto, o product owner.
- La responsabilidad de autoorganización y uso de prácticas y tecnologías ágiles es propia del equipo.

Lo más aconsejable en fases de implantación, en equipos no familiarizados con desarrollo ágil es la adopción del modelo de roles de scrum técnico.

En la evolución hacia un nivel más maduro y global de agilidad en la organización es aconsejable adaptar el marco de scrum a la realidad de la organización, de forma que lo relevante no sea la presencia de determinados roles y reglas, sino cubrir adecuadamente todas las responsabilidades necesarias a nivel de organización.

Un ejemplo de asignación flexible de las responsabilidades del ámbito de proyecto sobre el esquema de puestos ya existente en la organización podría ser:

- Garantía de funcionamiento de scrum => Calidad o procesos
- Garantía de gestión de producto => Product manager
- Autoorganización y tecnología ágil = Equipo

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

Metodologías

Mapa de metodologías.

Desde los 80 se han desarrollado tantos modelos de procesos, marcos y prácticas de trabajo para mejorar la calidad y eficiencia en los proyectos de software, que resulta útil trascender las etiquetas y llegar a la base de los principios que subyacen, y las estrategias con las que los desarrollan; de forma que usando como coordenadas tres conceptos: “desarrollo, trabajo y conocimiento”, y dos modelos de gestión: “predictiva y evolutiva” se despeja y simplifica el aparente laberinto de modelos de procesos, marcos o prácticas de trabajo a los que nos referimos: CMM-SW, CMMI, PMBOK, DSDM, Crystal, ISO 15504, RUP, XP, scrum, ITIL, ASD, PRINCE 2, LEAN, KANBAN, TDD, etc..

Las diferentes prácticas y metodologías responden a combinaciones de tres conceptos y dos patrones de gestión de proyectos.

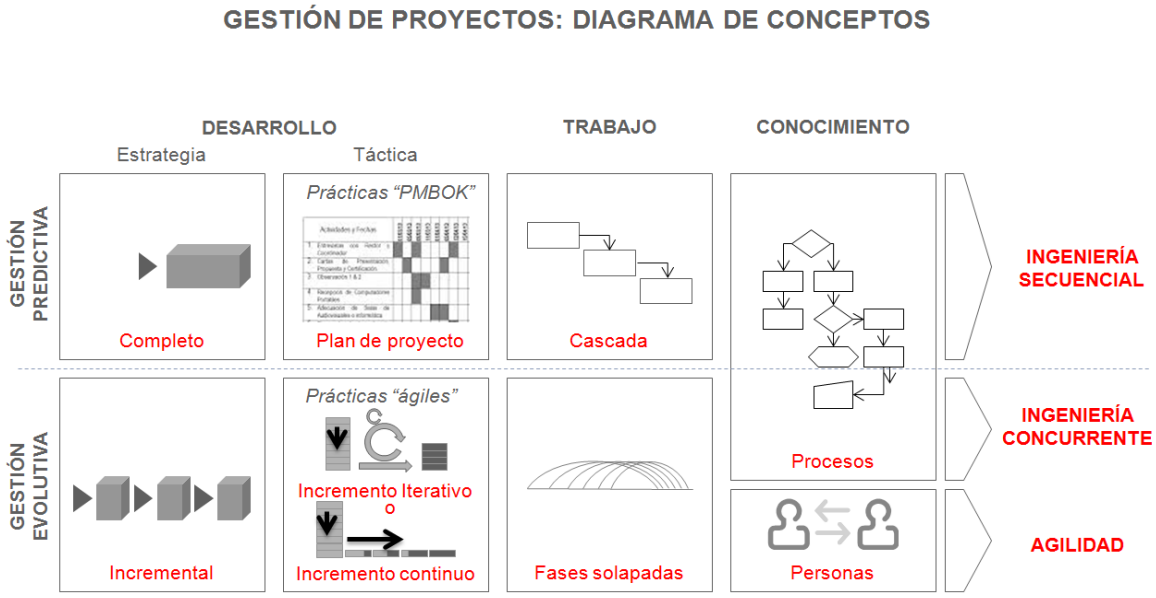


Ilustración 7: Diagrama de conceptos de la gestión de proyectos

Conceptos

1.- Desarrollo

Completo: La descripción de lo que se desea obtener está disponible al inicio del proyecto, es completa y detallada, sirve de base para estimar el plan del proyecto: tareas, recursos y agenda de trabajo. Durante la ejecución se gestiona su cumplimiento.

Incremental: La descripción de lo que se desea obtener no está disponible de forma completa y detallada al inicio: se complementa y evoluciona en paralelo al desarrollo, que genera el resultado de forma incremental y que se puede gestionar con dos tácticas diferentes:

- Desarrollo incremental continuo: Empleando técnicas para lograr un flujo continuo de desarrollo de las funcionalidades o partes del producto, que se entregan de forma continua al cliente.
- Desarrollo iterativo: Empleando técnicas de tiempo prefijado o *timeboxing* para mantener la producción de incrementos del producto de forma cíclica y continua. Este es el marco de producción empleado al aplicar el marco estándar de scrum, que define como sprint a cada iteración de desarrollo, al final de la cual se produce un incremento del producto.

2.- Trabajo

Secuencial (cascada): Divide el trabajo en fases, y cada fase comienza al terminar la anterior. El ejemplo más habitual es el ciclo de cascada definido en Ingeniería del software con las fases de requisitos, análisis, diseño, codificación, pruebas e implementación.

Concurrente: Solapa en el tiempo las diferentes fases. Siguiendo con el ejemplo de ingeniería de software, la definición de requisitos, el análisis, la codificación y el despliegue del resultado se realiza y revisa de forma simultánea y continua.

3.- Conocimiento

¿Dónde se encuentra el principal conocimiento empleado, en la correcta ejecución del proceso o en el saber hacer de la persona?

Producción basada en procesos: El conocimiento o know-how, responsable de la calidad del resultado se encuentra en mayor medida en los procesos y la tecnología empleada: “La calidad del resultado depende de la calidad de los procesos empleados“.

Producción basada en personas: El conocimiento o know-how responsable de la calidad del resultado se encuentra en mayor medida en el “saber hacer” tácito de las personas que lo construyen.

Patrones de gestión del proyecto

Gestión predictiva

Modelo de gestión cuyo objetivo es ofrecer resultados predecibles: desarrollo del producto previsto, en el tiempo previsto, e invirtiendo los recursos previstos. Emplea una estrategia de desarrollo completo con prácticas de planificación tradicional. Los principales referentes en el desarrollo de conocimiento para este tipo de gestión son PMI e IPMA y los modelos de procesos CMMI, ISO 15504, SPICE, entre otros, que emplean ingeniería secuencial y producción basada en procesos.

Gestión evolutiva

Modelo de gestión cuyo objetivo es entregar lo antes posible un producto mínimo viable, e incrementar su valor de forma continua. Emplea una estrategia de fases de trabajo solapadas, y desarrollo incremental, que se puede obtener con tácticas iterativas o de mantenimiento de flujo continuo. Puede emplearse con producción basada en procesos (ingeniería concurrente) o con producción basada en personas (agilidad).

Es importante esta distinción porque sin ella se generan situaciones confusas que llegan a considerar agilidad a la simple aplicación de las reglas estándar de scrum (ciclo de incremento iterativo con roles y artefactos definidos), o al simple uso de técnicas de gestión visual kanban para mantener un flujo continuo de tareas.

Agilidad y gestión evolutiva no son lo mismo. Se puede hacer gestión evolutiva empleando agilidad o empleando ingeniería concurrente

Personas, Procesos y Tecnología

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

Procesos

Para diferenciar los ~~procesos~~[1] – procedimientos en sus dos tipos posibles, podemos decir que en uno, las personas ayudan al proceso, y en el otro son ~~los procesos~~ las prácticas las 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~~ la práctica una ayuda, una herramienta que simplifica aspectos rutinarios para que pueda lograr más eficiencia y no diluir el esfuerzo en rutinas mecánicas.

Por eso a los primeros los llamamos procesos y a los segundos prácticas.

La principal diferencia entre unos y otros es el tipo de conocimiento con el que trabajan.

El conocimiento pueden ser:

- ***Explícito:*** contenido en los procesos y la tecnología
- ***Tácito:*** que es contenido por la persona

Scrum Manager aporta una consideración sobre el triángulo tradicional personas-procesos-tecnología, considerando que los procedimientos de trabajo pueden ser:

- ***Procesos:*** Si su ejecución aporta conocimiento clave para lograr el resultado. Son por tanto contenedores de conocimiento "explicitado" en el proceso y la tecnología que emplea.
- ***Prácticas:*** Si el procedimiento ayuda a las persona, que es quien aporta con su conocimiento tácito, el "saber hacer" clave para lograr el resultado.

Se puede decir que en los primeros la persona ayuda al procedimiento, y en los segundos es el procedimiento el que ayuda a la persona.

Ilustración 8: Personas, procedimientos y tecnología

Los modelos de ingeniería de procesos, consideran al binomio "proceso-tecnología" como principal responsable de la calidad del resultado. Su antítesis, la agilidad, da el protagonismo del resultado a las personas.

[1] No los llamaremos procesos sino "procedimientos" dejando así el nombre "proceso" para el procedimiento que tiene explicitado en él el principal conocimiento para la obtención del resultado.

Desde el punto de vista de Scrum Manager, ambas opciones son válidas, pero para tipos de trabajos distintos. En entornos de producción industrial, las personas aportan trabajo para ejecutar y supervisar los procesos. Sin embargo para las empresas del conocimiento que trabajan en escenarios rápidos e innovadores, las personas aportan con su talento el *know-how* que da valor al resultado.

Personas

Las organizaciones que necesitan imprimir un componente innovador importante y frecuente, o que se mueven en sectores de innovación muy rápido, obtienen mejores resultados si hacen responsables de esa innovación al talento de las personas más que a la ejecución de procesos.

En este tipo de organizaciones es importante asegurar, además del nivel de creatividad del equipo, su capacidad para aprehender. El modelo de conversión del conocimiento definido por Nonaka y Takeuchi (Nonaka & Takeuchi, The Knowledge-Creating Company, 1995) define con sus 4 fases el proceso para la adquisición de las personas del conocimiento tácito a través de compartir experiencias, comunicación directa, documentos, manuales y tradiciones, que añade conocimiento novedoso a la base colectiva de la organización.

Gestión visual kanban para scrum

Adaptación de las prácticas a la organización.

Gestión visual kanban para obtener incremento continuo.

La imagen siguiente muestra dónde se sitúa scrum técnico, según lo descrito en el capítulo anterior:

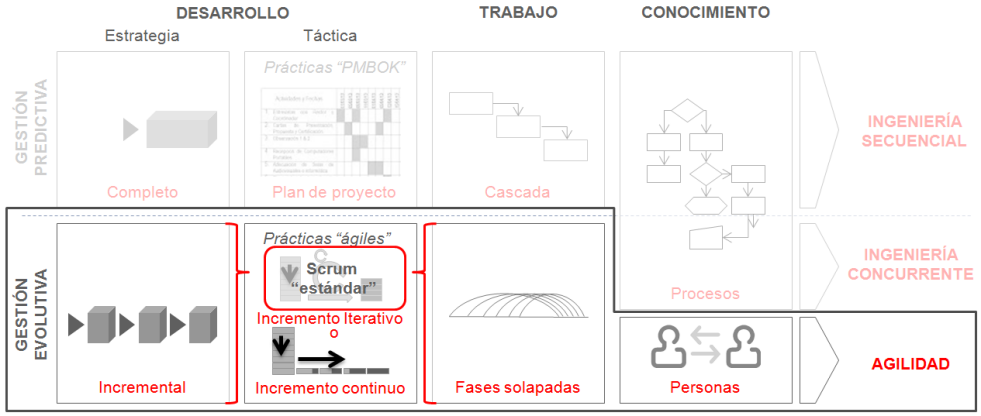


Ilustración 9: Agilidad con desarrollo incremental iterativo

Su característica principal es el uso de ***pulsos de sprint***, para emplear tiempo prefijado (*timeboxing*) como motor de avance al ritmo marcado por la secuencia de sprints.

La táctica de *timeboxing* ayuda al equipo a avanzar, al tiempo que mitiga la tendencia habitual a dilatar los tiempos de entrega previstos.

Los equipos originales de scrum observados y descritos por Nonaka y Takeuchi (Nonaka & Takeuchi, The New New Product Development Game, 1986) y los principios de la agilidad no prescriben el uso de una determinada táctica para lograr un desarrollo incremental. De hecho también es posible trabajar con un avance constante de las tareas una tras otra, sin empaquetar en incrementos.

Lograr un flujo continuo de tareas sin usar sprints no es fácil porque suelen formarse cuellos de botella que bloquean el avance, mientras en otras áreas del equipo se producen tiempos muertos sin tareas que realizar.

La gestión visual kanban es la técnica más empleada actualmente para regular un flujo de avance continuo en proyectos TIC y de servicios del conocimiento gestionados evolutivamente sin sprints.

Introducción: De la artesanía a la producción lean

Las técnicas de gestión visual que vamos a ver se inspiran en la producción lean. Esta introducción muestra de forma breve cuáles han sido los principales hitos en la producción industrial hasta llegar a lean.

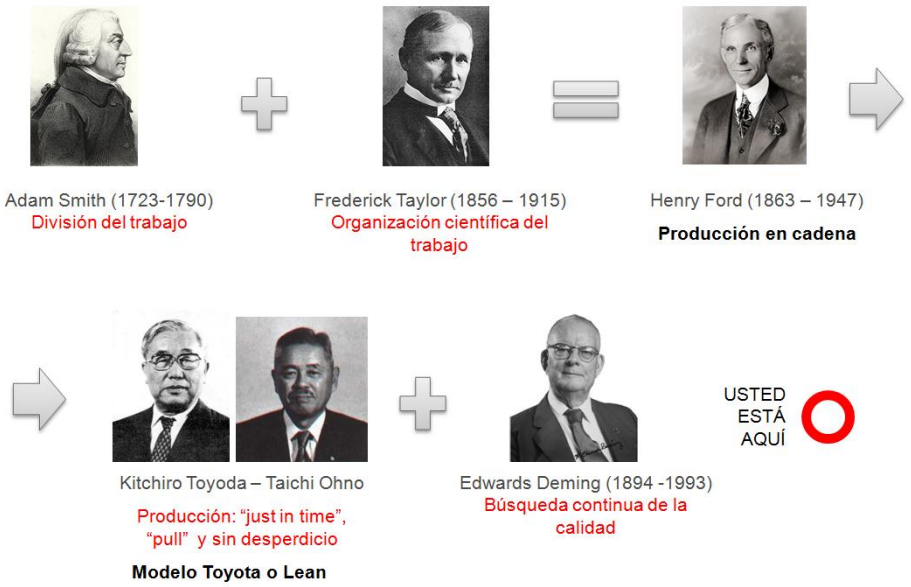


Ilustración 10: De la artesanía a la producción lean

División del trabajo

"La riqueza de las naciones" (Smith, 1776) es la obra más célebre de Adam Smith. Fue publicada en 1776 y se considera el primer libro moderno de economía. En él desarrolla la teoría económica sobre la división del trabajo.

Para Smith el principal factor para mejorar la productividad del trabajo es su división, que ilustró con su célebre ejemplo de la manufactura de alfileres, en el que comparaba la producción que puede alcanzar un herrero realizando todas las tareas necesarias, con la obtenida en un sistema con división del trabajo entre obreros especializados en cada tipo de tarea (estirado del alambre, cortado, afilado, etc.).

Como demostró, la división del trabajo hace posible producir 5.000 alfileres diarios por obrero, frente a los 50 que produciría un artesano.

Organización científica del trabajo

Frederick W. Taylor fue el primero en analizar y estudiar el trabajo de forma sistemática.

Taylorismo es el nombre dado al método de producción cuyo principal objetivo fue el aumento de la productividad, basado en la división y especialización del trabajo. Al taylorismo se le denomina también "organización científica del trabajo" o "gestión científica del trabajo" por aplicar principios básicos del método científico en el diseño de los procesos de trabajo.

Taylor expuso su sistema de organización racional del trabajo en su obra "Principles of Scientific Management" (Taylor, 1911) que de forma somera se puede condensar en cuatro principios:

- Reemplazar los métodos artesanales por métodos basados en el análisis científico del trabajo.
- Seleccionar y formar a los empleados con criterios científicos, en lugar de dejar que aprendan de forma autónoma.
- Dividir las tareas en gestión y trabajo, de forma que los gerentes puedan gestionar los principios de planificación y ejecución del trabajo.
- Proporcionar instrucciones y supervisión detallada a cada trabajador.

Producción en cadena

La producción en cadena, también denominada *fordismo*, es el sistema de producción desarrollado por el fabricante de automóviles Henry Ford para la fabricación de los primeros automóviles de su factoría en la primera década del siglo XX.

Pone en práctica los principios de la división y la organización científica del trabajo, y ha sido ampliamente utilizado para la producción industrial hasta que en la década de los 70 comenzó a ser reemplazada por el *toyotismo*.

El *fordismo* hizo posible:

- Reducción significativa del costo de producción. (El precio del "Ford T" pasó de $850 en 1908 a 250$ al producirse en cadena en 1927).
- Flujo constante de la producción.
- Ingeniería de procesos: la calidad del resultado depende de la calidad del proceso empleado en su fabricación.

Sistema de producción Toyota

El fundador de Toyota, Sakichi Toyoda, su hijo Kiichiro Toyoda y el ingeniero Taiichi Ohno, evolucionaron los sistemas de producción de sus factorías transformando el fordismo hasta llegar a principios de los 70 en lo que acabaría siendo y denominándose "Sistema de Producción Toyota" o toyotismo.

Cuando intentaron importar los sistemas de producción masiva norteamericanos para rehacer su industria tras la segunda guerra mundial, se dieron cuenta de que el fordismo no les valía. Creaba empresas para producir grandes cantidades del mismo producto, pero no servía para producir pequeñas cantidades de productos diferentes.

El sistema fordista, al producir muchas existencias del producto generaba sobreproducción y almacenes llenos ante la falta de clientes. Necesitaban un sistema que fabricara con costos reducidos, pero no a costa de incrementar la producción, puesto que no tenían gran demanda.

Estas circunstancias evolucionaron la producción hacia un sistema que elimina el número de existencias en el proceso de producción y el "sobreefectivo" tanto de personal como de equipo para lograr la fábrica mínima, capaz de producir lo necesario: lo que un cliente está esperando.

El sistema fordista es un sistema de empuje ("push") mientras que el toyotismo es un sistema de arrastre ("pull"). En el fordismo cada puesto empuja continuamente al siguiente, enviándole trabajo de forma sistemática. Sobre la estimación de consumo, se pone en marcha la cadena y se produce al ritmo previsto.

El toyotismo es un sistema de arrastre: un puesto no empuja al siguiente sino que tira del anterior, pidiéndole trabajo por la demanda de un cliente.

Para lograr la simplicidad organizativa y flexibilidad que permitan al sistema adaptarse a variaciones del flujo de demanda y diferencias en el producto, el toyotismo emplea diversas técnicas de apoyo:

- Información y control visual kanban.
- Información y control visual andon.
- Sistemas de prevención de errores Poka-yoke.
- Herramientas polivalentes y de cambio rápido.

El toyotismo o "Sistema de Producción Toyota" es conocido actualmente como Lean.

Lean

La palabra “lean” en inglés significa “magra”, es decir, sin grasa.

> Lean es un sistema de mejoramiento de procesos de manufactura basado en la eliminación de desperdicios y actividades que no agregan valor al proceso.

14 principios Lean

1. Las decisiones del negocio están basadas en una **visión a largo plazo**, aún a expensas de las pérdidas financieras a corto.
2. Los ciclos son cortos y rápidos.
3. Se prefieren los sistemas ***“pull”***, que evitan la sobreproducción.
4. La carga de trabajo debe ser balanceada (Heijunka).
5. La cultura lean comprende **detener la producción para arreglar problemas**, así como en enseñar el estudio metódico de los problemas (Jidoka).
6. Las tareas se estandarizan para lograr la **mejora continua** (Kaizen).
7. La **gestión visual simple** revela problemas y permite la coordinación.
8. Se utiliza solamente **tecnología probada** que pueda ser provechosa para la gente y su proceso.
9. Se forman ***líderes*** que comprendan el trabajo, vivan la filosofía de la empresa y la enseñen a otros.
10. Se desarrollan ***equipos y personas excepcionales que siguen la filosofía de la compañía.***
11. Se respeta la red de ***colaboradores y proveedores (Keiretsu) ,*** desafiándolos a crecer y ***ayudándolos a la mejora.***
12. *Se valora que los responsables* ***vayan y miren las situaciones en el lugar de trabajo, para entenderlas y poder ayudar.***
13. Decisiones basadas en el ***consenso y la consideración minuciosa de todas las opciones, y su posterior implementación*** rápida.
14. Empresa como organización que aprende a través de la ***reflexión constante (Hansei) y de la mejora continua (Kaizen)***

Algunas prácticas comunes en la producción lean

He aquí algunas de las prácticas que se utilizan en entornos lean:

- **Kanban**: o presentación de información visual relativa a la producción (identificación de componentes, estado del proceso, etc) presentada físicamente en el lugar de trabajo implicado y permanentemente actualizada.
- **5S**: Metodología de trabajo cuyo nombre procede de las 5 palabras japonesas que la configuran: seiri, sieton, seiso, siketsu and shitsuke (clasificar, ordenar, limpiar, estandarizar y sostener).
- **Poka-yoke**: o sistema a prueba de error, que busca crear mecanismos que sólo permiten hacer el trabajo de la forma adecuada.
- **JIT** (Just in time): producir en el momento que es requerido.
- **Andon**: Sistemas visuales de alertas sobre anomalías, fallos o problemas en el momento y lugar de la producción.
- **Jidoka**. Instalación en el proceso de sistemas que verifican su calidad.
- **Heijunka**. Técnicas para adaptar la producción a una demanda fluctuante del cliente.

Lean Software Development

Este término se refiere a la aplicación de los principios lean en el desarrollo del software. Mary y Tom Poppendieck (Poppendieck & Poppendieck, 2003) lo acuñaron. Gracias a sus aportes y los de la comunidad ágil, Lean Software Development está desarrollando un inventario de prácticas útiles para el desarrollo ágil de software. .

Se basa en 7 principios:

1. ***Eliminar el desperdicio***
   - Las actividades que no crean valor no sirven y deben ser eliminadas. Algunos ejemplos:
     - Tareas que no fueron solicitadas por el cliente.
     - Sobre-documentación del proyecto.
     - Procesos de desarrollo que se ejecutan sin analizar su nivel de eficiencia o vigencia.
     - Un mayor número de líneas de código no siempre es mejor, y además requiere mayor esfuerzo de testeo y de mantenimiento.
     - Los errores, bugs y fallos del software son verdadero desperdicio que se debe reducir.

2. ***Construir con calidad***

   Incluir en el procedimiento prácticas para mejora de la calidad en el producto (traslación de prácticas “poka-yoke” y “andon” al desarrollo de software).
   Un procedimiento que respeta la calidad es aquel que es conocido, entendido y mejorado por los propios participantes. Para lograrlo es necesario compromiso y respeto.

   Algunos ejemplos de prácticas que se deben contemplar al hacer software.
   - Técnicas como TDD (Test Driven Development) permiten que usuarios (clientes), programadores y *tester* definan claramente los requerimientos y confeccionen pruebas de aceptación antes de escribir el código. Ayuda a la comprensión de los programadores y mejora el entendimiento de los requerimientos.
   - El programador es responsable de su propio desarrollo. No debe esperar a que las pruebas o los procedimientos de aseguramiento de calidad descubran los errores.
   - Fomentar el desarrollo de pruebas automatizadas.
   - Refactorización del código, para lograr simplicidad y eliminar duplicidades.

3. ***Compartir conocimiento***

   Conocer lo que necesita el cliente requiere dedicación y esfuerzo, y debe convertirse en el aspecto principal, porque desarrollar un producto que no es útil, es el mayor desperdicio.
   Hacer software implica un proceso de aprendizaje: entender qué es lo que el cliente quiere y cómo entregar la mejor solución posible. El desarrollo incremental proporciona cuantiosa y frecuente retroinformación.

4. ***Diferir el compromiso***

   En los proyectos ágiles que parten con una visión que evoluciona con el desarrollo, el compromiso con el cliente se asienta y evoluciona en la misma medida que se van concretando y comprometiendo los incrementos del producto.

5. ***Entregar rápido***

   La gestión evolutiva realiza entregas rápidas a los clientes, que se encuentran con código operativo desde etapas tempranas. Dicho código debe ser desarrollado con calidad ya que no se puede mantener una velocidad importante de entrega si no se cuenta con calidad y un equipo disciplinado, comprometido y confiable.

6. ***Respetar a las personas***

   Lean se basa en el respecto por las personas que son el elemento único y diferenciador de cada organización.
   Deben estar suficientemente capacitadas y ser responsables de los procesos en los que intervienen, de modo que cuando resultan necesarios cambios y mejoras, cada persona colabora en su desarrollo.

7. ***Optimizar el todo***

   Lean invita a contemplar el proceso completo, es decir todo el flujo de valor, en lugar de hacerlo en cada etapa. El problema de optimizar cada fase por separado es que genera inventarios grandes en los puntos de transición. En el mundo del software, estos "inventarios" representan al trabajo parcialmente terminado (por ejemplo, requisitos completos, pero sin diseñar, codificar o probar). Lean demostró que un flujo de "una pieza" (por ejemplo, enfocarse en construir un ítem de manera completa) es un proceso más eficiente que concentrarse en construir las partes separadas de forma rápida.

Kanban: Origen y definición

El término japonés Kanban, fue el empleado por Taiichi Onho (Toyota), para referirse al sistema de visualización empleado en los procesos de producción que coordinan en una cadena de montaje la entrega a tiempo de cada parte en el momento que se necesita, evitando sobreproducción y almacenamiento innecesario de producto.

Se puede traducir como tablero o tarjeta de señalización, y su origen se remonta finales de los cuarenta o principio de los cincuenta.

El término kanban aplicado a la gestión ágil de proyectos se refiere a técnicas de representación visual de información para mejorar la eficiencia en la ejecución de las tareas de un proyecto

Kanban en el sector TIC

El uso de tableros kanban muestra y gestiona el flujo de avance y entrega, y ayuda a evitar los dos problemas más importantes: cuellos de botella y tiempos muertos.

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.

Desde 2005 es cada vez más frecuente reemplazar los formatos de lista para las pilas de producto y de sprint por notas adhesivas en tableros, que resultan más versátiles al poder cambiar su posición, bien para reordenar las prioridades de las historias de una pila de producto, o para reflejar a través de su posición en, cuáles se están programando, probando, o se encuentran terminadas.

Las prácticas kanban son válidas para gestión evolutiva con entrega continua. Deben emplearse con criterios de flexibilidad, sin considerar prescripciones ni excepciones en el método de trabajo, para lograr la implementación personalizada, que dé la mejor respuesta a los principios ágiles, de ingeniería concurrente, o de síntesis de ambos, con los que trabaje la organización.

Gestión técnica vs. gestión experta

Algunos autores consideran a scrum y kanban como marcos de reglas y prácticas diferentes.

Según Kniberg & Skarin al considerarlos así, se dibujarían las siguientes diferencias entre ellos(Kniberg & Skarin, 2009):

- Scrum prescribe roles, kanban no.
- Scrum trabaja con iteraciones de tiempo fijo, kanban con cadencias (simples, múltiples o dirigidas por eventos).
- Scrum limita el wip por iteración, kanban limita el wip por estado del flujo de trabajo.
- Los equipos de scrum son multidisciplinares, en kanban pueden ser de especialistas.
- Scrum no permite cambiar tareas del sprint, en kanban la tarea puede alterarse hasta entrar en el flujo.
- En scrum la pila de producto debe tener la longitud de al menos un sprint. En kanban se debe atender al ritmo de arrastre de tareas.
- En scrum se deben estimar las historias y las tareas y calcular la velocidad, kanban no mide tareas ni velocidad.
- Scrum necesita una pila de producto priorizada, en kanban es la siguiente historia o tarea arrastrada desde el cliente.
- Scrum prescribe reuniones diarias, kanban no.
- Scrum emplea diagramas burndown, kanban no.
- Los tableros scrum se *resetean* al final de cada sprint.

Al evolucionar hacia un modelo de scrum pragmático, basado en la aplicación de los valores de la agilidad con la experiencia y conocimientos propios, y abandonar los modelos basados en reglas, se aprende a romper éstas y flexibilizar las prácticas, quedando como triviales cuestiones “técnico-metodológicas” que de otra forma distorsionan la realidad y el foco de la gestión:

Situación A: “Por un lado se desea usar kanban, pero por otro se quiere estimar las tareas (por ejemplo para registrar la velocidad por razones organizativas de mi empresa) …”

Situación B: "La empresa gestiona proyectos simultáneos con una organización de oficina de proyectos y por eso encaja mejor el establecimiento de roles; pero sin embargo se quiere trabajar con kanban en lugar de con scrum… ¿Debería hacer scrumban? ¿qué es eso? ¿o lo que se va a hacer es Scrumbut? ¿es la solución de un mal gestor?."

Tableros kanban: conceptos

Las prácticas de gestión visual kanban son útiles para:

1. Gestionar el flujo de trabajo.
2. “Radiar” información.

Un tablero kanban puede emplearse con una de estas finalidades, o con ambas a la vez; aunque en realidad si se usa para gestionar el flujo de trabajo, siempre acompaña el hecho de mostrar la información del trabajo que se está controlando.

1.- Características de kanban para gestionar el flujo de trabajo.

Sobre un tablero kanban se pueden establecer pautas para regular el flujo de avance de las tareas.

La posición de cada tarjeta sobre el tablero refleja el estado en el que se encuentra el trabajo que representa.

Los estados mínimos habituales en un tablero kanban son “pendiente”, “en curso” y “terminado”.

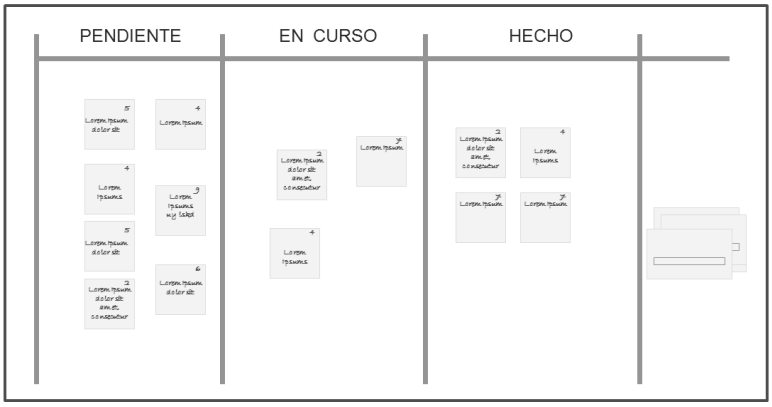


Ilustración 11 – Estructura básica de un tablero kanban

En algunos casos es conveniente incluir estados adicionales (por ejemplo: testeado, validado).

El orden de los trabajos desde el área “pendiente”, refleja la secuencia de tareas prevista, según sus prioridades.

Los trabajos monitorizados pueden ser tareas, historias de usuario o epics, según el uso al que se dedique el tablero

Kanban saca a la superficie la información de los problemas.

Los conflictos en la priorización de los trabajos, los problemas en el flujo por impedimentos o cargas de trabajo, las incidencias en el desarrollo, etc. se ponen de manifiesto de forma inmediata al actualizar sobre el tablero el estado de los trabajos.

Kanban Facilita un ritmo sostenido y evita la ley de Parkinson

Genera un avance continuo de trabajo cuyo ritmo no está “predestinado” por una planificación temporal: Gantt o Sprint (incremento iterativo).

La ausencia de hitos temporales evita la tendencia habitual de alargar el tiempo de trabajo hasta completar el tiempo estimado (ley de Parkinson).

> El trabajo se expande hasta llenar el tiempo que se había previsto
> Ley de Parkinson.

Por otra parte, la ausencia de hitos temporales, sin técnicas de monitorización y gestión del avance generaría alargamiento de tiempos y retrasos por aplazamiento y perfeccionismo.

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

2.- Radiador de información.

Kanban como “radiador de información:

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.

Facilita la detección temprana de problemas

- Kanban monitoriza continuamente la evolución 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 pasan 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.

Kanban: Operativa

Secuencia y polivalencia

Dos son los factores que delimitan cuatro escenarios de trabajo diferentes

- Secuencia (del trabajo).
- Polivalencia (de las personas).

Secuencia.

¿Los trabajos reflejados en las notas adhesivas del tablero deben ejecutarse en un orden determinado o pueden realizarse en cualquier orden?

No es lo mismo diseñar un tablero para el equipo de programadores de un sistema, que para el de mantenimiento de los sistemas informáticos de una empresa. Los primeros deben realizar las tareas en un orden determinado. Así por ejemplo, no es posible realizar la tarea de pruebas si antes no se ha hecho la de programación. Sin embargo las siguientes podrían ser las tareas de un equipo de mantenimiento: “instalación de nueva impresora en el equipo de dirección” “actualización del sistema operativo en el servidor web”, etc. Este tipo de tareas se pueden realizar en cualquier orden. No hay una relación de dependencia entre ellas de forma que no se pueda realizar una si la otra no se ha completado.

Polivalencia

¿Es un equipo polivalente o de especialistas? ¿Por el tipo de trabajo y el perfil de los integrantes del equipo, cualquier miembro puede realizar cualquier tarea?

Siguiendo con los ejemplos anteriores, es posible que en el equipo de mantenimiento una persona pueda indistintamente instalar una impresora, o un sistema operativo; o es posible que no, que haya técnicos de hardware y técnicos de software. De forma similar un proyecto de programación puede incluir tareas específicas de diseño gráfico, programación, integración, testing, etc. que pueden realizar sólo determinados miembros del equipo.

La imagen siguiente refleja el valor clave de kanban: la gestión de un flujo continuo de avance, y los factores que se deben considerar para configurar el tablero con la estructura más adecuada a nuestro trabajo y equipo.

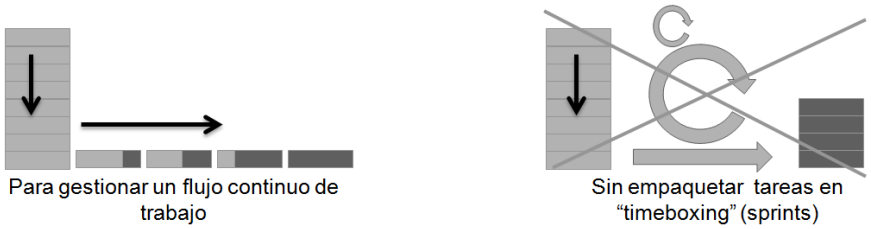


Ilustración 12: Fortaleza y variables clave de los tableros kanban

Cuatro son los patrones posibles según se combinen un tipo de trabajo secuencial o libre, con un equipo polivalente o de especialistas:

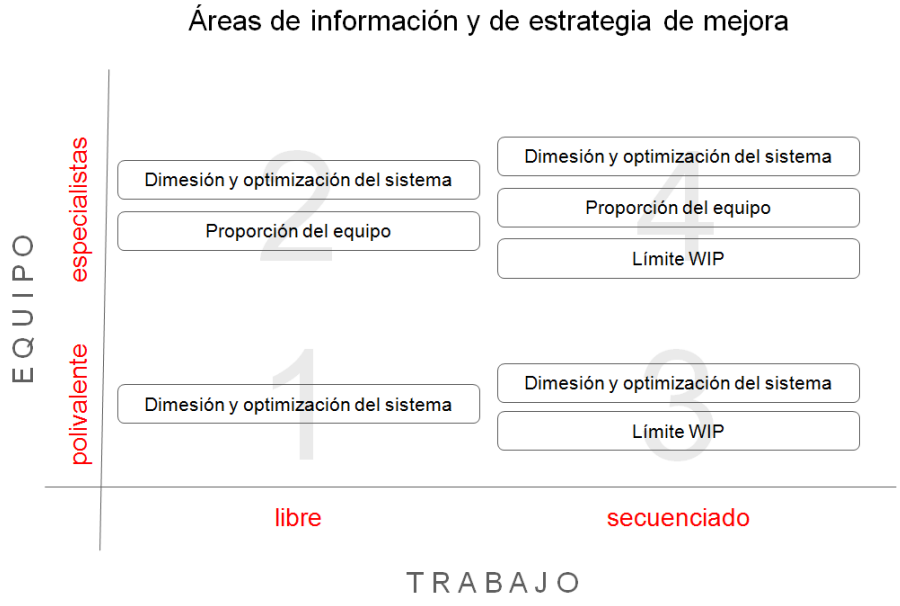


Ilustración 13: Áreas de información y mejora reveladas por los tableros kanban

1.- Equipo polivalente que realiza tareas no secuenciales.

Este es el entorno más fácil de gestionar: cualquier persona del equipo puede hacer cualquier tarea, y las tareas se pueden tomar en cualquier orden.

Para solucionar problemas de saturación de trabajo, o tiempos muertos se debe ajustar la dimensión del equipo y / o la optimización del sistema.

2.- Equipo de especialistas que realiza tareas no secuenciales.

La especialización del equipo aporta un factor de complejidad para solucionar cuellos de botella o los tiempos muertos.

Si se producen cuellos de botella, la estrategia con el tipo de tareas que los provocan debe ir en una o en ambas de las siguientes líneas:

- Dimensionamiento: bien del número de personas capacitadas para realizar ese tipo de tareas, bien en el número de tareas de ese tipo que se pueden comprometer, o en el tiempo de respuesta al cliente.
- Optimización del proceso de ejecución de ese tipo de tareas.

La presencia de tiempos muertos debe cuestionar el dimensionamiento de la demanda, o su distribución no homogénea.

3.- Equipo polivalente que realiza tareas secuenciales

En este caso es la dependencia entre tareas la principal causa de tensiones en el flujo.

La primera línea de mejora en estos casos es el ajuste del WIP de cada fase, antes de considerar modificaciones en el dimensionamiento del equipo y el compromiso.

WIP es un término inglés que en el campo de la manufactura lean, de donde proviene, se emplea para indicar la cantidad de productos en proceso de fabricación, que aún no están terminados.

En los tableros kanban, por analogía se emplea este término para indicar las tareas que se encuentran en una fase del proceso, pendientes de pasar a la siguiente o de completarse; y en este entorno el término WIP indica límite o número máximo de tareas que se pueden acumular en un área determinada. Así por ejemplo, decir que en un tablero kanban para programación de software el área de testing o pruebas “tiene un WIP de 3” quiere decir que no puede haber más de tres tareas simultáneamente en esa fase.

4.- Equipo de especialistas y trabajo que requiere un orden secuenciado.

Este es el entorno más complejo porque requiere el ajuste y revisión en todas las líneas de mejora posible: dimensión y equilibrio de especialistas en el equipo, dimensión o equilibrio de tiempos de respuesta en el compromiso, y ajuste de límites WIP en cada fase.

En cada una de estas cuatro situaciones posibles el tablero saca a la superficie los problemas, y el equipo o gestor puede realizar los ajustes en las líneas de trabajo posibles según cada caso y en función de su criterio y las circunstancias de su organización.

Casos prácticos de tableros kanban

A continuación se muestran ejemplos de posibles tableros para dar soporte a las diferentes posibilidades vistas.

Ejemplo de tablero para ofrecer información del desarrollo del producto.

El ejemplo muestra un tablero que podría encontrarse en la oficina del responsable de producto mostrando el estado en el que se encuentra la construcción del producto.

En este caso no se emplea como herramienta de gestión visual, sino solamente como "radiador" de información.

Tablero para mostrar la siguiente información relativa al estado de desarrollo del producto:

- Historias de usuario sugeridas, que se encuentran en evaluación sin determinar aún si se incorporarán al producto.
- Historias de usuario aprobadas: se incorporarán al producto.
- Historias de usuario "preparadas" (ya valoradas y priorizadas) previstas para ser programadas.
- Historias de usuario que se están programando actualmente.
- Historias de usuario ya programadas que se pueden evaluar en el servidor de pruebas.
- Historias de usuario ya evaluadas, pendientes de desplegar.
- Historias de usuario desplegadas en las dos últimas versiones.

Ilustración 14: Ejemplo de tablero kanban para monitorizar el estado del producto

Ejemplos de tableros para desarrollo evolutivo, con incremento continuo, y con incremento iterativo.

Ilustración 15 – Tableros: incremento continuo – incremento iterativo

Este es un ejemplo de flexibilidad, o adecuación de las prácticas a las características de la organización. Kanban, además de ofrecer información visual, permite aplicar técnicas como limitación del wip, y muestra las áreas susceptibles de mejora, para ajustar la cadencia del flujo. Resulta útil, como ya hemos visto, para trabajar con incremento continuo.

¿Pero puede emplear kanban un equipo que haga scrum con incremento iterativo para emplear la representación visual del avance de las tareas, en lugar de usar un gráfico de avance?

Este apartado muestra ejemplos de tableros para ambos casos.

Caso 1: Incremento continuo.

Las siguientes imágenes representan posibles tableros para guiar la gestión del trabajo de un equipo que está desarrollando un producto con incremento continuo, y en el que se muestra la siguiente información:

- Pila de tareas.
- Tareas “preparadas”.
- Tareas en análisis.
- Tareas en codificación.
- Tareas terminadas.
- Tareas integradas en el servidor de desarrollo (labs).
- Tareas integradas en producción.

Ilustración 16: Ejemplo de tablero kanban para monitorizar y gestionar incremento continuo.

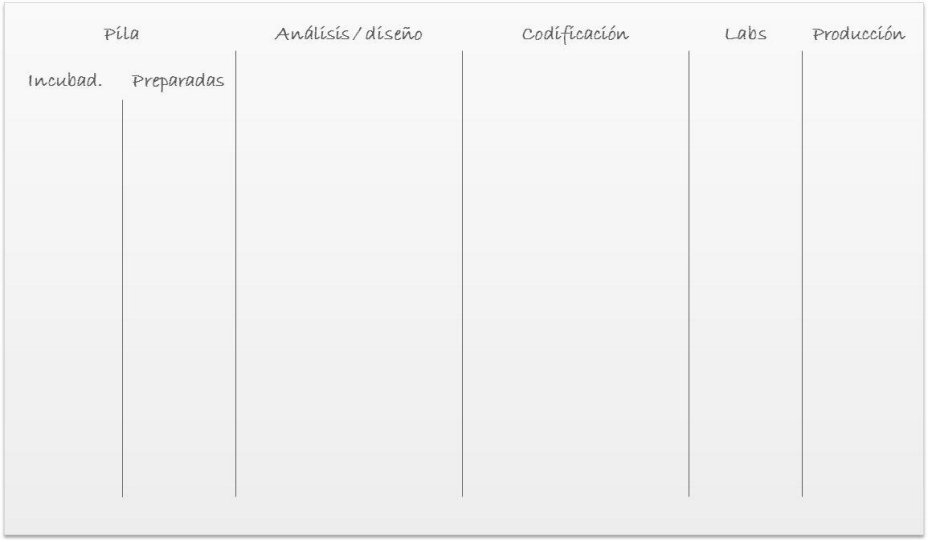


Ilustración 17 – Ejemplo de tablero kanban para monitorizar y gestionar incremento continuo.

Ilustración 18 – Ejemplo de tablero kanban para monitorizar y gestionar incremento continuo

Caso 2: Incremento iterativo.

Tablero para guiar la gestión de trabajo de un equipo que trabaja en incrementos iterativo (sprints) y que muestra:

- Pila de tareas.
- Tareas preparadas.
- Tareas en análisis.
- Tareas en codificación.
- Tareas terminadas.
- Tareas integradas en el servidor de desarrollo (labs)
- Tareas integradas en producción.

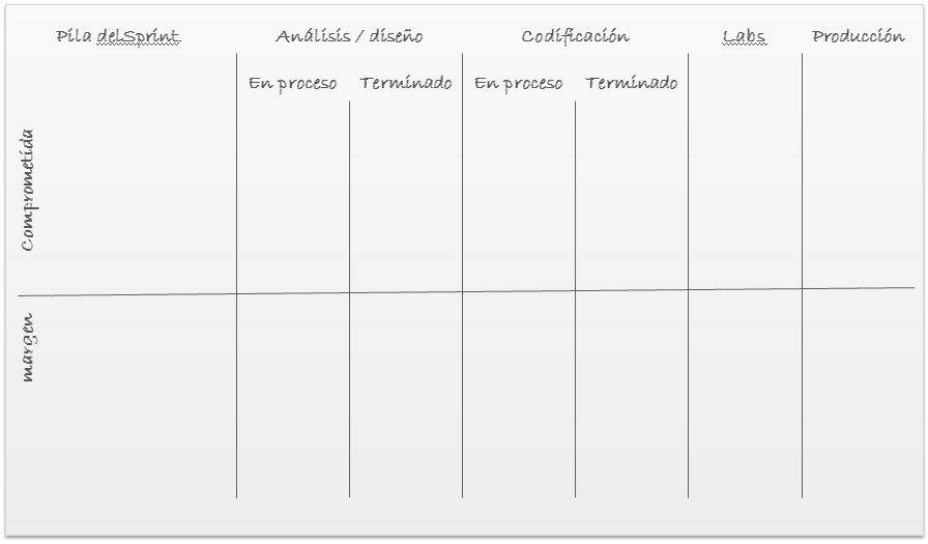


Ilustración 19: Ejemplo de tablero kanban para monitorizar y gestionar incremento iterativo

Ejemplo de tableros para un equipo de operación y mantenimiento.

Tablero guía para la gestión de un equipo de operación y mantenimiento que refleja:

- Estado de las tareas programadas para la semana y persona que está trabajando con cada una.
- Estado de incidencias no previstas y urgentes, y personas que están trabajando en cada una.

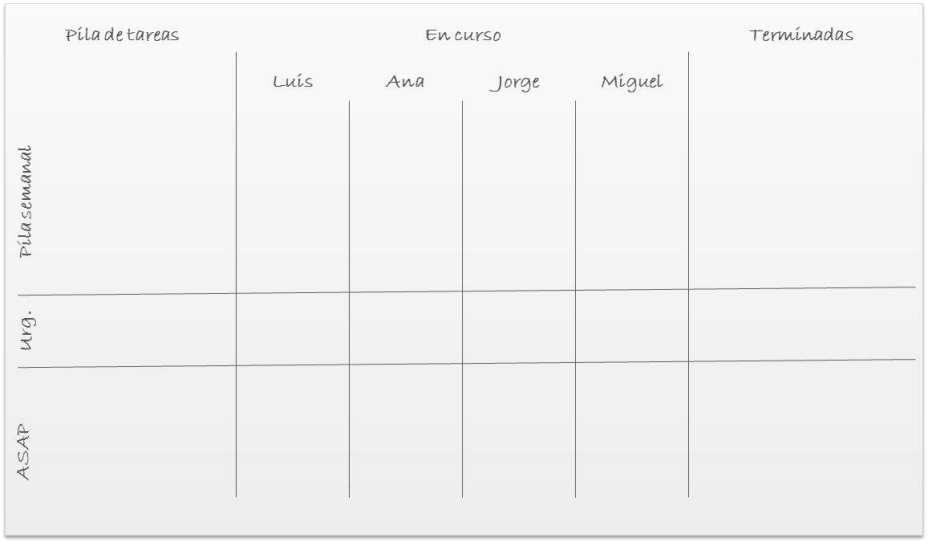


Ilustración 20: : Ejemplo de tablero kanban para monitorizar y gestionar tareas de mantenimiento

Kanban Box

Una práctica diseñada para gestionar tareas de varios proyectos en un mismo departamento de producción de software es una implementación Kanban, denominada Kanban Box.

La configuración es la siguiente:

La organización mantiene una “pila de producción” o lista de historias de usuario preparadas: pendientes, estimadas y priorizadas.

Si la organización trabaja en un único producto, la “pila de producción” es en definitiva la pila del producto.

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 21: Ejemplo de pila de producto con tarjetas kanban

En la pila de producción las tareas están preparadas, 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 se representan en una “caja kanban”:

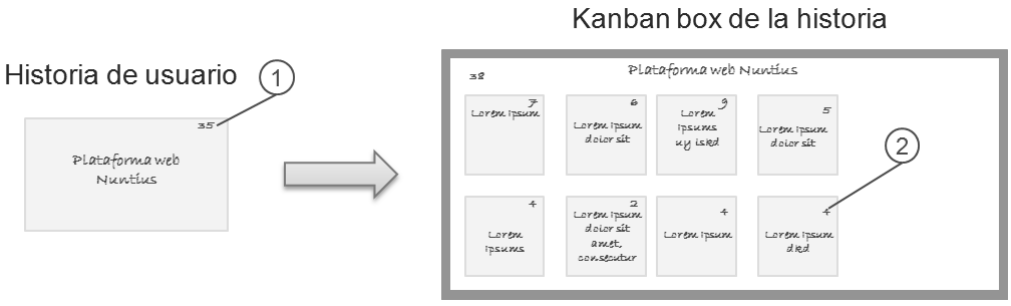


Ilustración 22: Ejemplo de implementación: kanban box

(1) Estimación de la historia.
(2) Estimación de la tarea.

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 23: Ejemplo de implementación: kanban box

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

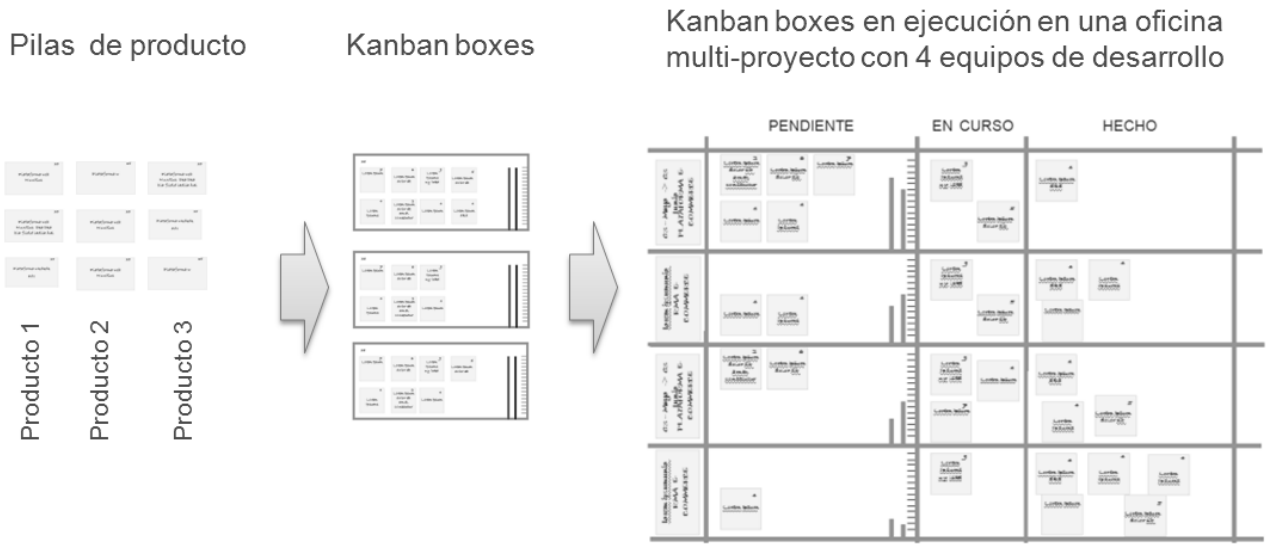


Ilustración 24 : Ejemplo de implementación: kanban box

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

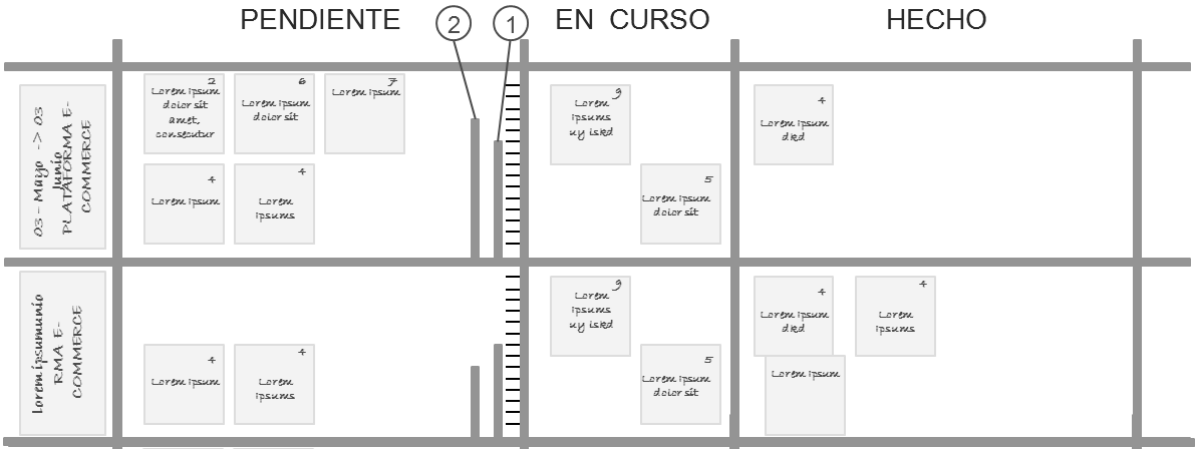


Ilustración 25 : Ejemplo de implementación: kanban box

A diario, cada equipo realiza el scrum diario, actualizando el estado de cada tarea (pendiente -> en curso -> hecho), y las barras de velocidad:

La barra de velocidad prevista (1) se actualiza todos los días considerando 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 de 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.

Muda, Mura y Muri y consejos para ajustar el flujo.

Muda, Mura y Muri son tres conceptos clave de la mejora continua Kaizen, que la producción Lean incorpora como variables protagonistas para incrementar la eficiencia.

- Muda: Desperdicio
- Mura: Discrepancia. Variabilidad del flujo de trabajo. Interrupciones en el flujo de trabajo. Tiempo muerto.
- Muri: Tensión. Sobrecarga de trabajo que produce cuellos de botella.

Mudas

Las mudas o desperdicios más habituales en los proyectos TIC son:

- Burocracia: Procedimientos, documentación y papeleo innecesario que no aportan valor al resultado.
- Sobreproducción: Desarrollar más características de las necesarias.
- Multiproyecto: Alternar el tiempo de trabajo entre varios proyectos e interrupciones del flujo de trabajo.
- Esperas: Tiempos de espera por falta de cadencia en el flujo de trabajo.
- “Ir haciendo”: Encargar trabajo para ir avanzando algo no definido y así no tener paradas a las personas.
- Desajustes de capacidad: Personas de gran talento asignadas a tareas rutinarias, y viceversa.
- Errores: Retrabajo por bugs.

Los tableros kanban detectan y ayudan a gestionar las otras dos variables kaizen: Mura y Muri.

Factores determinantes de Mura y Muri

Hay que tener en cuenta que cada organización o tipo de proyecto tiene sus características propias de:

- Secuencia: las tareas deben realizarse secuencialmente o no.
- Polivalencia o especialización de los miembros del equipo.

Y estos son los factores que en cada caso determinan la mayor o menor dificultad para mantener un flujo continuo de desarrollo.

Como ya se ha visto, los equipos de miembros polivalentes que trabajan con tareas no secuenciales son los que más fácilmente pueden conseguir un flujo de avance constante.

Cuando surgen dificultades, las variables que se deben combinar, según las posibilidades en cada caso, son:

- Volumen de la demanda.
- Orden del backlog o pila de historias de usuario: si se va a producir un cuello de botella en una fase, se debe procurar que la próxima historia que vaya a entrar al tablero requiera poco esfuerzo de esa fase.
- WIP o límite de tareas en una determinada fase.
- *Staffing*: Tamaño del equipo y especialización o polivalencia.

Muri

El WIP es una variable importante para ajustar los cuellos de botella (Muri):

Al emplear kanban como técnica con la que regular un incremento continuo, desaparece el concepto de sprint. El **incremento** no es el resultado de un sprint, sino cada historia de usuario que se termina y entrega. Para mantener un flujo continuo de funcionalidades que, una a una van incrementando el producto de forma sostenida, es necesario evitar la aparición de cuellos de botella (Muri): la acumulación de tareas en una determinada fase del proceso. Una técnica útil es limitar la cantidad de trabajo que puede acumularse en cada fase y generar cuellos de botella.

Al parámetro que indica el número máximo de tareas en un área del tablero kanban se le denomina WIP: Work In Process, o bien “in-process inventory” (inventario en el proceso). No se debe confundir con Work in progress (trabajo en progreso) término que designa un trabajo que ha comenzado pero aún no está terminado.

Un valor “WIP” demasiado bajo puede producir cuellos de botella en otras fases, en especial si el sistema es demasiado rígido (tareas secuenciales y equipo de especialistas).

La experiencia ayuda al equipo a ir mejorando el ajuste para mantener el flujo 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 (en equipos de miembros polivalentes) es empezar con un WIP igual al nº de miembros del equipo x 1.5, redondeando el resultado por exceso.

No 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.

Ejemplo:

La figura siguiente presenta un tablero kanban con límite de trabajo en los estados “Producto analizado” y “En curso”.

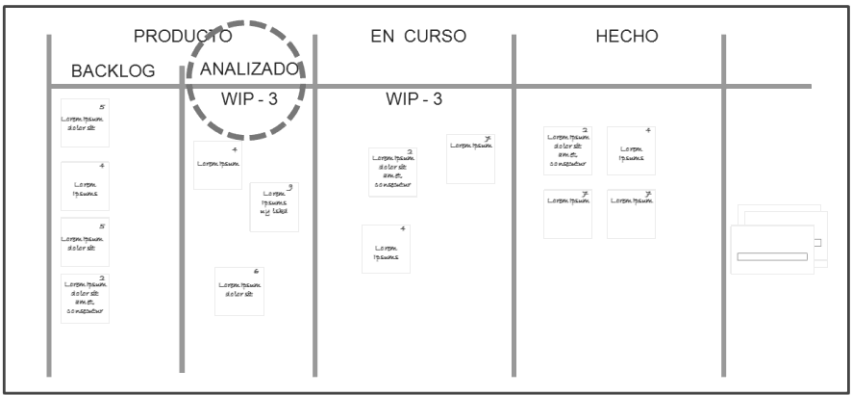


Ilustración 26 – WIP

En este ejemplo, el propietario de producto tiene una zona para ordenar 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 tres las historias que pueden estar en estado “analizado” para pasar a producción. Tres con la que ya está previsto analizar y revisar la estimación con el equipo.

De igual forma, el área “en curso” tiene un límite de tres historias. Hasta que una no pasa a “HECHO”, no puede entrar ninguna a producción, y de igual forma mientras haya tres en la zona “ANALIZADO” no se decide cuál será la próxima historia del backlog.

Así se fuerza un flujo de trabajo sin cuellos de botella, continuo y enfocado.

Mura

Los principales factores responsables de la variabilidad del flujo y la aparición de Mura o tiempos muertos son:

- Grado de multifuncionalidad de los miembros del equipo.
- Flexibilidad en el orden en el que se deben hacer las diferentes fases de cada tarea.
- Flexibilidad para alterar el orden de entrada de las historias de usuario desde la pila de producto.

Cuanto mayores sean estos aspectos, más fácil resulta evitar la aparición de tiempos muertos.

Bibliografía

Bauer, F., Bolliet, L., & Helms, H. (1969). Software Engineering. Report on a conference sponsored by the NATO SCIENCE COMITEE. *Software Engineering* (pág. 136). Garmisch: Peter Naur & Brian Randell.

Beck, K. (2000). *Extreme Programming Explained.* Addison-Wesley.

Hino, S. (2006). *Inside the Mind of Toyota: Management Principles for Enduring Growth.* Productivity Press.

Kniberg, H., & Skarin, M. (2009). *Kanban and Scrum, making the most of both.* crisp.

Nonaka, I., & Takeuchi, H. (1995). *The Knowledge-Creating Company.* University Press.

Nonaka, I., & Takeuchi, H. (1986). The New New Product Development Game. *Harvard Business Review* .

Nonaka, I., & Takeuchi, I. (2004). *Hitotsubashi on Knowledge Management.* Singapore: John Wiley & Sons.

Ohno, T. (1988). *The Toyota Production System: Beyond Large-scale Production.* Productivity Press.

Orr., K. (2003). CMM versus Agile Development: Religious wars and software development. *Cutter Consortium, Executive Reports 3(7)* .

Poppendieck, M., & Poppendieck, T. (2003). *Lean Software Development: An Agile Toolkit for Software Development Managers.* Addison Wesley.

Schwaber, K. (1995). *SCRUM Development Process.* Burlington: OOPSLA 95.

Smith, A. (1776). *An Inquiry into the Nature and Causes of the Wealth of Nations.* Londres: W. Strahan & T. Cadell.

Taylor, F. W. (1911). *The Principles of Scientific Management.* New York: Harper & Brothers.

Turner, R., & Jain, A. (2002). Agile Meets CMMI: Culture Clash or Common Cause? *XP/Agile Universe 2002* , 153-165.

Tabla de ilustraciones

Índice