Metodologías ágiles

INDICE

1. METODOLOGÍAS ÁGILES

Aunque existen otras, hay 4 metodologías ágiles que se utilizan muy habitualmente: Scrum, XP, Kanban y Lean.

Scrum

- Método simple
- Iterativo e incremental
- 5 valores
- 3 roles
- 3 artefactos
- 5 tipos de reunión

XP (Programación Extrema)

- 13 prácticas de ingeniería
- 5 valores
- 4 actividades
- 4 roles

Kanban

- Comenzar ahora con lo que hay que hacer
- Poner como objetivo pequeños cambios incrementales
- Cada miembro del equipo es un líder
- Mejorar de forma continua y colaborativa
- Todos los implicados ven el proceso claramente
- Limitar el trabajo en curso
- Hacer visible el trabajo
- Visualizar el flujo de trabajo
- Gestionar el flujo de trabajo

Lean

- Un enfoque, más que una metodología
- Eliminar el desperdicio
- Llevar la simplicidad al máximo

1.1 Scrum

Scrum es un método para solventar problemas complejos, entregando productos que aporten el mayor valor posible. Es una metodología:

- **Ligera**: Scrum tiene poca teoría, únicamente define algunas reuniones o ceremonias, los roles, y unos pocos principios básicos. El contenido teórico se lee en menos de 5 minutos.
- **Fácil de entender**: Es una metodología abierta, que no propone reglas complicadas ni demasiado específicas en función del proyecto.
- **Difícil de dominar**: La clave es adaptarla correctamente al entorno y al proyecto concreto. Por eso está definido el rol de Scrum Master, que es la figura que domina el método y ayuda a su aplicación y ajuste.

Está basada en procesos empíricos de control, es decir, el conocimiento viene de la experiencia, y se toman decisiones en función de la información que se tiene.
Su enfoque es iterativo e incremental.

- **Iterativo:** En cada sprint, se genera una nueva versión del producto, que mejora la versión del sprint anterior. Se trata de ir refinando y mejorando las propiedades del producto conforma avanza el proyecto.

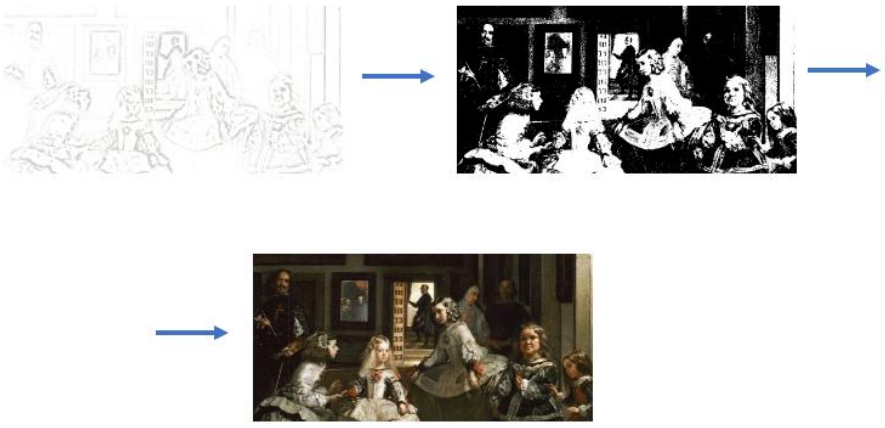

Figura 1: Desarrollo iterativo

- **Incremental:** En cada sprint, se añade alguna nueva característica al producto. Se trata de ir añadiendo nuevas capacidades o características al producto conforme avanza el proyecto.

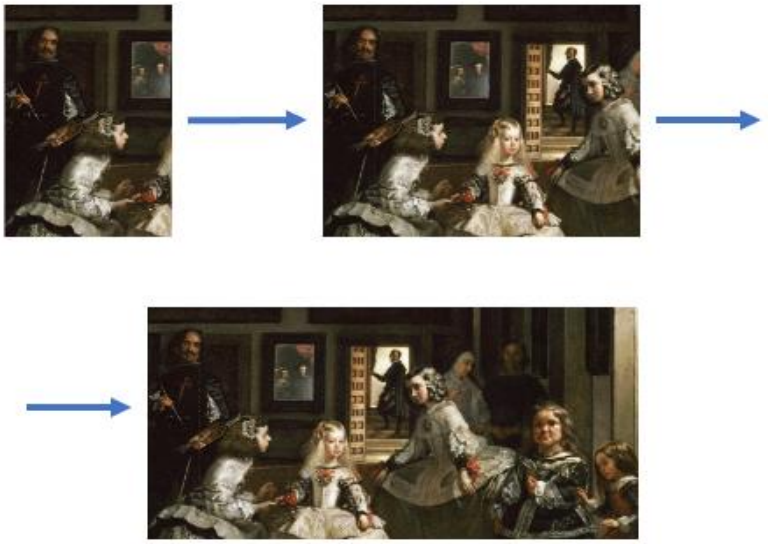

Figura 2: Desarrollo incremental

1.1.1. Elementos de Scrum

Sprints

El producto se construye de forma incremental en base a períodos de tiempo cortos, denominados Sprints.

Los Sprints tienen una duración fija y determinada, entre 1 y 4 semanas; mejor cuanto más cortas, es decir mejor 1 semana que 4.

Todos los Sprints tienen la misma duración a lo largo del proyecto, porque se rigen según el principio de “timeboxing”: cada elemento tiene un tiempo asignado que termina cuando acaba este tiempo.

Definición de Hecho

El equipo de trabajo tiene que encontrar una definición para el concepto de "hecho" -Done-. Cada incremento del producto debe cumplir dicha definición de "hecho" para darlo por finalizado, y poder ser entregado.

La definición de "hecho" puede aplicar a requisitos, sprints, releases, entornos... es decir, en cualquier elemento sobre el que se pueda plantear la cuestión de "¿está finalizado y puede

continuar al paso siguiente del proyecto?“ Son las condiciones para considerar el elemento terminado con éxito.

Veamos un ejemplo de "Hecho" para un requisito (habitualmente denominadas "historias de usuario"):

- Todo el código escrito sin errores
- Todas las pruebas unitarias pasadas correctamente
- Pruebas funcionales pasadas con éxito
- Documentación del requisito generada
- Pruebas de aceptación pasadas con éxito

Ejemplo de "Hecho" para un Sprint:

- Pruebas de rendimiento pasadas con éxito
- Elementos resultantes del sprint integrados
- Todos los bugs resueltos
- Pruebas de integración pasadas con éxito

Ejemplo de "Hecho" para el despliegue en Producción:

- Pruebas de stress pasadas con éxito
- Pruebas de seguridad validadas
- Plan de marcha atrás definido
- Entorno y servicio estable

Ciclo de Scrum

El proyecto se ejecuta en base a sprints, de duración fija, que se planifican al arrancar cada sprint, con las Daily cada 24 horas. En cada sprint se resuelve o construye el Sprint backlog, que se integra al final del sprint con el resultado de sprints anteriores, conformando un producto entregable.

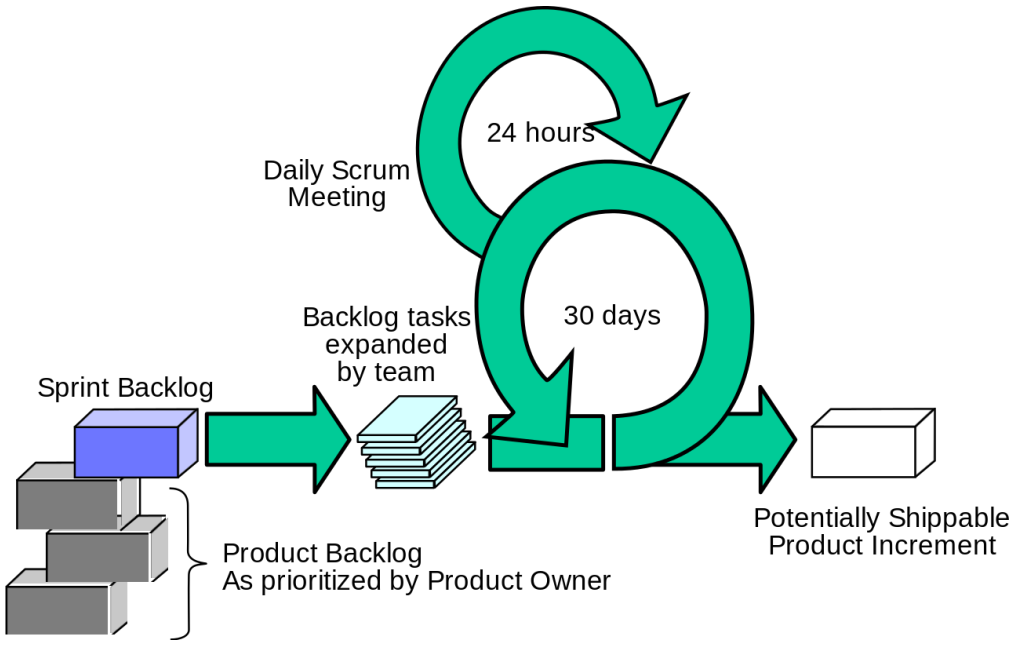

Figura 3: Ciclo de Scrum

Productos (También denominados "artefactos"):

- Incremento de producto: un subconjunto del producto que puede ser entregado, con componentes integrados, que funciona.
- Backlog de producto: la lista de requisitos del producto, ordenadas por su prioridad.
- Backlog del Sprint: el plan detallado para el desarrollo durante el sprint siguiente.

Pilares básicos

- Transparencia: los interesados comparten un entendimiento común del proyecto, de la visión, y de lo que significa "hecho"
- Inspección: a través de los artefactos o entregables (incremento de producto, backlog de producto y backlog del sprint).
- Adaptación: a través de las reuniones en Scrum

Valores fundamentales de la metodología

- Personas enfocadas en el resultado
- Motivación
- Transparencia
- Compromiso
- Respeto

Reunión Daily Scrum

Es una reunión con un timebox de 15 minutos, donde el equipo de trabajo sincroniza sus actividades y crea el plan para las siguientes 24 horas.

En esta reunión, cada miembro del equipo de trabajo debe responder a 3 preguntas:

- ¿Qué hice ayer?
- ¿Qué haré hoy?
- ¿Hay algún impedimento que me evite conseguir mis objetivos hasta mañana?

Reunión Sprint Review (también llamada "demo")

Es la reunión que se mantiene al final de cada Sprint para inspeccionar el Incremento de Producto, y adaptar el Backlog del producto si es necesario.

Durante el Sprint Review, el equipo de trabajo muestra al resto de interesados qué se ha conseguido en el sprint.

Si los sprints son de un mes, el timebox de la reunión serán 4 horas. Para sprints más cortos, la duración se reduce proporcionalmente.

Reunión Retrospectiva del Sprint

Se inspecciona cómo ha ido el sprint, en lo referente a las personas, sus relaciones, el proceso, y las herramientas.

Se identifican y ordenan los asuntos más importantes, tanto los que fueron bien, como los que suponen una mejora potencial.

Se crea un plan para implementar las posibles mejores detectadas.

Si los sprints son de un mes, el timebox de la reunión serán 3 horas. Para sprints más cortos, la duración se reduce proporcionalmente.

1.1.2. Roles en Scrum

En un proyecto ágil, típicamente bajo metodología Scrum, distinguimos 3 roles:

ProductOwner

- Decide qué se incluye (y qué no) en el backlog del proyecto
- Ordena los ítems en el backlog en función de su prioridad de negocio

- Explica y hace entender al equipo de trabajo en qué consisten esos ítems (historias de usuario)
- Decide cuándo se deben realizar las entregas

ScrumMaster

- Representa la figura de líder sirviente
- Es el experto en la metodología, guiando y enseñando al equipo a llevarla a cabo adecuadamente
- Soluciona problemas y elimina barreras, facilita el trabajo

Equipo de desarrollo

- Realiza el trabajo necesario para construir y entregar el producto final
- El equipo es un grupo de profesionales con todas las capacidades (en conjunto) para realizar el trabajo

Una idea fundamental es que se fomenta la comunicación directa entre el equipo de trabajo y el ProductOwner. No se debe pensar que el Scrum Master es el "traductor" del lenguaje de negocio a lenguaje técnico; al contrario, es el equipo de desarrollo el que trabaja directamente con la persona de negocio. El Scrum Master actúa como facilitador de esta relación.

Veamos de forma resumida los elementos de Scrum:

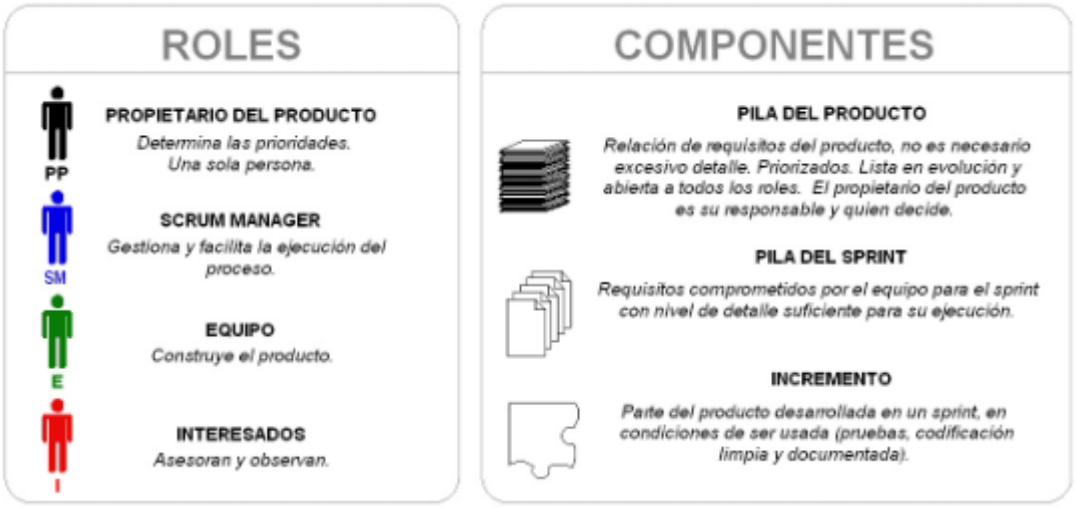

Figura 4: Roles y Componentes de Scrum

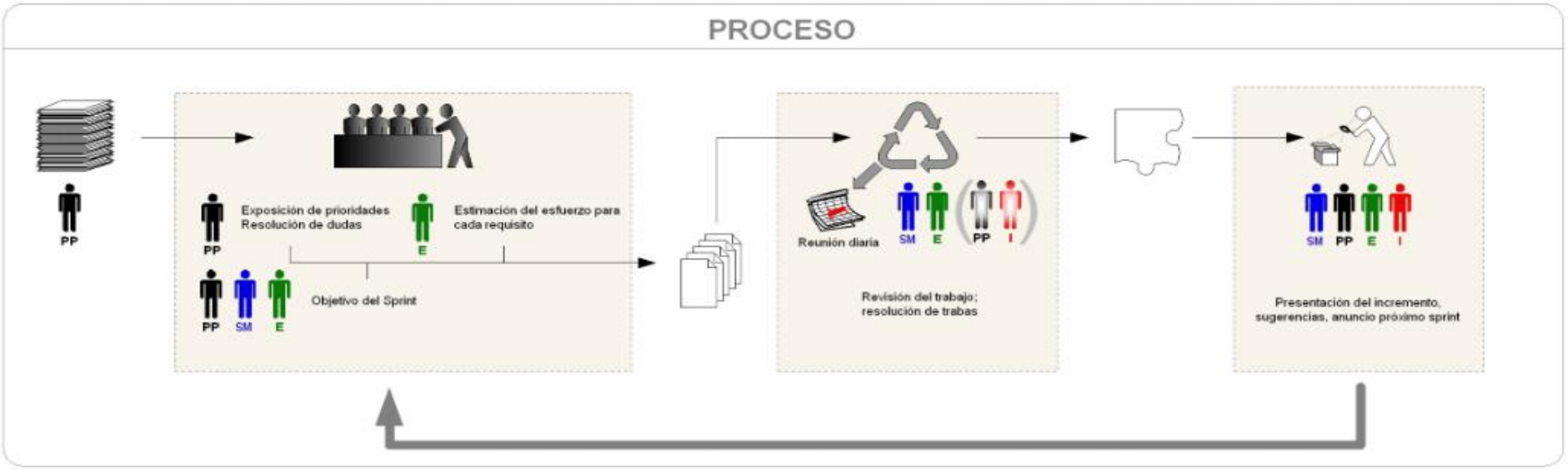

Figura 5: Proceso en Scrum

Figura 6: Reuniones, Sprint y Valores en Scrum

1.2 XP (Programación Extrema)

XP (del inglés "eXtremeProgramming") es una metodología ágil exclusiva para el desarrollo de software.
Al igual que Scrum, considera que los cambios durante el proyecto serán frecuentes, tanto, que se puede llegar a trabajar en iteraciones de 1 sólo día, con entregas y despliegues de los resultados a diario, incluso en períodos más breves de tiempo.
Contempla prácticas de ingeniería, valores, actividades y roles.
13 prácticas de ingeniería:

- Equipo compacto
- Juegos de planificación
- Pruebas de usuario
- Entregas (releases) pequeñas
- Diseño simple
- Programación por parejas
- Refactorización
- Desarrollo dirigido por las pruebas
- Integración continua
- Propiedad colectiva del código
- Estándares de codificación
- Metáfora del sistema
- Ritmo sostenible

5 valores

- Simplicidad
- Comunicación
- Feedback
- Motivación
- Respeto

4 actividades

- Codificar
- Probar
- Escuchar
- Diseñar

4 roles

- Líder ágil o coach
- Cliente
- Programador (desarrollador)
- Tester

1.2.1. Prácticas de ingeniería

Equipo compacto

Dentro de XP, el "cliente" no es el que paga la factura, sino el que realmente utiliza el sistema. XP dice que el cliente debe estar accesible en todo momento y disponible para preguntas. Por ejemplo, el equipo que desarrolla un sistema de administración financiera debe incluir o tener accesible un administrador financiero.

Juegos de planificación

El proceso principal de planificación dentro de la programación extrema se llama el Juego de Planificación.
El proceso de planificación se divide en dos partes:

- Planificación de la release
- Planificación de la iteración

Pruebas de usuario

Las pruebas de aceptación son propiedad del Cliente y definidos por dicha figura.
Debemos incluir un punto como "pasar las pruebas del cliente" en la definición de "hecho".

Entregas (releases) pequeñas

La entrega se realiza a través de frecuentes lanzamientos de funcionalidad en producción creando valor real de negocio. Los pequeños lanzamientos ayudan al cliente a ganar confianza en el progreso del proyecto.
El cliente ahora puede llegar a sus propuestas futuras sobre el proyecto basado en la experiencia real, sobre un producto tangible, usable (no sobre documentación, como ocurre en los proyectos no ágiles).

Diseño simple

Los programadores deben tomar un enfoque al diseño del software del tipo "lo simple siempre es lo mejor".

Cada vez que se escribe un nuevo código, el autor debería preguntarse si existe una manera más sencilla de introducir la misma funcionalidad. Si la respuesta es sí, la solución más simple debe ser elegida.

Programación por parejas

También llamada "programación por pares". Todo el código es producido por dos personas que programan en una estación de trabajo (un PC).Un programador tiene control sobre la estación de trabajo y está pensando sobre todo en la codificación en detalle. El otro programador está más centrado en el panorama general y está continuamente revisando el código que está siendo producido por el primer programador.

Los programadores intercambian estos papeles varias veces al día.

Refactorización

Refactorización es el proceso de cambiar un sistema de software de tal manera que no altera el comportamiento externo del código pero mejora su estructura interna.

Desarrollo dirigido por las pruebas

Dentro de XP, las pruebas unitarias se escriben antes de codificar el código final.

Este enfoque pretende estimular al programador a pensar en las condiciones en las que su código podría fallar.

Una sección de código fuente está terminada ("done", o hecha) cuando el programador no puede llegar a cualquier otra condición en la que el código pueda fallar.

Integración continua

El equipo de desarrollo siempre debe estar trabajando en la última versión del software.

Los miembros del equipo deben intentar cargar su versión actual en el repositorio de código cada pocas horas.

La integración continua evitará retrasos más adelante en el ciclo del proyecto, causado por problemas de integración.

Propiedad colectiva del código

Todo el mundo es responsable de todo el código; esto, a su vez, significa que todo el mundo está autorizado a cambiar cualquier parte del código.

Acelera el proceso de desarrollo, porque si se produce un error en el código cualquier programador puede arreglarlo, pero existe el riesgo de errores introducidos por los programadores que no prevén ciertas dependencias.

Las pruebas unitarias bien definidas solucionan este problema: si las dependencias imprevistas crean errores, cuando se ejecuten pruebas unitarias, se mostrarán fallos.

Estándares de codificación

El estándar de codificación es un conjunto acordado de reglas que todo el equipo de desarrollo acuerda seguir durante todo el proyecto. El estándar especifica un estilo y un formato consistentes para el código fuente, dentro del lenguaje de programación escogido, así como varias construcciones y patrones de programación que se deben evitar para reducir la probabilidad de defectos.

Metáfora del sistema

La metáfora del sistema es una historia que todo el mundo - clientes, programadores y administradores - puede decir acerca de cómo funciona el sistema.

Es un concepto de nomenclatura para las clases y los métodos que deberían facilitar a un miembro del equipo adivinar la funcionalidad de una clase / método particular, sólo a partir de su nombre.

Para cada clase u operación, la funcionalidad es obvia para todo el equipo.

Ritmo sostenible

El concepto es que los programadores o desarrolladores de software no deben trabajar más de 40 horas por semana, y si hay horas extras una semana, que la próxima semana no debe incluir más horas extras.

Este concepto incluye la idea de que las personas realizan mejor y más creativamente su trabajo si están descansados.

1.2.2. Valores en XP

Simplicidad

Recordemos que el diseño simple es una de las premisas que se deben seguir.

También simplicidad en el código, puesto que se aplica la refactorización.

Y simplicidad en la documentación: si el código está bien comentado, el diseño se mantiene simple, y está refactorizado, no será necesario escribir documentación adicional para su comprensión.

Comunicación

El código simple y bien documento comunica a los programadores muy bien qué funcionalidad persigue.

Las pruebas unitarias comunican por qué se ha elegido ese diseño para el código, y desde luego si funciona correctamente.

Y la comunicación con el cliente es fluida, porque éste forma parte del equipo de trabajo.

Feedback

La retroalimentación por parte del cliente es inmediata, puesto que forma parte del equipo de trabajo. Esto se refuerza porque las entregas se realizan continuamente en periodos muy cortos de tiempo.

Las pruebas dan feedback en tiempo real sobre el estado de salud del código.

Motivación

Motivación y valentía para centrarse sólo en lo necesario para la entrega de hoy, y no para la de mañana.

Hay que cumplir entregas todo el tiempo, aunque sean pequeñas. Hay que estar motivado para dar la mejor solución simple para la siguiente entrega.

Y ser persistente para mantener el ritmo sostenible pero necesario para cumplir con las entregas.

Respeto

Los programadores se respetan mutuamente porque nadie puede escribir código que haga fallar código o pruebas de otra persona.

El cliente respeta a los programadores, y los programadores al cliente, puesto que su comunicación es fluida, constante y debe haber entendimiento entre todos.

1.2.3. Actividades en XP

Codificar

El código se construye siguiendo las prácticas de ingeniería recomendadas (programación por parejas, refactorización, estándar de codificación, etc.)

Probar

Es fundamental que el desarrollo sea dirigido por las pruebas, por lo tanto codificar y probar son actividades que van de la mano.

Escuchar

Puesto que se trabaja por parejas, y también conjuntamente con el cliente, las personas del equipo tienen que aprender a escuchar a los demás y a compartir su información y opinión profesional.

Diseñar

Se contempla en todo momento el diseño simple, la refactorización, el respeto al estándar de codificación, la documentación en el código... estas actividades forman parte del diseño del software.

1.2.4. Roles en XP

Líder ágil o coach

Responsable del proceso global, conocedor de la metodología, y facilitador del trabajo. Guía a los miembros del equipo para seguir el proceso correctamente.

Cliente

Escribe los requisitos (las historias de usuario) y comprueba los criterios de aceptación de los mismos. Decide la prioridad, según criterios de negocio, de los requisitos para su implementación y entrega.

Programador (desarrollador)

Construye el código y las pruebas unitarias que resuelven los requisitos. En este rol se incluirían todos los perfiles necesarios para ello (programador, analistas programadores, diseñador, maquetador.... excepto pruebas funcionales).

Tester

Ayuda al cliente a escribir las pruebas funcionales. Gestiona y utiliza las herramientas para las pruebas funcionales.

1.3 Kanban

Se utiliza un "Kanban" o "TaskBoard" para buscar cambios de estado en el trabajo, que reflejen el progreso y el trabajo en curso.

Esta metodología consiste en la organización del trabajo diario en base a un panel de tareas como el siguiente:

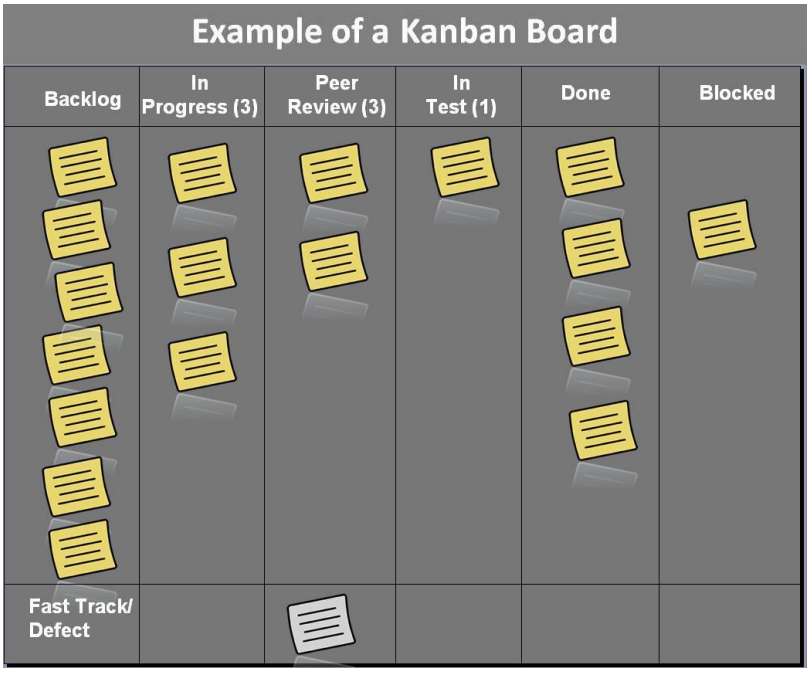

Figura 7: Ejemplo 1 de un Kanban

El Método Kanban no te pide que cambies tu proceso.No propone cambios en las prácticas de ingeniería ni una nueva definición de proceso o estilo de trabajo.

No existealgo como el "Kanban Software Development Process" o el "Kanban Project Management Method".

Kanban es una traducción libre del japonés de “cartas”.Es una parte fundamental del flujo tenso (*pull*). Se diseña para evitar la sobreproducción y para asegurarse de que los componentes pasan de un sub-proceso al siguiente en el orden adecuado. De este modo se diseña un

sistema de relleno que controla las cantidades producidas. Los componentes se reponen únicamente cuando sea necesario y en la cantidad adecuada.

En lugar de utilizar *kanban* diseñados específicamente para ello, se pueden poner en marcha otros sistemas reutilizables tales como contenedores, palets o bandas codificadas (o coloreadas) que designan materiales específicos. Al dejar el embalaje para el suministrador en una ubicación específica implica una solicitud para rellenar con el componente adecuado, sin necesidad de que se produzca ninguna comunicación oral o escrita.

El flujo tenso del producto desde aguas arriba se indica mediante un *kanban* de retirada (*withdrawal*). El flujo tenso del cliente retira componentes del “supermercado”; éste se define como un lugar de capacidad limitada para almacenar el producto proveniente del proceso de suministro. El supermercado se rellena emitiendo un *kanban* de producción cuando el inventario es demasiado bajo. Este *kanban* de producción da la orden adecuada al proceso de suministro para producir más componentes. El proceso de suministro emite las unidades necesarias para reponer lo extraído. Este método evita la sobreproducción, pero permite un inventario rígido que se sitúa entre los procesos de suministro y del cliente.

La alternativa al *kanban* es producir anticipándose a las necesidades basándose en predicciones, caso habitual en los sistemas *push*. Estos sistemas tienden a incrementar la cantidad de pérdidas (por ejemplo, largos tiempos de espera o inventarios excesivamente grandes) dado que están basados en la estimación e incluyen factores adicionales para tener en cuenta la incertidumbre. La incertidumbre puede manifestarse en mayor o menor medida en un proyecto; en ese caso los sistemas *pull* están mejor preparados para adaptarse que los sistemas *push*.

Se debe poner como objetivo pequeños cambios incrementales. El equipo de trabajo debe estar de acuerdo en que sus circunstancias actuales justifican un enfoque evolutivo para el desarrollo. Sin el acuerdo de que un enfoque evolutivo e incremental es el camino correcto hacia adelante, entonces no habrá el ambiente adecuado para una iniciativa Kanban.
Cada miembro del equipo es un líder, en el sentido de que se espera que tenga iniciativa para autogestionar su trabajo, asignarse tareas y compartir esta información con los otros miembros del equipo.
Hay que buscar el límite de WIP (Work In Progress, o "trabajo en curso") que en última instancia estimula las ideas sobre problemas de proceso que se pueden solventar o dónde se

puede mejorar. Las cosas que impiden el flujo o introducen perturbaciones en el flujo de trabajo, son las que limitan el WIP.

El trabajo en progreso en cada estado en el flujo es limitado; es decir, no debemos tener más de un número determinado de tareas en cada estado.

En un Kanban, el estado del trabajo es siempre claramente visible:

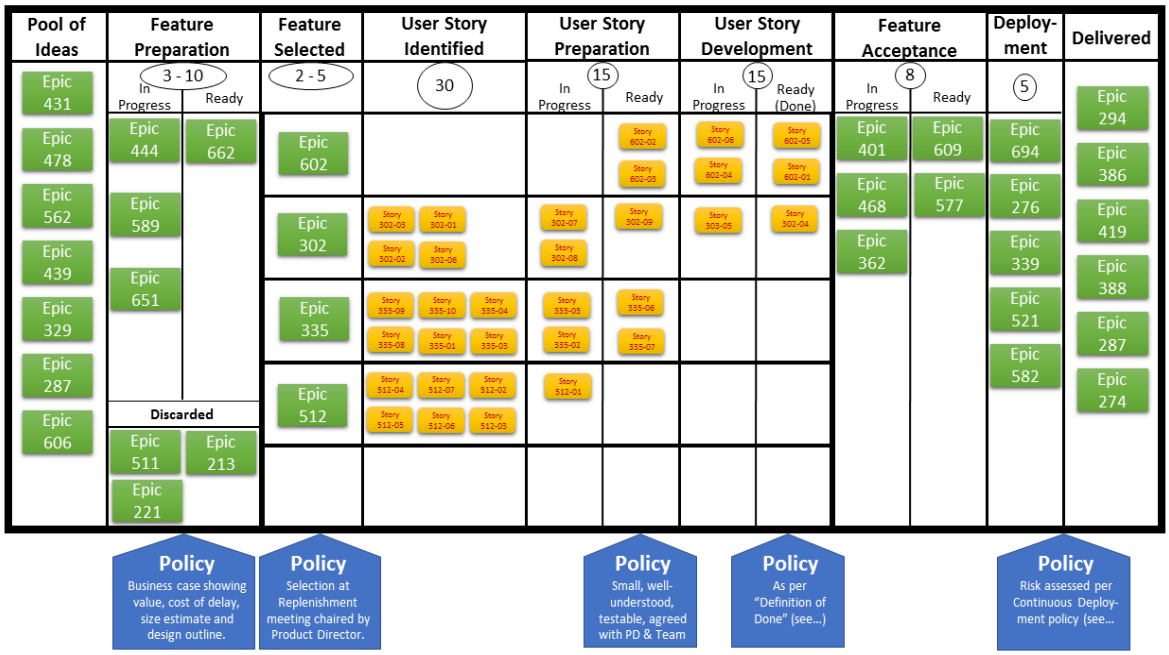

Figura 8: Ejemplo 2 de un Kanban

El proceso debe estar claro para todos: Sin una comprensión explícita de cómo funcionan las cosas y cómo se hace el trabajo, cualquier discusión de los problemas tiende a ser emocional, anecdótica y subjetiva. Con un entendimiento explícito es posible pasar a una discusión más racional, empírica y objetiva de las cuestiones. Esto es más probable que facilite el consenso sobre las sugerencias de mejora.

Cuando una tarea es definitivamente "hecha", se debe entregar o mostrar al cliente tan pronto como sea posible.

Podremos ver la información y el paso generado de una actividad a la siguiente.

Se monitoriza el trabajo observando el flujo de elementos de trabajo a través de cada estado (las columnas) en el panel. Nos interesa la velocidad y la fluidez de ese movimiento (no hay avances bruscos y grandes, sino pequeños y continuos).

Idealmente queremos flujo rápido y fluido.

Flujo rápido y fluido significa que nuestro sistema está creando valor rápidamente, lo cual minimiza el riesgo y evita el coste (de oportunidad) del retraso, y también lo hace de manera predecible.

Veamos algunos ejemplos de Kanban en proyectos reales:

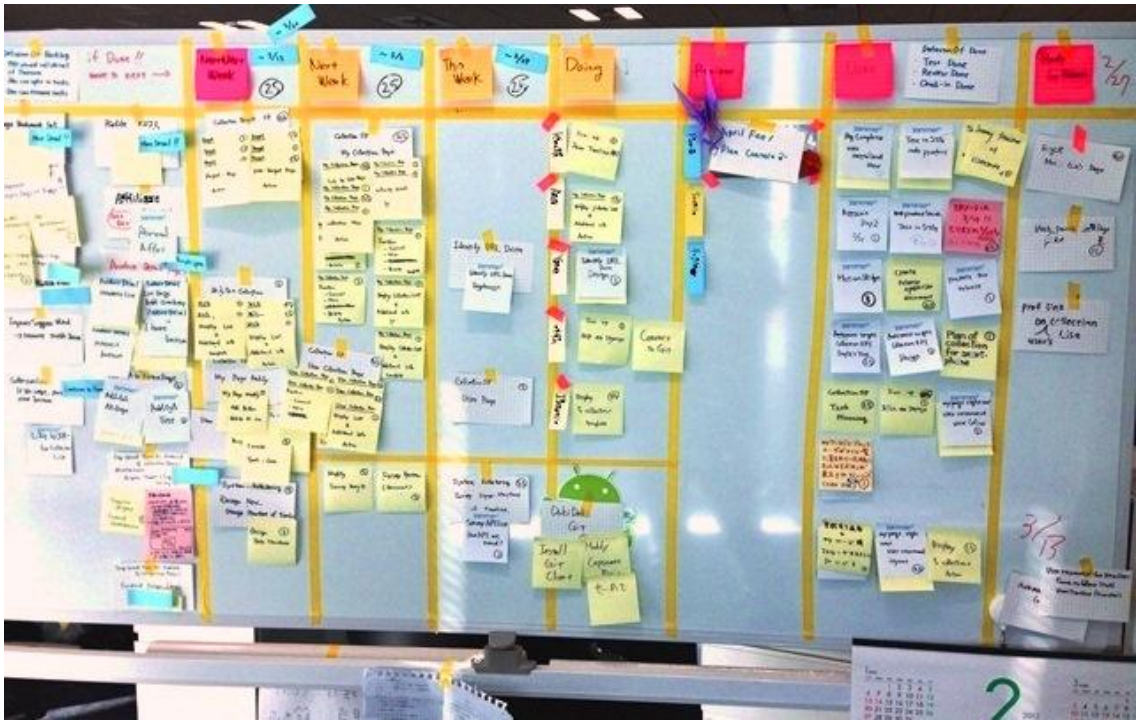

Figura 9: Ejemplo 3 de un Kanban

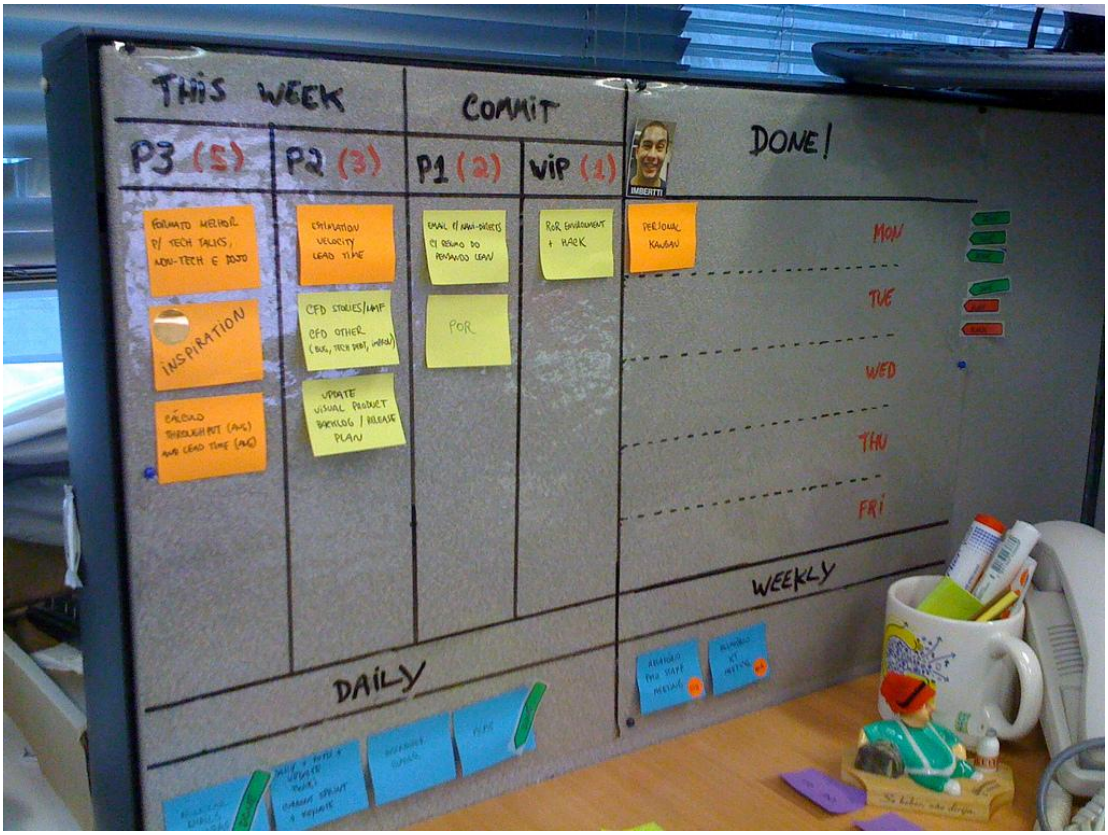

Figura 10: Ejemplo 4 de un Kanban

Figura 11: Ejemplo 5 de un Kanban

1.4 Lean

Más que una metodología, Lean es una filosofía o enfoque de trabajo.
Su origen es el Modelo de Manufactura Lean, es decir aplicado a procesos de fabricación y producción "pull": sólo se produce el producto que el cliente necesita, justo cuando lo necesita.
La idea es muy simple: Se centra en hacer obvio lo que añade valor al reducir todo lo demás. La idea central es maximizar el valor del cliente y minimizar el desperdicio. Simplemente, lean significa crear más valor para los clientes con menos recursos. Una organización “lean” entiende el valor del cliente y enfoca sus procesos claves para aumentarlo continuamente.

Los principios que definen esta filosofía de trabajo son:

- Eliminar residuos
- Ampliar el aprendizaje
- Decidir lo más tarde posible
- Entregar lo más rápido posible
- Capacitar al equipo
- Ver el conjunto

El objetivo final es proporcionar valor perfecto al cliente a través de un proceso de creación de valor perfecto que tiene cero residuos.

El término Lean se aplica por el hecho de una menor utilización de todo: Menor esfuerzo humano, menor espacio de fabricación, menor inversión en herramientas, menos horas de ingeniería. También requiere menos inventario en el sitio que se refleja en menores defectos y mayor variedad de productos.

Lean proporciona mejor calidad, a un menor coste y con los plazos de entrega más cortos a través de la eliminación de desperdicio, esto son improductividades o actividades que no añaden valor.

Puesto que hablamos de no producir desperdicio, ¿qué es el desperdicio? veamos que clasifican en 8 los desperdicios que causan las interrupciones del flujo de valor en el proceso de producción:

- **Sobreproducción**: Producción de mayores cantidades o más pronto de lo necesario, planos adicionales no esenciales o la utilización de equipamiento de mayor calidad

que el necesario. Dar más información de la necesaria, copias extra, informes que nadie leerá.

- **Esperas o tiempo de inactividad**: Esperas, interrupciones o tiempo inactivo debido a falta de datos, información u órdenes, planos o material. Esperas por la actividad precedente, resultados de laboratorio, escasez de equipos y otros motivos que contribuyen a inactividades. Esperar correos, faxes, materiales, el trabajo de un compañero.
- **Transporte innecesario**: Relacionado con la mala distribución y planificación del movimiento interno de los recursos. Transportar documentos, materiales, equipos.
- **Sobreprocesamiento**: Procesos que causan exceso de materia prima, equipos, energía, etc. Papeleos, formularios anticuados, software inadecuado.
- **Exceso de inventario**: Inventarios excesivos, innecesarios o antes de tiempo que pueden producir pérdidas de material, personal adicional para el control de ese exceso y costes por la compra anticipada. Materiales apilados, archivos abiertos, suministros esperando, emails sin leer.
- **Exceso de movimiento**. Buscar archivos, información, mirar manuales, mirar catálogos.
- **Defectos de calidad**: Errores en diseño, mediciones y planos, uso de métodos de trabajo incorrectos, mano de obra poco cualificada. El resultado es un producto de mala calidad y la insatisfacción del cliente. En definitiva, defectos.
- **Talento (creatividad del empleado desaprovechada)**: Se pierden ideas, aptitudes, mejoras al tener mano de obra poco cualificada, poco formada, mal informada y con falta de motivación.

Pero resulta difícil identificar los desperdicios debido a los siguientes motivos:

- La improductividad suele estar oculta y cuesta identificar las tareas que son productivas de las que no lo son.
- Muchas de las organizaciones terminan habituándose al desperdicio, trabajando con ellos y creando parches para tapar el problema.
- La falta de formación a los trabajadores, directivos y cargos intermedios hace que no sean capaces de identificarlos.
- Al no ser posible cuantificar el desperdicio, no es posible concienciarnos del dinero malgastado.

- El sistema productivo que habitualmente se utiliza se centra en una mejora individual del rendimiento de tareas individuales en lugar de tener una visión más general de todo el proceso.
- En general nadie sabe cómo afecta su trabajo en los demás y no suele haber un responsable de todo el flujo de valor.

Como consecuencia de esto, se puede decir que las empresas que mejor se han adaptado y superado las crisis son aquellas que son capaces de aprender de sus errores y consideran la empresa como un lugar donde aprender a mejorar el servicio y producto que ofrecen. Las habilidades que desarrollan este tipo de empresas son:

1. Aprender a ver y hacer visibles los problemas.
2. Una vez identificado el problema saben atacarlo inmediatamente justo dónde y cuando ocurren.
3. Lo aprendido es compartido a lo largo de toda la organización, para evitar que se repita.
4. Aprenden a liderar las tres capacidades mencionadas.

Los principios básicos del pensamiento Lean son:

1º Valor:

El valor es el punto de partida del pensamiento Lean. Crear valor para el cliente significa saber qué quiere el cliente. Se trata de realizar el diseño del producto y el proceso de fabricación basándose en el punto de vista del cliente.

Se distinguen dos tipos de clientes en una empresa Lean: El cliente externo que es el usuario o consumidor, es el que define el valor del producto o servicio. Por otro lado, está el cliente interno que es aquel que dentro del flujo de valor recibe material o información por parte de un proceso que se encuentra aguas arriba del flujo de valor.

2º ValueStream (Cadena o flujo de valor)

La cadena de valor son todas las actividades que se necesitan para transformar los materiales y la información en un producto o servicio que entregamos al cliente. Se identifican flujos de valor que abarcan toda la cadena de proveedores y clientes o flujos de valor más reducidos, a nivel de células de trabajo. Generalmente existe un flujo de valor por cada familia de productos o servicios que entrega la empresa. Para identificar el flujo de valor hay que dejar claro dónde empieza y dónde acaba este.

3º Flujo

Después de conocer el valor del cliente, definir la cadena de valor y eliminarlos desperdicios evidentes, lo que se pretende es que las operaciones creadoras de valor que quedan fluyan.
Las actividades que añaden valor son una fracción muy pequeña del total. El Lean consiste en identificar y eliminar la mayor parte de las actividades que no añaden un valor para la mejora de la productividad para entregarle más valor al cliente. La eliminación de desperdicio forma parte de crear ese flujo de la cadena de valor.

4º Sistema Pull

Se trata de un sistema productivo en el que son las actividades aguas abajo las que tiran de las actividades aguas arriba dando señales de sus necesidades mediante tarjetas Kanban. Son ellas las que piden material, en la cantidad, el lugar y en el momento necesario. En cierto modo no se produce nada hasta que no lo señala el proceso del cliente aguas abajo. Es el cliente quien tira de la demanda y no el fabricante el que empuja los productos hacia el cliente.
Este sistema Pull es el fundamento del *Just-in-Time* con el que se elimina el exceso de inventario y la sobreproducción. Lo opuesto era el sistema tradicional o Push, que se basaba en el almacenamiento de productos a gran escala según la demanda prevista, dirigiéndolos hacia aguas abajo o hacia el almacén de productos terminados.

5º Perfección

La idea es conseguir una producción de puro valor, tal y como lo pide el cliente, sin ninguna muda o desperdicio. Para conseguir esto se necesitan las 3 herramientas fundamentales ce la cultura Lean: El *Kaizen* o mejora continua, la estandarización de procesos y un plan de acción o PDCA.
No existe un límite para la mejora continua, se tiende a ofrecer un producto o servicio cada vez más cerca de lo que el cliente desea.

6º Transparencia

La transparencia es algo muy importante para todos, cuanto mayor sea la información a la que tenemos acceso mejores métodos se pueden descubrir para la creación de valor. A su vez se produce un feedback muy positivo e instantáneo para los empleados que hacen mejoras. Esto es la clave del Lean, un proceso de continua mejora.
Al proporcionar información acerca de los procesos de producción, se da alas para tomar acciones en el proceso.

7º Capacitación

Se les exige a todos los participantes de la cadena o flujo de valor, una atención continua para mantener el flujo y eliminar el desperdicio. Para que el resultado sea satisfactorio se debe informar de manera correcta a todos los empleados, así como darles autoridad para solucionar los problemas y trabajar en una continua mejora.

Los trabajadores están todos ellos capacitados y trabajan de forma colaborativa con sus compañeros a través de toda la cadena de valor.

1.4.1. Lean Start Up

Se trata de extender la metodología o filosofía Lean, al lanzamiento de nuevas empresas al mercado.

En lugar de hablar de eliminar el desperdicio en el código fuente, por ejemplo, hablamos de eliminar los stocks intermedios en un proceso de distribución y comercialización.

Sus principios son los mismos que el método Lean clásico, resaltando todas aquellas actividades que aportan valor al negocio, y tratando de minimizar, o incluso eliminar, todas aquéllas que ralenticen o no aporten valor real.

La idea es ir validando cada aprendizaje de forma continua, experimentando con nuevas ideas en el negocio real y no de forma teórica; e iterando sobre este estilo de funcionamiento.

Figura 12: Lean Start Up

El gran objetivo es minimizar el riesgo en el lanzamiento de nuevos productos al mercado, o incluso en la creación de nuevas empresas, aprendiendo del cliente cuando más rápido y barato mejor.

1.5 Relación entre metodologías ágiles

A menudo no tienen por qué elegirse unas metodologías u otras de forma separada.
Es decir, no hay que pensar en términos de "Lean, o Scrum", por ejemplo. Scrum es una metodología concreta, que nos dice qué reuniones debemos mantener, cuánto duran, qué roles hay en el proyecto, qué artefactos... Pero todo esto es una forma de hacer específicos y aplicables los principios del agilismo (entregas continuas, comunicación fluida, simplicidad...). Como por ejemplo, también propone XP.
A la vez, tanto en XP como Scrum, el equipo de trabajo puede (y debe, porque es muy buena práctica) organizar su trabajo utilizando Kanban, para ver con facilidad las tareas en curso, si hay alguna columna donde se acumulan muchas tareas (eso es signo de algún problema), si el ritmo de avance es el correcto...
Más aún: en cualquiera de estas combinaciones, aplica la filosofía Lean: no hacer tareas que no aporten valor; maximizar el valor del producto final; no generar desperdicio; mejorar de forma continua...

Si la "unión hace la fuerza", en el caso de las metodologías ágiles se aplica perfectamente.

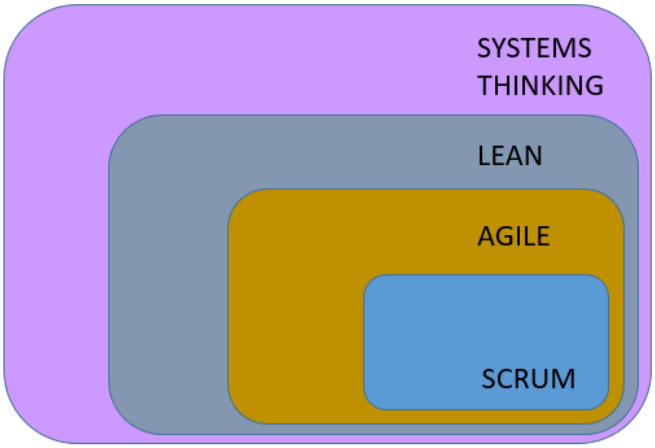

Figura 12: Unión de metodologías y marco ágil

A su vez, todo este enfoque mental está dentro de lo que podemos denominar "pensamiento sistémico", es decir, pensar en global, en dar la mejor solución al problema completo, eligiendo y afinando las mejores prácticas para ello.

Por ejemplo, un proyecto se organiza en Sprints de 2 semanas, se llevan a cabo las ceremonias de Scrum, se utiliza un Kanban para la gestión de tareas, se aplican algunas prácticas de ingeniería de XP (programación por pares, desarrollo dirigido por las pruebas, e integración continua), generando en todo ello el menor desperdicio posible y aumentando el valor del resultado del proyecto.

2. REFERENCIAS

https://youtu.be/PlLHc60egiQ

Vídeo explicativo que narra toda la teoría de Scrum completa en 7 minutos.