Metodologías ágiles

Ciclo: DAW
Módulo: ENDE
Alumno: Santiago Alarcón

ÍNDICE

ÍNDICE 2
INTRODUCCIÓN 3
MANIFIESTO 3
ALGUNAS PRÁCTICAS COMUNES 3
CONCLUSIÓN 8
LINKS VISITADOS Y BIBLIOGRAFÍA 9

INTRODUCCIÓN

A falta de una integración real de la actividad informática en la administración y dirección de empresas y como alternativa a los modelos de integración tecnológico-empresariales clásicos, los métodos ágiles son una de las propuestas más aceptadas en el campo del desarrollo software. Aquí presento un resumen acompañado de recursos que considero pueden ser útiles.

MANIFIESTO

Con el desarrollo de la informática en los años 60 los conceptos de software y hardware empiezan a tener una relación de liminalidad. A finales de los 60, el coste de desarrollo software había crecido exponencialmente al contrario que el coste de la producción de hardware (sin entrar aquí en si esta interpretación económica es ajena o no a la separación de los dos conceptos). El problema de una producción de software pobre o incompleta de forma generalizada ha recibido el nombre de crisis del software, en respuesta de la cual a mediados de los 90 se empieza a asentar una definición sobre las metodologías de desarrollo software ágiles, y aparecen una serie de modelos cuya autoridad se documenta en *El manifiesto Ágil* de 2001. Algunos de sus firmantes formarían poco después la “Alianza Ágil”.

Este manifiesto se fundamenta en cuatro puntos principales:

- Valorar a los individuos y sus interacciones, frente procesos y herramientas
- Valorar más el software (producto) que funciona, que una documentación exhaustiva.
- Valorar más la colaboración con el cliente que la negociación de un contrato.
- Valorar más la respuesta al cambio que el seguimiento de un plan.

El manifiesto traducido al español puede ser consultado en www.agilemanifesto.org/.

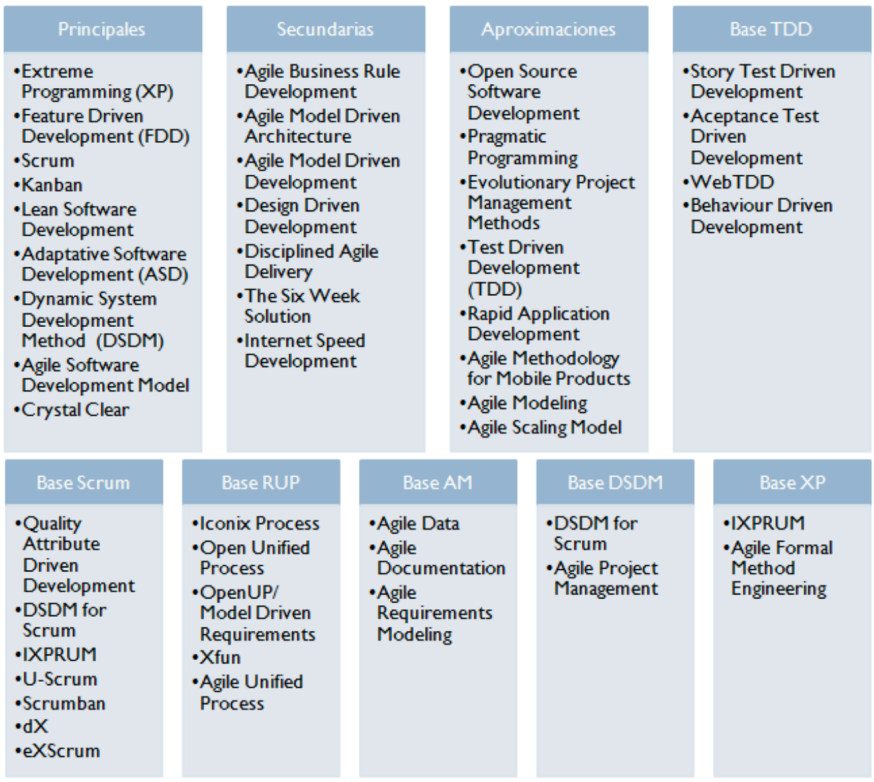
ALGUNAS PRÁCTICAS COMUNES

Para cumplir los valores descritos en el punto anterior se siguen doce principios marcadores del ciclo de vida de un desarrollo ágil frente a uno tradicional:

- La mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software (productos) con valor.
- Se acepta que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos Ágiles aprovechan el cambio para proporcionar una ventaja competitiva al cliente.
- Se entrega software funcional con frecuencia, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible.
- Los responsables del negocio y los desarrolladores (miembros del equipo) trabajan juntos de forma cotidiana durante todo el proyecto
- Los proyectos se desarrollan en torno a individuos motivados, idealmente provistos del entorno y el apoyo que necesiten.

- El diálogo cara a cara es el método más eficiente y efectivo para comunicar información dentro de un equipo de desarrollo.
- La principal medida de progreso es el software que funciona.
- Los procesos ágiles promueven un desarrollo sostenible. Los promotores, desarrolladores y usuarios deberían ser capaces de mantener una paz constante.
- La agilidad mejora con la atención continua a la calidad técnica y al buen diseño.
- La simplicidad es algo esencial.
- Las mejores arquitecturas, requisitos y diseños surgen de los equipos que se auto organizan.
- Cada cierto tiempo el equipo debe reflexionar sobre cómo ser más efectivo y según estas reflexiones ajustar su comportamiento.

Para terminar es importante señalar que no hay un único procedimiento de implementación de las metodologías ágiles al desarrollo software y que ya existe una literatura extensa sobre los retos y los atajos principales para una adaptación exitosa. Así, los recursos prácticos de la conclusión serán entero dependientes del caso concreto al que aplican.



EJEMPLOS DE METODOLOGÍAS ÁGILES

EXtreme Programming: Es un método ágil para el desarrollo de software muy útil a la hora de abordar proyectos con requisitos vagos o cambiantes con equipos de desarrollo pequeños o medianos.

Se realizarán pruebas automáticas de forma constante para poder detectar los fallos rápidamente y el trabajo se focalizará en una serie de valores:

- Comunicación
- Simplicidad
- Feedback
- Valentía
- Respeto

Para concretar estos valores se describen en 2004 (por Kent Beck) una serie de prácticas primarias y prácticas corolario, que copio aquí a modo de recordatorio.

Prácticas primarias:

- Trabajar con historias de usuario
- Realizar ciclos semanales de desarrollo
- Organizar revisiones trimestrales
- Trabajar con holgura
- El equipo debe sentarse junto
- El equipo debe ser completo
- Tener información sobre el proyecto en el puesto de trabajo
- Mantener la energía en el trabajo o un ritmo sostenible
- Realizar con frecuencia la programación en parejas
- Diseño incremental
- Realizar las pruebas antes de programar
- Construir en diez minutos
- Integración continua

Las prácticas corolario son:

- Participación real de los clientes
- Despliegue incremental
- Negocie el alcance3 del contrato
- Pague por funcionalidad
- Continuidad de los equipos
- Reducir los equipos
- Análisis de las causas
- Código y pruebas
- Código compartido
- Código base único

- Despliegue diario.

Crystal Methodologies:

Esta metodología fue impulsada por uno de los padres del Manifiesto Agil, Alistair Cockburn, al definir y clasificar una serie de políticas dependientes del tamaño del equipo: Crystal Clear (3 a 8 miembros), Cristal Yello (10 a 20 miembros), Crystal Orange (25 a 50 miembros), Crystal Red (50 a 100 miembros) y Crystal Blrealizanue (más de 100 miembros). Cada política tiene una definición específica y una serie de características asociadas. Por ejemplo para el caso de crystal Clear: liberación frecuente de funcionalidad, mejora reflexiva, comunicación osmótica, seguridad personal, atención, fácil acceso para usuarios expertos y requisitos para el entorno técnico.

Dynamic Software Development Method (DSDM):

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

Sus principales características son interacción con el usuario, poder del equipo de desarrollo, liberaciones frecuentes de funcionalidad, regirse siguiendo las necesidades del negocio, desarrollo iterativo e incremental, adaptación a los cambios reversibles, fijar el nivel de alcance al comienzo del proyecto, pruebas durante todo el desarrollo y eficiente y efectiva comunicación.

Aunque como en todos los casos de metodologías ágiles las diferencias en las líneas generales son sutiles, frente a los métodos tradicionales en los que se fijaban unos requisitos se estimaba en función de ellos, se estimaba tanto los recursos como la fecha de entrega el DSDM propone fijar los recursos destinados a un producto y la fecha de entrega y hacer una estimación del a funcionalidad que se entregará (se sabe cuando se va a entregar algo valioso al cliente pero a priori no se sabía qué es exactamente lo que se va a entregar).

Feature Driven Development:

Combina el desarrollo dirigido por modelos con el desarrollo ágil Sus puntos clave son el trabajo en iteraciones, control continuo, la calidad de lo creado y entregas frecuentes para poder realizar un seguimiento continuo en colaboración con el cliente y poder así incorporar sus necesidades en el producto con frecuencia.

Los pasos secuenciales de esta metodología son:

- Crear un modelo global - llegar a una descripción general del sistema
- Crear una lista de funcionalidades- El modelo global se plasma en una única lista de necesidades o funcionalidades a cubrir.
- Planear por funcionalidades- Se establece una prioridad en las funcionalidades y las dependencias entre ellas.
- Diseñar y construir por funcionalidad- Se procede a diseñar y construir las funcionalidades de forma iterativa. En cada iteración se seleccionará un conjunto de funcionalidades y sus ciclos deben oscilar entre algunos días y dos semanas.

Kanban

La aplicación de Kanban data de una fecha tan reciente como 2004. Es un método recomendado cuando las planificaciones o estimaciones de un equipo se alarguen demasiado y dejen de ser productivas. De forma simplificada se siguen los siguientes pasos:

- Visualizar el flujo de todo el trabajo: En un panel debe estar representado todo el flujo de trabajo que hay que realizar en el proyecto desde el principio hasta el último momento. Kanban destaca por su enfoque visual.
- Limitar el trabajo en curso- Es imprescindible poner un límite al número de ítems permitidos en cada columna yd e esta forma evitar colapsos, cuellos de botella y eliminar cuanto antes los impedimentos que surjan y que impidan trabajar con un ritmo sostenible
- Medir el tiempo empleado en completar un ciclo completo- Esta práctica ayuda también a ser predecible y poder hacer una estimación previa de cuánto tiempo se empleara en completar cada ítem.

La gestión visual del proyecto y la accesibilidad de la información hacen que Kanban no solo se aplique a proyectos de desarrollo de software sino que sea valioso para otros departamentos o proyectos diferentes.

SCRUM

Es una de las metodologías ágiles más difundidas. Se basa en los siguientes principios:

- Inspección y adaptación: Se trabaja en iteraciones llamadas Sprint que tienen una duración entre 1 y 4 semanas. Cada iteración termina con un producto a entregar (por ejemplo software ejecutable). Al final izar cada iteración, este producto se muestra al cliente para que opine sobre él. A continuación, el equipo se reunirá para analizar la manera en que está trabajando.
- Auto-organización y colaboración. El equipo se gestiona y organiza a sí mismo. Este nivel de libertad implica asumir una responsabilidad y un gran nivel de compromiso por parte de todos, lo que exige una alta colaboración y espíritu de equipo.
- Priorización- Como el resto de métodos ágiles, es crucial no perder tiempo y dinero en algo que no interese inmediatamente para el producto, esto es, requisitos perfectamente priorizados reflejando el valor del negocio.
- Mantener un latido- Hay que mantener una pauta de trabajo que ayude a los equipos a optimizar su trabajo y lo haga predecible.

Una de las principales características de Scrum es que en cada iteración todas las etapas de la creación de un producto se solapan, es decir, en cada Sprint se realizan la planificación, análisis, creación y comprobación de lo que se va a entregar al final del mismo. Así mismo se asignan una serie de roles (product owner, scrum master y equipo) y se busca apoyo en una serie de artefactos o definiciones:

- El product backlog: Contienen los requisitos del cliente priorizados y estimados
- El sprint Backlog: Es la sección de requisitos del product Backlog negociados para el Sprint y que se han descompuesto en tareas por el equipo
- El Burndown Chart: en esta gráfica se representa el trabajo pendiente del equipo.

Otra característica importante del método Scrum es la disciplina en las reuniones. Se recomienda una duración en el caso del Daily Meeting o reunión diaria de entre 10 y 15 minutos y para el resto de reuniones una hora Der unión por semana de iteración en el caso de Sprint Planning y Sprint Review y de aproximadamente una hora para la Retrospective. Estos términos en inglés son las fases del ciclo scrum que se pueden seguir en este esquema:

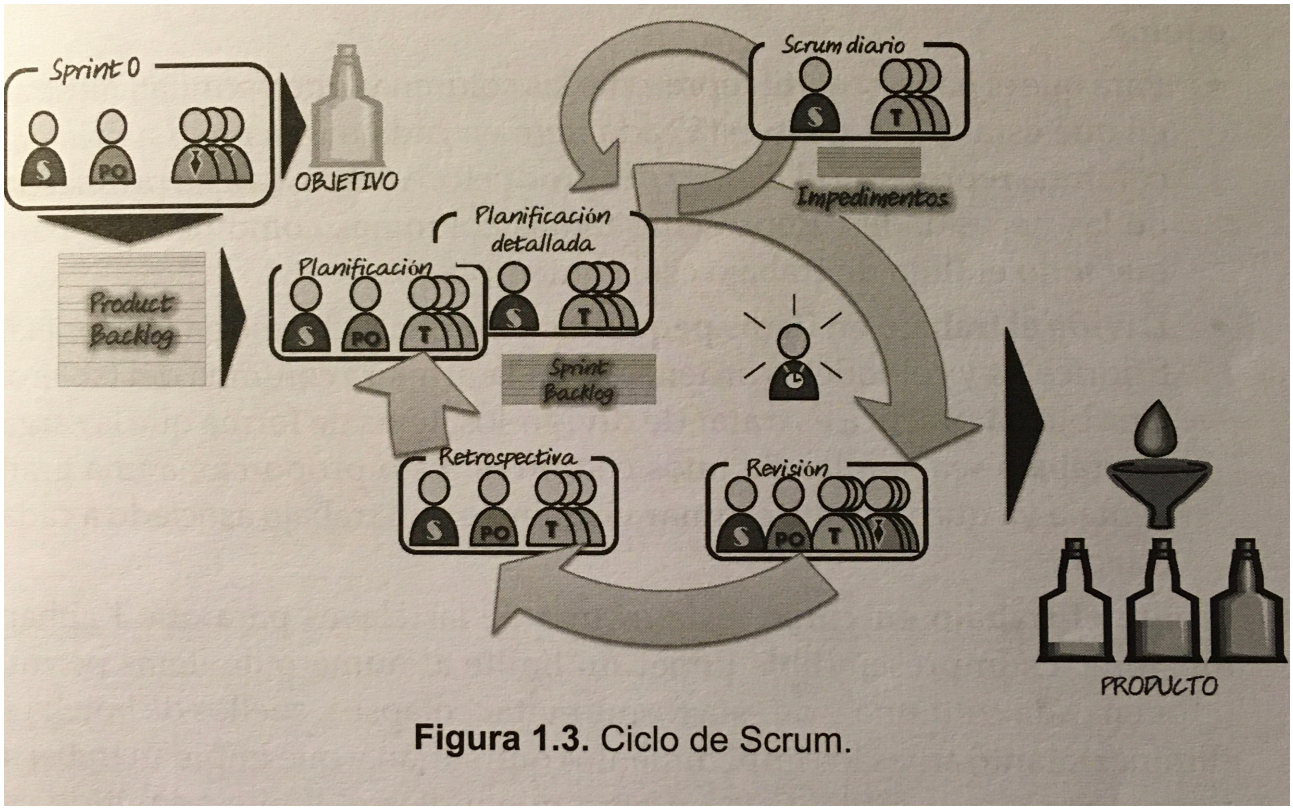


Figura 1.3. Ciclo de Scrum.

CONCLUSIÓN

Me interesa para este apartado a modo de cierre tener una lista de recursos en caso de necesitar aplicar estas metodologías a la empresa; dado que este escrito se enmarca en un ciclo superior, es improbable que en los primeros años tras su finalización tenga agencia para proponer o implementar metodologías ágiles a una empresa que llegue a tener más de 10 empleados, sumado a que las metodologías ágiles tienen su atractivo en lo mucho que dependen del problema a resolver; este resumen que sigue es algo estrictamente personal, memoria de las herramientas más interesantes de los links y libros consultados y que creo pueden serme útiles para el estudio del ciclo superior.

TDD

Una de las prácticas más conocidas dentro del eXtrem programming es el Test Driven Development (TDD). El TDD permite realizar de manera simultánea el diseño, las pruebas,

la arquitectura y la codificación. Los desarrolladores construyen el código a la vez que lo prueban. Por otro lado, los usuarios realizan pruebas de aceptación.

A esto se suma cómo buena práctica la programación en parejas, que consiste en que dos personas escriban el código en una única máquina; la revisión frecuente del código, la integración continua y la automatización de pruebas contribuyen al aseguramiento permanente de la calidad de lo que se está construyendo. La idea de la simultaneidad y la comunicación se reflejan en el principio de "flujo": ir creando con frecuencia pequeños incrementos de funcionalidad valiosa y para ello imbricar las fases de desarrollo.

ERPs y accesibilidad

Desde el punto de vista de la metodología Kanban, la principal fuente de desperdicio es la sobreproducción o producción innecesaria en una determinada etapa del flujo, es decir, hacer más de lo que se puede asimilar. Para una información inmediata sobre el estado del proyecto esta metodología propone la creación de tableros en los que cada columna se reserva para una etapa de flujo (solicitud, aprobación, análisis, construcción maquinación y liberación), con una serie de restricciones y permisos sobre la información contenida en el tablero.

Esto me recuerda el tiempo que habría ahorrado en el estudio del ciclo superior si hubiese dedicado los primeros días a familiarizarme con google.classroom. Uno de los problemas recurrentes que encuentro cuando leo sobre metodologías ágiles es la diferencia entre la implementación de éstas mediante un ERP, la propia trayectoria de la empresa, un equipo que ya se conoce, un tejido empresarial con facilidad de sinergia… etc y la problemática de su integración desde 0.

A mi nivel y para familiarizarme con la comunicación en empresa más allá del mail, WhatsApp y teléfono móvil he encontrado especialmente recomendadas para metodologías ágiles las aplicaciones web gratuitas slack.com, monday.com y discord.

Trabajo en equipo y herramientas

Siguiendo el punto anterior, creo que para una empresa pequeña lo más interesante de cara a implementar una metodología ágil desde un programa piloto hasta ser parte fundamental de la empresa sería definir claramente los roles a desempeñar por cada componente, como se hace en scrum, en vez de simplemente aceptar una jerarquía que se presupone y confiar en el trabajo en sí.

Así mismo, la simplicidad en cuanto a la planificación y las pruebas sobre el proyecto permite hacer un desarrollo a largo plazo complejo y que progrese con la idea del cliente. Esto puede ser contraintuitivo a primera vista pero también marca una línea de separación en el trabajo en sí (en el libro del que he sacado la mayor parte de la información "Métodos Ágiles y Scrum" se dedica solo un epígrafe a los "contratos ágiles" pero se dan algunas pinceladas sobre los retos y el estado de la situación laboral de los desarrolladores de software).

LINKS VISITADOS Y BIBLIOGRAFÍA

VVAA (Álvarez García, Alonso; de las Heras del Dedo, Rafael; Lasa Gómez, Carmen) (2012), *Métodos ágiles y Scrum,* editorial Anaya

Rodriguez González, Pilar (2008) *Estudio de la aplicación de metodologías ágiles para la evolución de productos software* (Tesis de master) Universidad politécnica de Madrid.

Santiago Quilligana, (2021, Enero), *Diferencias entre ciclos de vida y metodologías de desarrollo de software,* https://issuu.com/santiagoquilligana/docs/diferencias_entre_ciclos_de_vida_y_

Universidad tecnológica de Chalmers (2021, Enero), *readme.md about Agile Development Processes - ip4 vt2017,* https://github.com/oerich/EDA397

Tim Ferris (2021, Enero), *What's your start up bus count,* https://tim.blog/2011/06/07/whats-your-start-up-bus-count-7-myths-of-entrepreneurship-and-programming/

Fredy Humberto Vera-Rivera, Boris Rainero Pérez, (2021, Enero), *Agile development model of a cloud ERP for small and medium sized business smb of Norte de Santander,* https://www.researchgate.net/publication/310795974_AGILE_DEVELOPMENT_MODEL_OF_A_CLOUD_ERP_FOR_SMALL_AND_MEDIUM_SIZED_BUSINESS_SMB_OF_NORTE_DE_SANTANDER

Chris Kimble, (2021, Enero), *The Software Crisis*, http://www.chris-kimble.com/Courses/World_Med_MBA/Software_Crisis.html