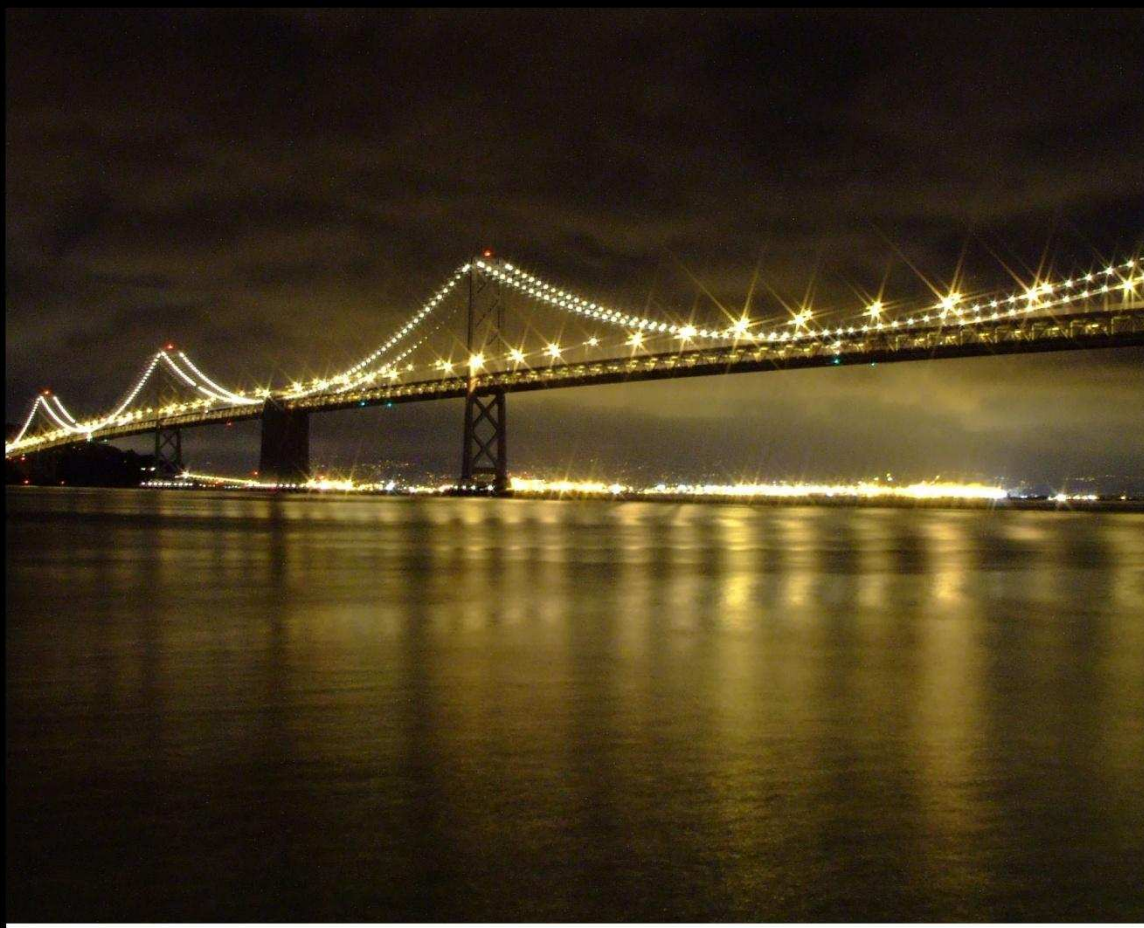


Adaptando los Procesos a la Empresa

Flexibilidad con Scrum

Principios de diseño e implantación de campos de Scrum

Juan Palacio

Flexibilidad con Scrum

Principios de diseño e implantación de campos de Scrum

Apuntes, conceptos y principios para diseñar, implantar y gestionar proyectos ágiles y equipos multidisciplinares.

Juan Palacio

Título

Flexibilidad con Scrum

Autor

Juan Palacio

Imagen de Portada

Carlo D.C.

Edición

Octubre – 2008[1]
Noviembre - 2007

Impresión

Versión impresa, disponible en http://www.lulu.com

Derechos



http://www.safecreative.org/work/0710210187520

Las condiciones en las que se puede usar y distribuir este trabajo están registradas y se pueden consultar en Safe Creative

---

[1] Fe de errata. Dice "Octubre 2008", debe decir "Octubre 2007"

A Ana

Del mismo autor

Si compartes una visión de los modelos de procesos, las prácticas y la agilidad para empresas del conocimiento, flexible y global: cubriendo a toda la organización, no sólo a la gestión del proyecto o las buenas prácticas de programación…

Si cuestionas tu propio conocimiento, desconfías de las soluciones de marca y las verdades absolutas.

Te invito también a estos dos proyectos de conocimiento abierto, a los que doy forma por la afición de aprender y el gusto por compartir y ayudar.

Mi blog sobre gestión de proyectos y organizaciones



http://www.navegapolis.net

La ilusión de dar forma a una plataforma de formación y un recurso educativo abierto para formación profesional continua y acreditación libre: basada sólo en el conocimiento y la profesionalidad.



http://www.scrummanager.net/ok

Contenido

Contenido 1

Prólogo 11

FORMATO Y ORGANIZACIÓN DEL LIBRO 14

GESTIÓN DE PROYECTOS: DIMENSIÓN PREDICTIVA Y DIMENSIÓN ÁGIL. 17

Gestión de proyectos predictiva 19

¿QUÉ ES UN PROYECTO? 19

ORIGEN DE LA GESTIÓN DE PROYECTOS 21

PRINCIPIOS DE LA GESTIÓN DE PROYECTOS PREDICTIVA (CLÁSICA) 24

Patrón de trabajo de la gestión predictiva: 24

ÁMBITO DE LA GESTIÓN DE PROYECTOS 25

ERRORES FRECUENTES DE ENFOQUE EN LA GESTIÓN PREDICTIVA 26

Los sub-productos no son "deberes" 28

Gestionar no es controlar 28

La cultura de cumplimiento es contagiosa 29

El nuevo escenario 31

VELOCIDAD 31

INCERTIDUMBRE 34

Campos de Scrum: nuevo modelo para el nuevo escenario 37

Diferencias entre el "campo de Scrum" y el modelo clásico de desarrollo. 40

CARACTERÍSTICAS DE LOS CAMPOS DE SCRUM. 41

1.- Incertidumbre 42

2.- Auto-organización. 42

3.- Fases de desarrollo solapadas. 43

4.- Control sutil 44

5.- Difusión del conocimiento 45

Nuevos principios: el manifiesto ágil 47

ESTAMOS PONIENDO AL DESCUBIERTO MEJORES MÉTODOS PARA DESARROLLAR SOFTWARE… 47

EL MANIFIESTO ÁGIL 48

Los individuos y su interacción por encima de los procesos y las herramientas. 48

El software que funciona por encima de la documentación exhaustiva 49

La colaboración con el cliente por encima de la negociación contractual 50

La respuesta al cambio por encima del seguimiento de un plan 50

LOS PRINCIPIOS DE MANIFIESTO. 51

Gestión ágil de proyectos 55

INTRODUCCIÓN 55

OBJETIVOS DE LA GESTIÓN ÁGIL 56

Valor 56

Reducción del tiempo de desarrollo 58

Agilidad y flexibilidad 59

Resultados fiables 60

PRINCIPIOS DE FUNCIONAMIENTO DE LA GESTIÓN ÁGIL 60

ESTRUCTURA ÁGIL DE SCRUM. 61

1.- Concepto 62

2.- Especulación 62

3.- Exploración 63

4.- Revisión 63

5.- Cierre 63

Gestión, ¿predictiva o ágil? 65

INTRODUCCIÓN: HAY DOS FORMAS DE VIAJAR 65

¿ÁGIL, CLÁSICA, PREDICTIVA…? 66

PREMISAS DE LA GESTIÓN DE PROYECTOS PREDICTIVA 67

CARACTERÍSTICAS DE LA GESTIÓN DE PROYECTOS PREDICTIVA 68

HAY OTRAS PREMISAS 68

1.- ¿El objetivo de cualquier proyecto siempre es: producto, costes y fechas planificadas? 69

2.- ¿Todos los proyectos comparten los mismos patrones de ejecución? 69

Por las circunstancias de negocio del cliente 71

Por las circunstancias del proyecto 72

Por las circunstancias de la organización suministradora 76

MANAGEMENT EN EL DESARROLLO DE SOFTWARE 79

Modelos y metodologías: El mapa del bosque 81

¿QUÉ HACER CON LA CRISIS DEL SOFTWARE? 81

PROPUESTA CLÁSICA. 83

PROPUESTA ÁGIL. 86

DSDM 86

Extreme Programming 87

Scrum 87

Otros modelos o prácticas ágiles. 88

Software, personas y procesos 91

INTRODUCCION 91

LA MADUREZ DE LOS PROCESOS 91

NO SÓLO SON PROCESOS 94

RELEVANCIA DEL CAPITAL ESTRUCTURAL Y RELEVANCIA DEL CAPITAL HUMANO 96

LAS CARACTERÍSTICAS DEL SOFTWARE 99

COSTE DE LA MATERIA PRIMA 99

MALEABILIDAD 100

VALOR APORTADO POR LAS PERSONAS 100

FACTOR DE ESCALA 101

Scrum Management: Síntesis, flexibilidad y gestión sistémica 103

ESTRATEGIA DE SCRUM MANAGEMENT: SÍNTESIS + FLEXIBILIDAD + GESTIÓN SISTÉMICA 106

SÍNTESIS 106

FLEXIBILIDAD 106

GESTIÓN SISTÉMICA 107

PROCESOS Y PERSONAS EN LA NUEVA ESTRATEGIA DE GESTIÓN. 108

SCRUM MANAGEMENT 111

Los criterios de la gestión flexible en el Software 113

INTRODUCCIÓN 113

¿QUÉ HACER PARA QUE LOS PROYECTOS DE SOFTWARE SALGAN BIEN? LA TESIS: 113

¿QUÉ HACER PARA QUE LOS PROYECTOS DE SOFTWARE SALGAN BIEN?: LA ANTÍTESIS 114

CRITERIOS PARA CUESTIONAR LOS "POR QUÉS" 115

LA ORGANIZACIÓN 117

EL TRABAJO 118

ADQUSICIÓN – SUMINISTRO. 119

DESARROLLO 120

MANTENIMIENTO 121

PROCESOS ORGANIZACIONALES 121

LA PRÁCTICA DE SCRUM 123

El modelo Scrum 125

EL ORIGEN 125

SCRUM PARA SOFTWARE 125

INTRODUCCIÓN AL MODELO 125

CONTROL DE LA EVOLUCIÓN DEL PROYECTO 126

Revisión de las Iteraciones 126

DESARROLLO INCREMENTAL 127

Desarrollo evolutivo 127

Auto-organización 127

Colaboración 128

Visión general del proceso 129

LAS REUNIONES 129

LOS ELEMENTOS 130

LOS ROLES O RESPONSABILIDADES 131

Responsabilidad del producto: El propietario del producto 131

Responsabilidad del desarrollo: El equipo 132

Responsabilidad del funcionamiento de Scrum (scrum manager) 132

HERRAMIENTAS 133

Gráfico Burn-Up 133

Gráfico Burn-Down 133

Juegos y protocolos de decisión 133

CONCEPTOS Y MÉTRICAS 134

Tiempo real o tiempo de trabajo. 134

Tiempo teórico o tiempo de tarea 134

Puntos de función o puntos de funcionalidad 134

Estimaciones 134

Velocidad absoluta 134

Velocidad relativa 134

VALORES 135

Scrum: Los elementos 137

INTRODUCCIÓN 137

LOS REQUISITOS EN EL DESARROLLO ÁGIL 137

Requisitos y visión del producto 139

PRODUCT BACKLOG: LOS REQUISITOS DEL CLIENTE 140

FORMATO DEL PRODUCT BACKLOG 142

SPRINT BACKLOG 143

CONDICIONES 143

FORMATO Y SOPORTE 143

EJEMPLOS 144

EL INCREMENTO 146

Las reuniones 147

INTRODUCCIÓN 147

PLANIFICACIÓN DEL SPRINT 147

DESCRIPCIÓN GENERAL 147

Pre-condiciones: 148

Entradas: 148

Resultados: 148

FORMATO DE LA REUNIÓN 149

Primera parte: 150

Segunda parte: 151

FUNCIONES DEL ROL DE SCRUM MANAGER 151

Pizarra de trabajo 152

Un ejemplo de pizarra. 153

MONITORIZACIÓN DEL SPRINT 157

DESCRIPCIÓN 157

PRE-CONDICIONES 157

ENTRADAS 157

RESULTADOS 157

FORMATO DE LA REUNIÓN 157

REVISIÓN DEL SPRINT 158

DESCRIPCIÓN 158

OBJETIVOS: 158

PRE-CONDICIONES 159

ENTRADAS 159

RESULTADOS 159

FORMATO DE LA REUNIÓN 159

Las herramientas 161

GRÁFICO BURN-UP 161

GRÁFICO BURN-DOWN 164

JUEGOS Y PROTOCOLOS DE DECISIÓN 165

Estimación de póquer 165

Ejemplo de uso en la reunión de planificación del sprint 167

Scrum Management: Responsabilidades 169

INTRODUCCIÓN 169

RESPONSABILIDADES DE MANAGEMENT 170

Equilibrio sistémico de la empresa 170

Coherencia del modelo 170

Medios y formación 170

RESPONSABILIDADES DE PROCESOS 171

Configuración de Scrum 171

Mejora continua 171

Garantía de funcionamiento de Scrum 171

RESPONSABILIDADES DE PRODUCCIÓN 171

Visión del producto 171

Auto-organización 171

Tecnología ágil 171

¿RESPONSABILIDADES O ROLES? 172

Trabajos citados 177

Índice 181

Prólogo

El modelo de procesos Scrum para desarrollo de software es muy simple; tanto que seguramente llamarlo "modelo de procesos" sea un exceso, porque en realidad es un conjunto de buenas prácticas de trabajo, que han demostrado en muchos proyectos la capacidad para ofrecer valor al producto final y agilidad en el desarrollo.

Por su simplicidad, la descripción detallada del formato Scrum, que no la de su fondo, no llenaría ni una docena de páginas, y por eso mismo a muchas empresas les cae simpático, porque no necesita de los despropósitos de tiempo y dinero que requiere el asentamiento de modelos tipo ISO 15504, ITIL, CMMI…

Y lo cierto es que implantar Scrum en la empresa no es difícil, pero habría que tener en cuenta que una cosa es mudar la forma y otra más compleja, hacer lo mismo con el fondo. Una cosa es adoptar prácticas ágiles y otra hacer una empresa ágil; pero, ¿porqué agilidad, y no procesos?; que esta es otra cuestión que da mucho que hablar.

Hay dos tendencias: por un lado empresas que se, o les prescriben como remedios de mejora a ISO 9003 o CMMI o ITIL… o incluso combinaciones de varios; y por otro las que optan por algún modelo o combinación de modelos ágiles: Extreme Programming, Scrum, FDD, etc.

También están las adictas que, seguramente sin pensarlo mucho, se meten todo lo que pillan: las veintitantas áreas de proceso CMMI, más las prácticas de Scrum, Extreme Programming…

Y es que estamos en un sector con una imagen de confianza bastante pobre. Los informes que tenemos por más rigurosos (Group, 1994 - 2004) afirman que nuestra industria no es capaz de terminar con éxito ni uno de cada tres proyectos; realidad que lanza a muchos técnicos a abrazar modelos y *best-practices*, y por la que muchos empresarios compran todas las certificaciones que pueden, aunque éstos generalmente más preocupados por mejorar la fotografía que la radiografía de su empresa

Y por aquí va la razón de ser de este libro, y la de empezar a hablar de Scrum sin mencionar siquiera, si es mejor la agilidad o los procesos; que no es por pasar de puntillas sobre el tema, presuponiendo que quien toma

un libro con el título "Flexibilidad con Scrum" es un convicto de la agilidad, a quien ya sobran las razones de adoctrinamiento; sino porque ésta es precisamente una de las razones del libro: el sinsentido de la dualidad: "o eres de procesos o eres de agilidad".

Con el mismo criterio de Nonaka y Takeuchi (Takeuchi & Nonaka, 2004), que describieron por primera vez el trabajo ágil en "campos de scrum"; el libro comparte y se asienta en la premisa de que el conocimiento está en continua evolución a través del patrón dialéctico de tesis, antítesis y síntesis (Ikujiro & Takeuchi, 1986).
De que al cuestionar la tesis desde su antítesis, se genera la tensión entre contrarios que mueve la evolución del conocimiento.

Desde esta perspectiva, y tomando una cierta distancia; el primer conocimiento de nuestra industria es el que cristalizó en los 80 y 90 prescribiendo procesos de ingeniería, producción y calidad centrada en los procesos.
Esta es la tesis inicial de nuestra evolución y ha generado el primer conocimiento útil para nuestra industria (ISO 9000-3, CMM'S, Bootstrap, PMI, SPICE, etc.)

Su aplicación a partir de los noventa engendra a su réplica dialéctica: la agilidad. El Manifiesto Ágil (Beck, y otros, 2001) cuestiona la validez de los modelos de procesos y el uso de la gestión predictiva para desarrollar software.

La tensión entre procesos y agilidad está produciendo como síntesis la "aplicación flexible desde el cuestionamiento": cuestionamiento, porque las prácticas y modelos no son de talla única válida para todas las empresas; y flexibilidad, porque siempre que sea posible es mejor adaptar el modelo a la empresa que lo contrario.

Además de esta propuesta como base, la razón de este libro se completa con el convencimiento personal de que:

Decir software es decir mucho y no decir nada. Que de software es el sistema de guiado de un misil balístico, y de software también el juego de la videoconsola; pero no es razón suficiente para que ambos se tengan que desarrollar con los mismos procesos de ingeniería, o con las mismas prácticas de gestión.

De que en nuestro sector hay empresas como SAP, Google o start-up's de 5 empleados; y que cada uno anda mejor con zapatos de su talla, diseñados para su terreno, que calzando todos talla y modelo único.

De que las organizaciones y pseudo-organizaciones de estandarización internacional como ISO o ESI empiezan a reconocer que sus modelos actuales no son los más apropiados para según qué empresas (Laporte & April, 2005) (April, Laporte, & Renault, 2006) (Garcia, Graettinger, & Kost, 2006) y están comenzando a dibujar la tendencia hacia la flexibilidad, aunque sus estructuras y la necesidad de coherencia con las versiones anteriores de modelos y certificaciones lastran la velocidad de evolución. Que sean ellas las que maquen el ritmo y la forma de trabajar, puede resultar suficiente en algunas empresas, pero es excesivamente lento y costoso para otras.

De que formatos pre-definidos para requisitos o backlog, para reuniones o para juegos de estimación pueden servirme, pero tampoco son tallas únicas para cualquier equipo de trabajo.

Por estas razones tomar el modelo que más encaja con nuestra empresa conociendo y respetando su fondo de conocimiento, pero no su forma, permite flexibilizarlo:

- Eliminar lo prescindible para nuestros proyectos.
- Adaptar las formas a nuestra organización, y no al contrario.
- Adoptar valores y prácticas útiles, con independencia de si son de la tesis o de la antítesis.

Supone tomar del modelo lo que nos ayuda, y olvidar lo que nos enreda. ¡Casi nada!; primero, porque se necesita nuestro criterio. No vale lo de hacer lo que nos dicen; y segundo porque supone que "uno tiene su propia fe" y como decía Georges Brassens (Brassens, 1952) "Non, les brav's gens n'aiment pas que L'on suive une autre route qu'eux[2]".

Si las razones de marketing de una certificación oficial en su empresa tienen más relevancia que las de la mejora real, estos consejos empiezan a ser proscritos. El marketing es una razón tan respetable como cualquier otra y cada empresa decide según su criterio y circunstancias, pero "los abanderados" no dan certificaciones por tener tu propia fe, sino por seguirles a ellos.

[2] "A la gente no gusta que, uno tenga su propia fe."

Y terminando también con palabras de Georges Brassens (Brassens, 1952): "… Je ne fait pourtant de tort à personne, en suivant les chemins que n'mènent pas à Rome[3]".

Los apuntes que recopilo en el libro son de conclusiones a las que llego por la experiencia y lo aprendido hasta ahora. Son los consejos que daría a un amigo, al que al mismo tiempo diría que siempre que sea posible, antes de copiar las formas de trabajar de otros, las cuestione, y si su realidad le demuestra que es mejor adaptarlas, que no lo dude. Por eso estas páginas no dan recetas para calcar, sino conocimiento que ojalá resulte útil para adaptar o diseñar las propias.

Formato y Organización del libro

Los conceptos expuestos están agrupados en tres áreas, que de forma conjunta dan perspectiva para aprehender mejor el fondo de los modelos de mejora y extraer conclusiones propias.

Las prácticas de Scrum se describen en la tercera, porque profundizar en ellas, sin caer en la cuenta de que hay dos modelos de gestión, ambos útiles, y con objetivos diferentes, puede dar una falsa sensación de seguridad al movernos con instrucciones concretas de supervivencia en un bosque del que desconocemos el mapa general.

Algo similar ocurre si se pasan por alto las características específicas de la industria del software, o de cada proyecto en particular.

Las tres partes que componen el libro son:

- ***Gestión de proyectos***: que define las diferencias entre la gestión tradicional y la gestión ágil; sus principios y objetivos, y criterios para determinar cuándo sus respectivas prácticas son necesarias, y cuando son molestas.

- ***Management***: en la que se expone que el desarrollo de software tiene particularidades propias, que marcan diferencias importantes de los entornos de producción industrial.
  Que a su vez, los sistemas de software son muy diferentes unos de otros, y que las empresas son sistemas complejos y

[3] "Que no quiero yo ningún lio si no va a Roma el camino mío"

relacionados.
Esta sección incluye principios que se suelen ignorar al implantar modelos de mejora en nuestras empresas. Principios para las áreas de gestión de la organización, necesarios porque la implantación de un modelo ágil implica mucho más que cambiar el formato de los requisitos o el protocolo de las reuniones.

- ***Scrum***: en la que se describen los principios de un marco de trabajo ágil, y se exponen como punto de referencia cómo son las prácticas de trabajo más habituales.

El libro se ha formado a partir de material de formación, seminarios y artículos, con el fin de compendiar los conceptos necesarios para obtener el mayor provecho en la implantación de un modelo Scrum; y con el fin adicional de transmitirlo de la forma más concisa y breve posible, por lo que emplea un formato más cercano al libro de texto o a los apuntes, que a una guía o descripción de trabajo.

GESTIÓN DE PROYECTOS: DIMENSIÓN PREDICTIVA Y DIMENSIÓN ÁGIL.

Gestión de proyectos predictiva

¿Qué es un proyecto?

Algunos productos se desarrollan "a medida", comenzando por el diseño, y ejecutando después un plan de ejecución; otros sin embargo son el resultado en serie de cadenas o procesos de producción.
Con los servicios ocurre algo similar: algunos son actuaciones únicas y específicas concebidas y realizadas para las necesidades de la ocasión, y otros son procedimientos normalizados, ejecutados según protocolos y prácticas estandarizadas, que con carácter repetitivo se emplean siempre para prestar el mismo servicio, o servicios del mismo tipo.

Se dice que los primeros son proyectos, y los segundos operaciones.
Unos y otros tienen tres características comunes:

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

Los productos o servicios realizados por las organizaciones pueden ser el resultado de operaciones o de proyectos. Las operaciones desarrollan productos de características similares, o prestan servicios con un mismo protocolo de actuación.

Los electrodomésticos, muebles, automóviles, refrescos, prendas de vestir, etc. son ejemplos de productos realizados a través de operaciones.

La evaluación SCAMPI para medir la madurez CMMI de una empresa de software, o la impartición de un curso reglado de java son ejemplos de servicios habitualmente realizados como operaciones

Las operaciones se ejecutan de forma repetitiva para obtener resultados de características similares.

Un proyecto produce un resultado único.

Los proyectos, además de las tres características anteriores:

- Producen un resultado único.
- Se desarrollan en un marco temporal pre-establecido.



Realizados por personas
Recursos limitados
Ejecución controlada

+

Obra única
Inicio y fin definidos

La gestión de proyectos predictiva define proyecto como:

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

Además de ser trabajos realizados por personas, de forma controlada, y con recursos limitados, los proyectos tienen por objetivo desarrollar resultados únicos, en un marco de tiempo pre-establecido.

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

Cada proyecto tiene objetivos y características propias y únicas.

Algunos necesitan el trabajo de una sola persona, y otros el de cientos de ellas; pueden durar unos días o varios años.

Algunos ejemplos de proyectos:

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

Origen de la gestión de proyectos

En los años 50, el desarrollo de grandes proyectos militares evidenció la necesidad de coordinación entre los equipos y disciplinas diferentes que trabajaban de forma simultánea en la construcción del mismo sistema.

Los proyectos grandes y complejos requerían el trabajo concurrente y sincronizado de múltiples ingenierías, e hicieron evidente, en los 60, la necesidad de elaborar modelos de organización y gestión para evitar los problemas que aparecían con recurrencia en todos los proyectos:

- Incumplimiento de agendas.
- Desbordamiento de costes.
- Funcionalidad deficiente.

Los proyectos han existido siempre.

Todo trabajo para producir un resultado único es un proyecto; pero la gestión de proyectos es una disciplina relativamente reciente que comenzó a forjarse en los años sesenta.

La necesidad de su profesionalización surgió en el ámbito militar.

Bernard Schriever, arquitecto de desarrollo de misiles balísticos Polaris, es considerado el padre de la gestión de proyectos, porque desarrolló el concepto de “concurrencia”, integrando todos los elementos del plan de desarrollo en un solo programa y presupuesto, ejecutándolos en paralelo y no secuencialmente. Consiguió de esta forma reducir considerablemente los tiempos de ejecución de los proyectos Thor, Atlas y Minuteman.

Siguiendo los pasos de la industria militar, la del automóvil también comenzó a aplicar técnicas de gestión de proyectos para la coordinación del trabajo entre áreas y equipos funcionales diferentes.

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

En los 50, Meter Norden, (Norden, Julio 1958)del laboratorio de investigación de IBM, identificó las necesidades de descomposición y planificación del trabajo en el desarrollo de sistemas complejos, para evitar los problemas que se repetían en este tipo de proyectos:

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

A partir de los años 60 surgieron las organizaciones que han desarrollado el cuerpo de conocimientos y las prácticas necesarias para gestionar los trabajos con las mejores **garantías de previsibilidad** de los resultados.

Ese cuerpo de conocimientos se ha ido desarrollando y configurando como currículo de una nueva profesión garante del éxito en los proyectos: La gestión de proyectos.

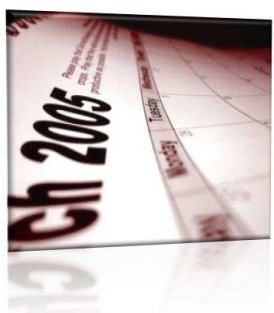
Las principales organizaciones que a partir de 1960 comenzaron a diseñar y asentar el cuerpo de conocimiento de la gestión de proyectos, y con él la definición de una nueva profesión son:

- Internacional Project Managenet Association (IPMA), fundada en 1965
- Project Management Institute (PMI) constituido en 1965
- Más tarde surgió Prince2, que comenzó a trabajar en 1989.

Éstas han trabajado en la identificación de patrones de comportamiento comunes a todos los proyectos, sobre los que construir una base de conocimiento y prácticas de gestión válidas para cualquier proyecto.

IPMA y PMI surgieron como organizaciones profesionales que con el tiempo han elaborado metodologías de gestión que llevan su nombre.

Prince2 ha seguido una evolución inversa. Comenzó siendo una metodología, y alrededor de ella el tiempo ha dado forma a una organización.



1965.- IPMA

1969.- PMI

1989.- PRINCE 2

Por el ámbito de aplicación de estas metodologías, también el sentido de evolución ha sido diferente para Prince2.

PMI e IPMA tuvieron desde el principio la finalidad de desarrollar un conocimiento de gestión válido para cualquier proyecto.

Prince2, sin embargo, comenzó como modelo de referencia para proyectos específicos de Tecnologías de la Información, desarrollado por la Central Computer and Telecommunications Agency (CCTA) del Gobierno Británico; y en la revisión realizada en 1996, se decidió ampliar el ámbito de validez, para cualquier tipo de proyecto.

La búsqueda de patrones comunes a todos los proyectos parte de los tres puntos señalados en los 50, por Meter Norden, (Norden, Julio 1958)

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

Se considera que un proyecto se ha desarrollado con éxito cuando se consigue la finalidad prevista, con el presupuesto y en las fechas que previamente se han estimado.



Los proyectos desarrollan obras, artefactos o servicios con un plan exclusivo. El plan de trabajo recorre un camino nunca antes realizado, y por tanto con niveles de riesgo altos, que suelen terminar torciendo, y a veces mucho, las estimaciones iniciales, e incluso generando problemas de calado suficiente para abortar la ejecución del proyecto.

Principios de la gestión de proyectos predictiva (clásica)

La gestión de proyectos nació para ofrecer previsibilidad en la construcción de grandes sistemas, con garantías de que el producto final se obtendrá en el tiempo y con el coste previamente estimados.

Patrón de trabajo de la gestión predictiva:

El producto final

1.- ¿Qué hay que construir?.

La gestión de proyectos parte de la descripción detallada de cómo debe ser el resultado (planos, requisitos…)

Con fecha y coste pre-estimados

2.-El plan del proyecto

Identificación de las tareas y recursos que se necesitarán para construir el producto.
Diseño del plan que consigue la coordinación y ejecución, con una combinación de recursos y tiempo adecuada a las necesidades y posibilidades del proyecto.

3.- Supervisión y coordinación de la ejecución para evitar desviaciones del plan.



Gestión basada en

PLANIFICACIÓN

SEGUIMIENTO

1. **¿Qué hacer?**
2. **Planificación del trabajo**
3. **Ejecución y control**

La gestión de proyectos desarrollada a finales del siglo pasado se basa en la planificación del trabajo, y en el posterior seguimiento y control de la ejecución.

La planificación se realiza sobre un análisis detallado del trabajo que se quiere realizar y su descomposición en tareas.

Parte por tanto de un proyecto de obra o requisitos iniciales detallados de lo qué se quiere hacer.

Sobre esa información se desarrolla un plan apropiado a los recursos y tiempos disponibles, y durante la construcción se sigue de cerca la ejecución para detectar posibles desviaciones, y en su caso, tomar medidas que las enmienden, o adaptar el plan inicial.

Es una gestión "predictiva", que pronostica, gracias al conocimiento detallado de lo que se va a hacer, y al plan del proyecto, las fechas, costes y recursos necesarios; así como la secuencia y coordinación de las operaciones.

Su principal objetivo es conseguir que el desarrollo resulte según lo "previsto"; y basa el éxito del proyecto en los tres puntos señalados: agendas, costes y calidad.

Ámbito de la gestión de proyectos

Desde la perspectiva ortodoxa, un gestor de proyectos es un "gestor formal," un planificador y controlador de las áreas que intervienen en el desarrollo del proyecto.

Para PMI (por ejemplo) las áreas de gestión que tiene a su cargo son (PMI, 2005):

- Gestión de la integración del proyecto.
- Gestión de costes.
- Gestión de la calidad.
- Gestión de tiempos y agendas.
- Gestión del alcance del proyecto.
- Gestión de la comunicación en el proyecto.
- Gestión de los riesgos.
- Gestión de proveedores.

Las industrias militar y automovilística fueron las primeras en adoptar las nuevas prácticas de gestión de proyectos, y por los buenos resultados que obtenían en la calidad y previsión de fechas y costes, su uso se ha ido extendiendo a prácticamente todos los sectores, y se ha ido formando un cuerpo de conocimiento común y único de gestión de proyectos.

Errores frecuentes de enfoque en la gestión predictiva

La gestión de proyectos predictiva centra la atención en la planificación, ejecución y control del trabajo.

La base de conocimiento desarrollada pone a disposición de los gestores técnicas y herramientas útiles para: ordenar ideas, registrar, consultar y analizar información.

- Diagramas de Gantt.
- Ruta crítica.
- Plan de comunicación.
- Plan de riesgos.
- Plan de calidad.
- Plan de recursos.
- Matriz de responsabilidades.
- Actas de reuniones.
- Etc.

Pero esto son herramientas, y no el trabajo que deben realizar.

El trabajo del gestor de proyectos no es: hacer el Gantt, el presupuesto, el plan de comunicación, el plan de riesgos, moderar las reuniones y redactar actas o registrar tiempos y gastos.

Su misión es garantizar el seguimiento del plan previsto, y según sea la organización de la empresa en la que trabaje tendrá mayor o menor libertad para usar unas u otras herramientas.

El uso de las herramientas es el medio, no el fin.

Las organizaciones que gestionan los proyectos con patrones predictivos, y disponen de modelos de procesos maduros, tienen definidas e institucionalizadas las prácticas de trabajo de los gestores de proyectos.

Procedimentar el trabajo es útil y necesario en entornos basados en procesos, pero se debe evitar que la rutina desvirtúe el objetivo de la gestión.

El objetivo no es tener dibujado un plan sobre un diagrama de Gantt. El objetivo es diseñar el plan con la distribución de recursos y con las rutas de tareas más adecuadas. El diagrama es solo un lenguaje para transmitir ese diseño.

Ídem con todas las tareas de gestión: selección de proveedores, recursos, comunicación, reuniones, riesgos, etc.

Los procesos ponen junto a ellas modos y protocolos de trabajo y registro, y se pueden hacer malas gestiones, malos planes y malos seguimientos con documentos y registros preciosos, confeccionados siempre en fecha y según las normas de la empresa.

El trabajo del gestor de proyectos no es observar y cumplir los pasos establecidos para la gestión del proyecto. El trabajo consiste en la gestión del proyecto

El hecho de que el cumplimiento de "lo prescrito" suela funcionar bien como escudo para eludir responsabilidades cuando las cosas van mal, refuerza el "complejo burócrata" en los mismos gestores.

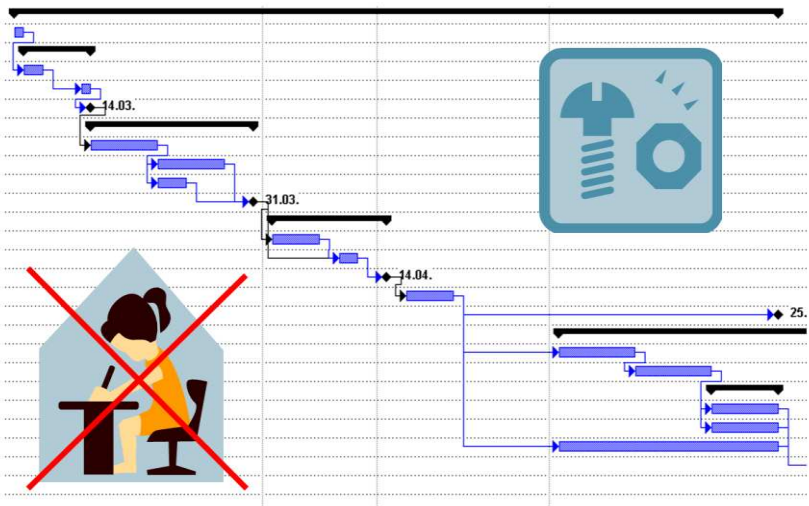
Por estas razones a veces los gestores de proyectos terminan trabajando como cumplidores de procesos.

Se deben evitar las ideas erróneas:

- Considerar que los sub-productos de la gestión son el trabajo del gestor.
- Asumir la obligación de control del equipo.
- Transmitir al equipo una cultura de cumplimiento.

Los sub-productos no son “deberes”

Considerar a los sub-productos del proyecto (requisitos, planificación, informes, registros…) como los “deberes del colegio” o los “deberes del gestor”, termina por darles el rango de “su obligación” o, la “obligación de su trabajo”; considerando por tanto que si hace la obligación, hace su trabajo.

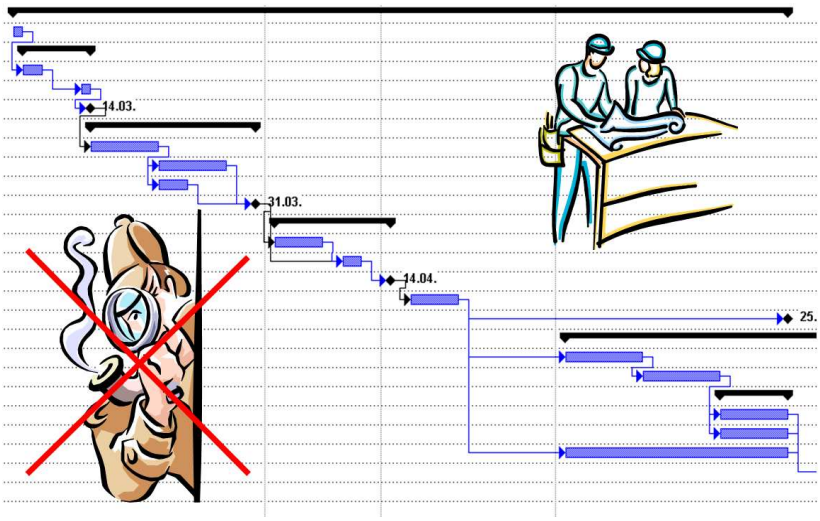


La mejor medida para evitar este vicio es que sean conscientes del riesgo, y lo conozcan tanto el departamento de calidad y procesos (si lo hay) como los responsables y directivos de las diferentes áreas, porque tienen una compromiso notable en los valores culturales de la organización, y son por ello los más indicados para evitar una “cultura de cumplimiento”, que hace olvidar que el fin de un diseño, o de un plan, una gestión… no es escribirlas y registrarlas según las normas, sino que sean el mejor, o el más innovador o el más adecuado (diseño, plan, gestión…) según los casos.

Gestionar no es controlar

En la gestión de proyectos predictiva el gestor diseña, traza el plan y es el responsable de su cumplimiento.

Además, a diferencia de la gestión ágil, no tiene por qué ser un conocedor de la tecnología empleada en el desarrollo.



La mezcla de estos factores tiende a dibujar puestos de "controladores". No son miembros integrados del equipo con aportación en las decisiones técnicas; y por el sentimiento de propiedad del plan y la responsabilidad de su ejecución es fácil que se limiten a adoptar posturas de control y orden jerárquico.

Sin entrar en preferencias sobre estilos de gestión, puede ser un patrón válido en trabajos y producciones de tipo industrial o mecánico cuyos resultados se deben más al valor de los procesos y la tecnología que al de las personas.

En proyectos de desarrollo de software, este modelo puede crear ambientes de trabajo inhibidores del talento personal.

La cultura de cumplimiento es contagiosa

La gestión que cae en el error de la cultura del cumplimiento, funciona como una fuente más de propagación en la organización, y la transmite al equipo.

Las formas pasan a ser también más importantes que los fines, y se da más relevancia a las horas de dedicación o al cumplimiento de las normas que a la eficiencia o calidad de los resultados.

El nuevo escenario

Dos variables modificaron el escenario del desarrollo de nuevos productos, a finales del siglo pasado.

- Velocidad
- Incertidumbre.

Velocidad

Hasta entonces los productos, una vez desarrollados, permanecían en los catálogos de ventas bastantes años, y en las cuentas anuales pesaban mucho más los ingresos por productos de catálogo, que los de las novedades.

A partir de los 80 la vida de los productos empieza a ser más breve, y una vez desarrollados apenas se mantienen unos meses el catálogo de novedades, y enseguida quedan fuera del mercado. Los ingresos de las empresas no dependen ya de los productos veteranos, sino de los últimos desarrollos.

En 6 años, Apple ha lanzado al mercado 5 modelos de iPod, y ha creado las líneas iPod mini e iPod Shuffle.



A principios de los años 30 se instalan los primeros aparatos de radio en el salpicadero de un automóvil. Desde entonces hasta principios de los 70, éstos no incorporaron modificaciones significativas.

Hasta los 70 las estrategias de posicionamiento, ventas, beneficio… se desarrollan sobre catálogos y escenarios de mercado muy estables.

En los 70 se incorporó el reproductor de casete.

En 1983 aparecen los primeros con CD-ROM

A los pocos meses incorporan gestión *juke-box* de los discos a través de los *cd-charger*.

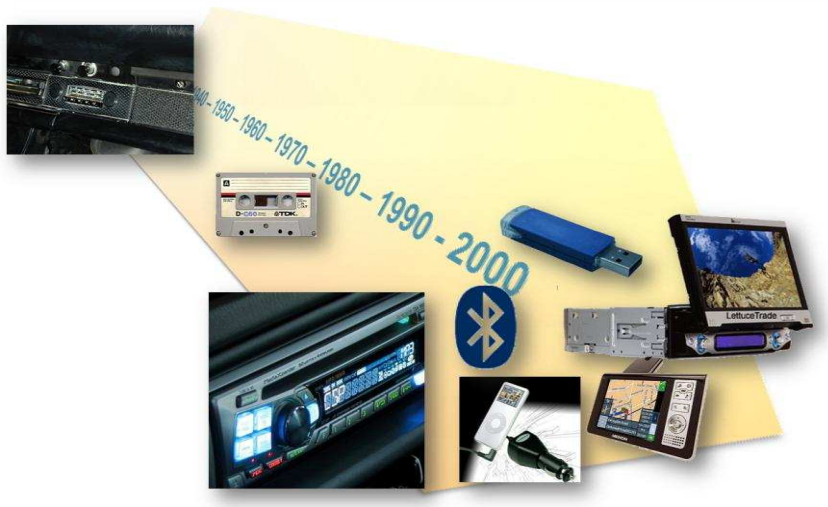
En 1988 aparece el sistema de información digital RDS.

A mediados de los 95 algunos incorporan sistemas de navegación.

En 2000 ya son habituales los reproductores de cd-rom con decodificación mp3.

Ahora mismo al diseñar un nuevo aparato, el concepto ya no es de radio o radio-casete sino de "*media-center*".
Se podría tener en cuenta:

- Bluetooth para comunicación con los teléfonos móviles de los pasajeros.
- Integración con navegación GPS.
- Compatibilidad no sólo con MP3: ¿ogg, wma, Real audio…?
- ¿Reproduccción de vídeo?
- Conexión con ipod y reproductores digitales
- Consolas de juegos
- Etc.



Hasta los 70 las estrategias de posicionamiento, ventas, beneficio… se desarrollan sobre catálogos y escenarios de mercado muy estables

Era un entorno estable, en el que el desarrollo de un nuevo producto se podía planificar sobre una descripción inicial completa, detallada y cerrada, en el que no era necesario tener en cuenta que antes de salir al mercado el producto pudiera estar ya obsoleto.

A partir de los 80, en muchas industrias el principal factor estratégico es: innovación continua.

A principios de los 90 la tecnología de grabación y lectura en discos ópticos estaba preparada para ofrecer una solución de vídeo doméstica de evidentes ventajas sobre los vídeos VHS: el videodisco láser.

- Soporte físico mucho más robusto y duradero.
- Calidad de imagen superior.
- El uso no degrada la calidad de la imagen.
- Pausa de imagen nítida y estable.

Además del ámbito doméstico, se comenzaron a diseñar aplicaciones de consulta y formación. Programas de PC que ofrecían textos formativos, y a través del puerto RS232C controlaban el videodisco para sincronizarlos con imágenes o animaciones relacionadas.

Se desarrolló el que iba a ser el “formato del futuro para vídeo”: El videodisco láser, capaz de contener “1 hora de vídeo en cada cara del disco” (65535 imágenes en formato analógico).

Discos de 12 pulgadas de diámetro (como los LP de vinilo).

Los departamentos de I+D de empresas como Sony o Pioneer, consideraron que la tecnología de discos ópticos digitales CD-ROM que ya se comenzaba a emplear en ordenadores, no era una amenaza porque:

Un CD-ROM sólo podía almacenar 640 Mb.
El formato digital (en lugar del analógico empleado por el videodisco) requería la necesidad de proceso para tratar la imagen en tiempo real.

Considerando que un segundo de vídeo debe incluir de 20 a 24 imágenes, y que una imagen a calidad aceptable tendría un tamaño aprox. De 256 Kb. Un CD-ROM, en el mejor de los casos, no almacenaría más de 20 minutos de vídeo.

Esos 20 minutos de vídeo, no podrían reproducirse si no se desarrollaban lectores de CD-ROM capaces de leer a mayor velocidad (entonces transferían 150 ó 300 Kb /sg).

Incertidumbre

En un escenario rápido e inestable, encajan mal las estrategias de negocio predictivas, que diseñan un producto y trazan un plan de negocio.

En entornos estables la previsión es una estrategia válida, pero no es realista para entornos rápidos.

En los entornos que evolucionan con rapidez las empresas que invierten trabajo y análisis para trazar estrategias y planes, deben cambiarlos constantemente para poder sobrevivir.

En los entornos rápidos los productos y estrategias que alcanzan el éxito no son los que se desarrollan sobre planes pre-concebidos, sino los que crecen en adaptación y replanteamiento constante guiados por la evolución del propio entorno.

Larry Page y Sergey Brin constituyeron una empresa llamada Google en 1998 para explotar comercialmente el motor de búsqueda que, durante el doctorado habían desarrollado en la Universidad de Stanford.

En 2.000 comienza a vender publicidad por palabras a 0,05$ por click, copiando la idea de goto.com.

¿Google sigue un plan de producto detallado o una visión general que se retroalimenta de forma continua?

Skype

En 2002 los autores del sistema P2P de ficheros Kazaa (Niklas Zennstrom y Janus Friis idean un proyecto de telefonía IP sobre el concepto P2P.

En septiembre de 2002 la firma de inversión de capital riesgo Draper Investment Company invierte en el proyecto.

En Octubre de 2005 Ebay compra a Skype.

Flickr

Ludicorp desarrolla en 2002 un web de herramientas para su proyecto de juego masivo Neverending.

La sala de chat llamada LicerLive incluía la posibilidad de publicación de fotos en tiempo real.

Los usuarios lo empleaban tanto, que el proyecto original de un juego masivo quedó desterrado y surgió un servicio para compartir fotos.

Marzo 2005. Yahoo compra flickr

YouTube

En febrero de 2005 tres empleados de Pypal publican un servicio con Flash para mostrar vídeos.

Sobre la idea Sequoia Capital invierte 3,5 millones de dólares, y en abril de 2006 otros ocho millones más.

En octubre de 2006 Google anuncia la compra de YouTube por 1.650 millones de dólares.

La incertidumbre es una consecuencia de la velocidad. En los sectores sometidos a una evolución muy rápida, no es posible, ni aconsejable, trazar un plan de negocio para un producto en el momento de su diseño.

Para triunfar en el nuevo escenario, lo importante no es tener garantías de que se va a cumplir un plan inicial.

Las variables claves para triunfar en el nuevo escenario son:

- Tomar retro-información del producto y del entorno de forma continua.
- Dar el mayor valor innovador al producto.
- En el menor tiempo posible.

- Para salir lo antes posible al mercado
- No hay producto "terminado" el producto está en continua evolución

Estos productos no necesitan modelos de gestión predictivos, sino adaptables.

En modelo predictivo el objetivo es lograr lo planificado: en un modelo de negocio predictivo el objetivo es producir el producto diseñado, con los costes y las ventas previstas en el plan de negocio.

En un modelo adaptativo el objetivo es dar al producto el mayor valor posible de forma constante.

Como el escenario es muy rápido es inestable no es realista pensar que se podrá mantener el plan de negocio trazado al empezar. Es más realista pensar que se podrá mantener el producto con el mayor valor posible.

El plan de negocio pasa a ser plan de valor. El objetivo es el valor, y el negocio la consecuencia lógica si se dispone de un producto que mantiene su valor de forma continua.

El modelo adaptativo no pide por tanto la predictibilidad de que el plan inicial se ejecute en las fechas y con los costes previstos, sino poder disponer en el menor tiempo posible de valor para el cliente, y mantener el producto o servicio en evolución continua para incrementarlo, o al menos mantenerlo.

Campos de Scrum: nuevo modelo para el nuevo escenario

A finales del siglo pasado, entre las industrias más afectadas por la velocidad y la inestabilidad de los entornos de negocio, algunas dejan de lado los modelos de desarrollo predictivo, y generan patrones propios con los que obtienen mejores resultados que sus competidores.

Cuando la teoría desarrollada sobre criterios de producción basada en procesos, y gestión predictiva va alcanzando una cierta madurez, o al menos abandonando la adolescencia, en los entornos de producción relacionados con la vanguardia tecnológica las empresas más competitivas empiezan a ignorar su teoría.

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

Es el referente que identifica el surgimiento de un nuevo modelo de desarrollo.

Los autores afirmaban en la introducción:

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

El artículo analizaba la forma de desarrollar nuevos productos empleada por las empresas que obtenían mejores resultados que su competencia a pesar del dinamismo del sector: Fuji-Xerox, Canon, Honda, Nec, Epson, Brother, 3M y Hewlett-Packard.

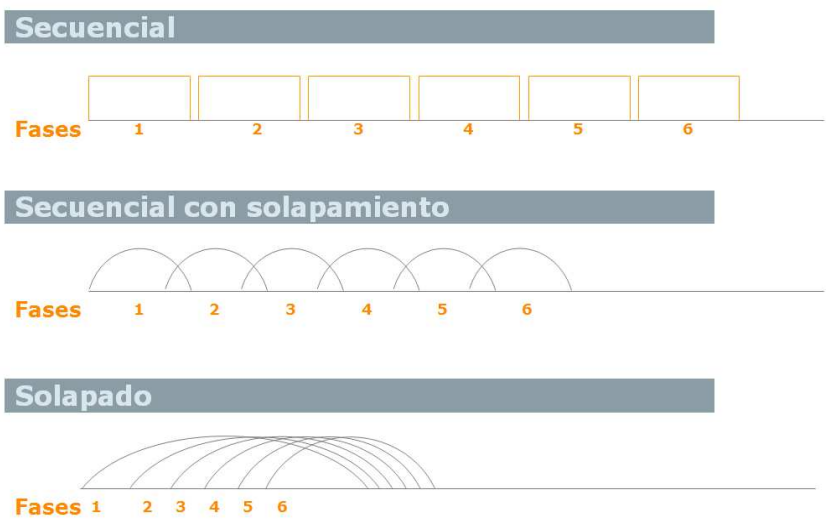
Analizó la forma en la que se concibieron y desarrollaron:

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

Estos productos se desarrollaron en tiempos inferiores a la media del sector, y aportaron valores innovadores muy importantes.

La principal diferencia con el modelo habitual fue el solapamiento de las fases de desarrollo.

El desarrollo tradicional de nuevos productos se basa en la especialización de funciones, de forma que cada departamento realiza la fase del desarrollo para la que es especialista, y como en una carrera de relevos, pasa el testigo al departamento siguiente hasta que finalmente se obtiene el producto terminado.



Las empresas observadas por Nonaka y Takeuchi obtenían mejores resultados, y tenían en común el uso de ciclos de desarrollo en el que las diferentes fases se realizaban de forma muy solapada.

La gente de marketing explora y estudia las necesidades de los clientes para generar el concepto del producto. Los ingenieros de investigación y desarrollo crean un diseño adecuado. Los de desarrollo llevan a cabo el proceso de producción, que pasará a los equipos de pruebas o integración…

La figura anterior muestra de forma gráfica el ciclo de vida del desarrollo de un producto en un patrón de gestión secuencial y la diferencia con la nuevo patrón que estaban constatando Nonaka y Takeuchi: las empresas con mejores resultados utilizaban ciclos en los que las fases se solapan de forma muy amplia.

El ciclo de desarrollo clásico, en el caso del software, es el denominado secuencial o en cascada, que comienza con la fase de requisitos, que una vez cerrados, da paso al diseño, posteriormente la codificación, las pruebas y la integración.

Los desarrollos secuenciales puros suelen ser más teóricos que prácticos y en realidad quienes los adoptan generalmente producen ciclos "secuenciales con solapamiento", en los que cada fase puede comenzar con el material aún no terminado por completo en la fase anterior. Así no es raro que partes de diseño de la arquitectura, por ejemplo, se comiencen cuando los requisitos aún no se han cerrado por completo; de igual forma, con partes de análisis aún pendientes de concretar, el equipo de codificación puede disponer ya de partes de diseño que permiten comenzar la codificación, etc.

Pero en el nuevo modelo de desarrollo no se trata de un cierto solapamiento entre fases, sino de un solapamiento tan amplio que durante la práctica totalidad del desarrollo concurren todas las actividades.

De esta forma, más que fases que se realizan de forma secuencial, pasan a ser actividades que se ejecutan en el momento que se requieren. Requisitos, análisis, codificación, pruebas, integración se van realizando en cada momento según las necesidades en la evolución del proyecto.

Diferencias entre el “campo de Scrum” y el modelo clásico de desarrollo.

En los proyectos analizados en el artículo todos los especialistas trabajaban de forma conjunta en un equipo único, y generalmente compartiendo el mismo espacio físico.

> ***A este entorno de trabajo Nonaka y Takeuchi le denominaron “campo de Scrum”, por la analogía entre el equipo de trabajo y un equipo de rugby.***

Los principales contrastes entre el desarrollo tradicional y el desarrollo ágil empleado en los “campos de Scrum” son:



DESARROLLO TRADICIONAL	DESARROLLO ÁGIL
Especialización	Equipo multidisciplinar
Fases	Solapamiento
Requisitos detallados	Visión del producto
Seguimiento del plan	Adaptación a los cambios





No lo desarrollan equipos diferentes con especialistas en distintas áreas. Hay un sólo equipo, formado por personas muy competentes, con perfiles y conocimientos que cubren las disciplinas necesarias para construir el producto.

No hay fases. Éstas pasan a ser tareas que se ejecutan cuando se necesitan. No se hace primero el diseño del concepto o los requisitos, más tarde el análisis, luego el desarrollo, etc.

Lo que en el software serían las fases de requisitos del sistema, requisitos del software, análisis, diseño, construcción, pruebas e integración, y se ejecutarían de forma secuencial, pasan a ser tareas que se llevan a cabo cada vez que hacen falta. Normalmente a lo largo de pequeñas iteraciones durante todo el desarrollo.

No se espera a desarrollar requisitos detallados antes de empezar el análisis o el desarrollo. Muchas veces éstos no se pueden conocer si no se avanza en el desarrollo, y se va viendo y "tocando" el resultado.

Otras veces el mercado es tan rápido, que a mitad de trabajo las tendencias o la competencia obligarán a modificar el producto.

La participación de todo el equipo en el diseño aporta mucho más talento innovador y diferencial; un valor clave en el mercado de productos y servicios TIC.

Se empieza a trabajar sin el detalle cerrado de lo que se va a producir. Se parte de la visión general. El descubrimiento paulatino durante el desarrollo, y las circunstancias que se irán produciendo en el entorno, dibujarán el detalle de forma paralela al desarrollo.

Características de los campos de Scrum.

Las características comunes que se identificaron en los entornos de desarrollo de las empresas analizadas fueron:

- La incertidumbre como elemento consustancial y asumido en el entorno y en la cultura de la organización.
- Equipos de desarrollo auto-organizados.
- Fases de desarrollo solapadas
- Control sutil
- Difusión y transferencia del conocimiento

1.- Incertidumbre

Como elemento consustancial y asumido en el entorno y en la cultura de la organización.

> ***El equipo de trabajo para diseñar el Honda City tenía una edad media de 27 años, y el nivel de detalle que para el nuevo producto le dieron los directivos de Honda fue: "El tipo de coche que a la gente joven de su segmento le gustaría conducir***

En estas empresas, desde la dirección se apunta cuál es la visión genérica que se quiere conseguir, o la dirección estratégica que hay que seguir, pero no un plan detallado del producto y su desarrollo.

Al mismo tiempo se da al equipo un margen de libertad amplio.

Los ingredientes clave que sirven de acicate para la creatividad y compromiso del equipo son:

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

2.- Auto-organización.

Son equipos auto-organizados. No hay roles de gestión que marquen pautas o asignación de tareas.

No se trata de equipos auto-dirigidos, sino auto-organizados. La gestión marca la dirección, pero no la organización.

Parten de cero. Deben empezar por crear su propia organización y buscar el conocimiento que necesitan.

Sin similares a una pequeña empresa "Start-up" en la que todos los integrantes trabajan de forma conjunta y auto-organizada, sin unos patrones organizativos impuestos por una estructura empresarial ajena al grupo.

La dirección de la empresa actúa como un inversor de capital riesgo que aporta los recursos necesarios para que trabajen en su proyecto

Para que los equipos puedan conseguir auto-organizarse deben reunir tres características:

- Autonomía: son libres para elegir la estrategia de solución.
- Auto-superación. El equipo va desarrollando soluciones que evalúa analiza y mejora.
- Auto-enriquecimiento: La multi-disciplinaridad de los componentes del equipo favorece el enriquecimiento mutuo y la adopción de soluciones valiosas y complementarias.

3.- Fases de desarrollo solapadas.

En el desarrollo ágil las "fases" pasan a ser "actividades". El concepto de fase implica sucesión secuencial de unas a otras. En un campo de Scrum los trabajos que se llevan a cabo pierden el carácter de fase y son actividades que se realizan en cualquier momento, de forma simultánea, o a demanda según las necesidades en cada iteración.

Por ejemplo, lo que para el desarrollo en cascada es una "modificación de requisitos", en un modelo ágil es información que enriquece o concreta la visión del producto. El mismo término "modificación" tiene implícito el concepto de que estamos "cambiando" algo que ya se había definido. La fase de requisitos ya se hizo; ya está completada.

En el desarrollo tradicional:

- Las transiciones entre fase y fase acaban funcionando como fronteras. Cada fase la realiza un equipo que siente como responsabilidad, más su trabajo, que el desarrollo conjunto. Los documentos de diseño, o requisitos o los prototipos pueden acabar siendo barricadas que en lugar de favorecer la comunicación directa favorecen la separación.
- El retraso en una fase hace de cuello de botella en el proyecto. El solapamiento diluye el ruido y los problemas entre fases.

4.- Control sutil

El equipo trabaja con autonomía en un entorno de ambigüedad, inestabilidad y tensión.

La gestión establece puntos de control suficientes para evitar que el ambiente de ambigüedad, inestabilidad y tensión del "campo de Scrum" derive hacia descontrol.

Pero la gestión no ejerce un control rígido que impediría la creatividad y la espontaneidad.

El término "control sutil" se refiere a generar el ecosistema adecuado para un "auto-control entre iguales," consecuencia de la responsabilidad y del gusto por el trabajo que se realiza.

Las acciones para generar el ecosistema de este control son:

- Seleccionando a las personas adecuadas para el proyecto, y analizando los cambios en la dinámica del grupo para incorporar o retirar a personas si resulta necesario.
- Creando un espacio de trabajo abierto.
- Animando a los ingenieros a "mezclarse" con el mundo real de las necesidades de los clientes.
- Estableciendo sistemas de evaluación y reconocimiento basados en el rendimiento del equipo.
- Gestionando las diferencias de ritmo a través del proceso de desarrollo.
- Siendo tolerante y previsor con los errores: son un medio de aprendizaje, y el miedo al error merma la creatividad y la espontaneidad.
- Implicando a los proveedores en el proyecto y animándoles también a su propia auto-organización

5.- Difusión del conocimiento

Tanto a nivel de proyecto como de organización.

Los equipos son multidisciplinares; todos los miembros aportan y aprenden tanto del resto del equipo como de las investigaciones, innovaciones de su producto y de la experiencia del desarrollo.

Las personas que participan en un proyecto, con el tiempo van cambiando de equipo en la organización, a otros proyectos; de esta forma se van compartiendo y comunicando las experiencias en la organización.

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

Nuevos principios: el manifiesto ágil

A mediados de los 90 algunos profesionales de la industria del software, dejan también de lado a los modelos predictivos y adoptan los principios de agilidad identificados en la década anterior por Nonaka y Takeuchi.

Estamos poniendo al descubierto mejores métodos para desarrollar software…

> *La palabra "mejores" de la introducción del Manifiesto Ágil es tendenciosa .La agilidad es una nuevo marco de desarrollo, no mejor o peor, sino más apropiado para entornos rápidos.*
> *Creer que un modelo de gestión es mejor que otro es la razón del enfrentamiento entre agilidad y disciplina.*
> *Una redacción más objetiva sería:*
> *"Estamos poniendo al descubierto **nuevos** métodos para desarrollar software…"*

En marzo de 2001, 17 críticos de los modelos de mejora para el desarrollo de software basados en procesos, convocados por Kent Beck, que había publicado un par de años antes el libro "Extreme Programming Explained" (Beck, extreme Programming explained Embrace Change, 1999)en el que exponía una nueva metodología denominada Extreme Programming, se reunieron en Salt Lake City para discutir sobre el desarrollo de software.

En la reunión se acuñó el término "Métodos Ágiles" para definir a los que estaban surgiendo como alternativa a los modelos formales, (CMM-SW, PMI, SPICE) que los consideraban excesivamente "pesados" y rígidos por su carácter normativo y fuerte dependencia de planificaciones detalladas, previas al desarrollo.

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

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

El Manifiesto Ágil

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

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

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

Firmado por:
Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland, Dave Thomas.

Los individuos y su interacción por encima de los procesos y las herramientas.

Este es posiblemente el principio más relevante del manifiesto.

Por supuesto que los procesos ayudan al trabajo. Son una guía de operación. Las herramientas mejoran la eficiencia, pero sin personas con conocimiento técnico y actitud adecuada, no producen resultados.

Los modelos de procesos (ISO 9000, CMMI) se fundamentan en el principio de calidad de Jurán, que en palabras del creador de CMMI (Watts Humphrey) afirma: “La calidad del resultado obtenido depende en

su mayor parte de la calidad de los procesos empleados" (Beth Chrissis, Konrad, & Shrum, 2003).

El Manifiesto Ágil afirma que en el desarrollo de software, la aportación de las personas es más relevante que la de los procesos o la tecnología empleados.

La defensa a ultranza de los procesos lleva a postular que con ellos se pueden conseguir resultados extraordinarios con personas mediocres, y lo cierto es que este principio es peligroso cuando los trabajos necesitan creatividad e innovación (v. personas y procesos en scrum management, pág. 108)

El software que funciona por encima de la documentación exhaustiva

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

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

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

La colaboración con el cliente por encima de la negociación contractual

Las prácticas ágiles están especialmente indicadas para productos difíciles de definir con detalle al principio, o que si se definieran de forma cerrada tendrían al final menos valor que al ir enriqueciendo la funcionalidad con la retro-información continua del desarrollo.

También son apropiadas las prácticas ágiles para los casos en los que se prevé inestabilidad en los requisitos por la velocidad del entorno de negocio.

Para el desarrollo ágil el objetivo no es dar garantías de previsibilidad sobre un plan previo, sino dar el mayor valor posible en cada iteración, y de forma continua.

Un contrato no aporta valor al producto. Es una formalidad que establece líneas divisorias entre responsabilidades, refleja cómo se han cerrado los requisitos, fechas y costes previstos, como referentes para posibles disputas contractuales entre cliente y proveedor.

En el desarrollo ágil el cliente es un miembro más del equipo, que se integra y colabora en el grupo de trabajo.

La respuesta al cambio por encima del seguimiento de un plan

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

Los principios de manifiesto.

Tras los cuatro valores descritos, los firmantes redactaron los siguientes principios que se derivan de ellos:

> ***Nuestra principal prioridad es satisfacer al cliente a través de la entrega temprana y continua de software de valor.***

La gestión predictiva satisface al cliente entregándole el producto definido en las fechas y costes previstos.

¿Qué es más necesario, previsibilidad o valor? ¿El producto se puede detallar con todo el valor en el momento inicial? ¿Qué prefiere el cliente: que le planifique y cumpla una fecha de entrega para darle un producto que no es capaz de definir hoy, o que le entregue lo máximo que se pueda hacer en la primera fecha que él necesita, y con el mayor valor que el equipo pueda conseguir?

> ***Son bienvenidos los requisitos cambiantes, incluso si llegan tarde al desarrollo. Los procesos ágiles se doblegan al cambio como ventaja competitiva para el cliente.***

La gestión ágil necesita enriquecer y dar valor a la visión del producto. La nueva información durante el desarrollo no son "modificaciones de requisitos", sino fuente de análisis y valor para el producto. Los requisitos que surgen al probar partes ya desarrolladas o lo que la competencia lanzó al mercado ayer, no son modificaciones que amenazan el plan, sino requisitos con información que aumenta el valor del producto.

> ***Entregar con frecuencia software que funcione, en periodos de un par de semanas hasta un par de meses, con preferencia en los periodos breves.***

Es el objetivo de la gestión ágil: entrega temprana y constante de valor: de partes funcionales cerradas que resultan valiosas, bien porque ya puede salir al mercado, o bien porque generan información para los requisitos siguientes con mayor valor para el producto.

Las personas del negocio y los desarrolladores deben trabajar juntos de forma cotidiana a través del proyecto.

Característica de los campos de Scrum: el equipo está compuesto tanto por las personas de desarrollo como por las de negocio. No trabajan en fases separadas sino de forma solapada e intercambian el conocimiento y la comunicación de forma directa.

Se produce un enriquecimiento mutuo al compartir el conocimiento, y un campo más apropiado para enriquecer las ideas de partida.

Construcción de proyectos en torno a individuos motivados, dándoles la oportunidad y el respaldo que necesitan y procurándoles confianza para que realicen la tarea.

Contar con personas valiosas y motivadas es un factor clave. La agilidad necesita talento y motivación.

El talento es algo que las personas sólo pueden proporcionar a través de la motivación.

Las organizaciones son sistemas relacionados. La cultura y política de gestión de personal debe estar alineado en este sentido.

La forma más eficiente y efectiva de comunicar información de ida y vuelta dentro de un equipo de desarrollo es mediante la conversación cara a cara.

La comunicación directa transmite con mayor precisión y riqueza que la escrita. Siempre que resulta posible la gestión ágil prefiere la comunicación directa a la realizada a través de la documentación del proyecto.

La auto-organización del equipo debe conducir la comunicación de forma eficiente.

El software que funciona es la principal medida del progreso

La gestión ágil no mide el progreso sobre la cantidad de trabajo realizada, sino sobre la cantidad de producto realizado, considerando producto a partes ya terminadas y funcionales.

En gestión ágil, que en una tarea con duración prevista de 40 horas, se hayan trabajado 30 no mide nada. A las métricas ágiles lo que les interesa es si está terminada, o cuánto le queda; pero no cuánto se ha trabajado.

Es posible que se hayan trabajado 40 horas, y aún queden 10 más. Eso no indicaría que se ha completado el 100% de la tarea. La tarea se ha completado cuando se entrega el resultado.

Los procesos ágiles promueven el desarrollo sostenido. Los patrocinadores, desarrolladores y usuarios deben mantener un ritmo constante de forma indefinida.

La gestión ágil no produce a través de esfuerzos heroicos, y marca pautas de organización para evitar la dispersión y el trabajo sin ritmo.

La atención continua a la excelencia técnica enaltece la agilidad.

La excelencia técnica es un objetivo interno de la agilidad, tanto para la organización, como para el proyecto y para las personas.

La adaptación continua al cambio requiere excelencia técnica en el diseño de la arquitectura, refactorización, simplicidad… Sin excelencia técnica por parte del equipo el resultado no tiene la sencillez, robustez y flexibilidad necesarias para desarrollarse en un entorno ágil, que exige cambio y modificación continua.

La simplicidad como arte de maximizar la cantidad de trabajo que se hace, es esencial.

Las construcciones elaboradas, densas y complejas hacen incómoda la evolución.

El desarrollo ágil se basa en la construcción iterativa. En la entrega continua de pequeños módulos de valor.

Los desarrollos se basan en la modularidad sobre "piezas" funcionales simples.

Las mejores arquitecturas, requisitos y diseños emergen de equipos que se auto-organizan

La "fertilización cruzada" entre el conocimiento de todos los miembros del equipo logra mejores resultados de diseño y arquitectura que la que pueden obtener las personas individualmente.

En intervalos regulares, el equipo reflexiona sobre la forma de ser más efectivo y ajusta su conducta en consecuencia

Las prácticas de trabajo y auto-organización incluyen análisis de mejora continua del propio modelo ágil empleado.

El objetivo de mejora continua del equipo o la organización no es realizar el modelo de forma ortodoxa según la implementación ágil definida por DSDM o Extreme Programming o Agile Alliance; sino mejorar de forma continua el propio modelo ágil para obtener mejores resultados

Este punto suele no tener implicación en las implementaciones de modelos ágiles, y es muy importante para dotar también a los modelos ágiles de patrones institucionalizados de mejora continua (v. institucionalización y mejora continua, pág. 115)

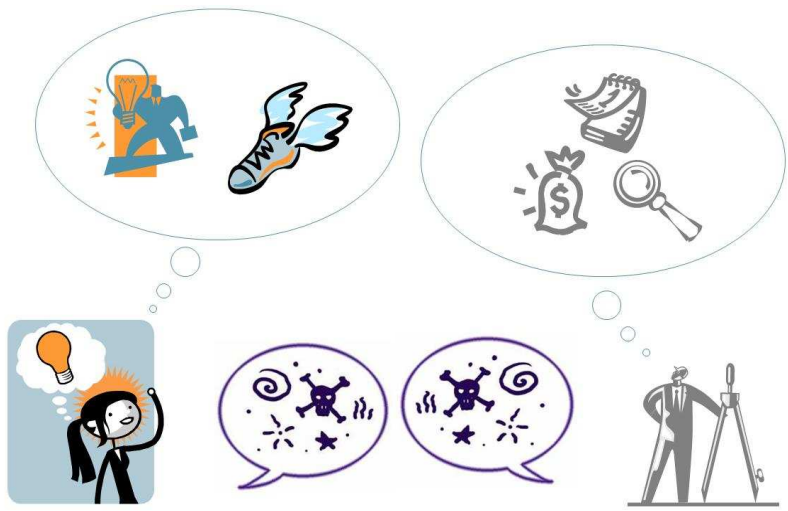
Gestión ágil de proyectos

Introducción

El nuevo escenario de negocio de muchos sectores necesita modelos diferentes para desarrollar sus productos.

Las circunstancias de los mercados y de las empresas no se pueden cambiar, y es la gestión de proyectos la que debe adaptarse y responder a las nuevas necesidades.

Las empresas acuden a los expertos en procesos de desarrollo con descripciones abiertas, solicitando adaptación continua y valor, y éstos les piden descripciones cerradas y les ofrecen garantías de cumplimiento de un plan.



Ahora es necesario desarrollar y construir el producto a la par de la investigación y del descubrimiento de los requisitos, y hacerlo con la capacidad de adaptarse a los cambios dictados por el entorno.

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

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

Quizá ya no hay "productos finales", sino productos en evolución, revisión mejora o incremento continuo a partir de la versión beta



Objetivos de la gestión ágil

La gestión ágil ha nacido de las prácticas empleadas en las empresas que mejor respuesta han sabido dar a las nuevas demandas:

- Valor
- Reducción de tiempo de desarrollo
- Agilidad y flexibilidad
- Fiabilidad.

Valor

Aunque ya está muy acuñado como principio que la gestión ágil es más adecuada para dar valor, ésta ha sido una redacción tendenciosa desde los principios ágiles.

Resulta más realista considerar que la gestión ágil es más adecuada para dar valor innovador.

El principal valor del producto para las empresas que promueven el desarrollo ágil es

LA INNOVACIÓN

El valor para el cliente no es un concepto de validez absoluta para todos los proyectos y sectores.

El desarrollo de una obra de ingeniería civil, el de un nuevo producto dietético, o un sub-sistema de software para un servicio "web 2.0" tienen valores diferentes.

La permanencia de las empresas de entornos ágiles, en el mercado depende de la capacidad de innovación continua. Del lanzamiento continuo de novedades, que tienen que competir con los productos competidores, que también están en innovación continua.

En la gestión tradicional cada equipo realiza la fase para la que está especializado con la información que necesita, y entrega el resultado al equipo siguiente, a modo de carrera de relevos.

En la gestión ágil todo el trabajo lo realiza un equipo multidisciplinar de forma conjunta, compartiendo toda la información del proyecto.

En este modelo de ingeniería concurrente, la información no está departamentalizada, y el conocimiento técnico de cualquier especialista puede "fertilizar" cualquier actividad del desarrollo, y no sólo la suya específica.

La ingeniería concurrente genera mayor valor innovador

La gestión tradicional aborda el desarrollo por fases, secuenciales, y cada una la realizan personas o departamentos diferentes. Los ingenieros de requisitos describen las funcionalidades que debe tener el sistema.

Con la especificación de requisitos, los analistas diseñan el sistema.

Los programadores traducen el diseño a código.

Luego se integra y se ejecutan las pruebas según las indicaciones del análisis.

El cliente ve al final el producto desarrollado.

Los analistas reciben un documento funcional ya elaborado y trabajan con la misión de elaborar el diseño. No pueden sugerir las posibles

mejoras que supondría considerar las funcionalidades de otra forma, que ellos pueden intuir al combinar su conocimiento de arquitectura de sistemas con el del problema del cliente; que además lo conocerían contado directamente por él.

De igual forma ocurre con las aportaciones de valor que podrían dar los programadores al combinar su conocimiento técnico con el conocimiento de las razones del cliente y del diseño.

Y cuando el cliente ve el producto, este ya está cerrado. Las sugerencias que le surgirán al probarlo quedan ya fuera del proyecto.

> ***El solapamiento de las fases produce el modo de producción que se ha venido a denominar "ingeniería concurrente", en la que todas las personas implicadas en el proyecto, cliente incluido trabajan de forma simultánea, en comunicación directa y combinando de forma simultánea toda la información del proyecto con el conocimiento y experiencia profesional de todo el equipo de desarrollo***

La ingeniería concurrente produce "fertilización cruzada de conocimiento" que produce resultados con mayor valor innovador.

Reducción del tiempo de desarrollo

En la década de 1990 la media de la salida de un nuevo producto al mercado en U.S. se redujo de 35,5 meses a 11 meses (Wujec & Muscat, 2002)

Esta reducción es un factor competitivo de primer orden para muchos productos. Las estrategias de la gestión ágil para producir resultados en menos tiempo que la gestión tradicional son:

- Entrega temprana de los primeros incrementos funcionales de producto, que corresponden con las partes que con mayor urgencia necesita el cliente, de forma que puede lanzar la primera versión de producto en el menor tiempo posible.
- Solapamiento de las fases de desarrollo.

Agilidad y flexibilidad

Agilidad: capacidad de responder rápidamente a las modificaciones de las directrices de trabajo.

Flexibilidad: capacidad de evolución del producto incorporando cambios y mejoras de forma continua.

La agilidad es la principal fortaleza y diferencia con el modelo predictivo. Éste para cumplir con su objetivo, cierra los requisitos al inicio y centra su esfuerzo en cumplir el plan. Le molesta el cambio porque supone una incidencia que lo altera.

En la gestión ágil el objetivo es dar valor innovador. Los requisitos están abiertos, y espera y busca información en el propio avance del proyecto y en el entorno, de forma continua.

Flexibilidad

La innovación debe ser continua, porque en las circunstancias del mercado, el producto no sólo es valioso por el componente innovador que tenga en el momento de su lanzamiento, sino también por su capacidad de adaptación y evolución, a través de versiones, modificaciones, actualizaciones, ampliaciones, etc.

No tiene por qué tratarse de innovación radical o invención (que también) sino generalmente es más el aporte de pequeños, pero continuados elementos diferenciadores para despertar el interés y la satisfacción de los consumidores, que de esta forma pasan a ser clientes.

El valor que necesitan estas empresas se basa en innovaciones de producto sistemáticas y continuas, lo que en Japón se denomina kaizen: mejora competitiva basada en la mejora continuada del producto.

En función de la maleabilidad del material con el que se trabaja,(v. pág. 100) la necesidad de flexibilidad puede traducirse en productos de rápida obsolescencia y producción constante de nuevos dispositivos (p.ej.: electrónica de consumo), o en la modificación continua del mismo producto a través de versiones (p. ej: software).

Para el primer caso la reducción de tiempo de desarrollo es el factor clave que permite la flexibilidad, y en el segundo juegan un papel relevante el uso de técnicas de desarrollo basadas en refactorización.

Resultados fiables

Los procesos de producción empleados por la gestión de proyectos tradicional tienen como finalidad la predictibilidad de los resultados: conseguir el trabajo planificado (y conocido de antemano) en el plazo planificado y por el coste previsto.

La gestión ágil no tiene un carácter predictivo o de anticipación. No conoce de antemano el detalle del producto o servicio que va a desarrollar; por eso su objetivo no es la fiabilidad en el cumplimiento de los planes, sino en el valor del resultado.

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

La fiabilidad es un valor relativo al modelo de gestión de proyectos empleado.

Un modelo de gestión predictiva es fiable si obtiene el producto definido, en las fechas y con los costes estimados.

Un modelo de gestión ágil es fiable si entrega de forma temprana, y repetida valor innovador.

Principios de funcionamiento de la gestión ágil

La forma de gestión puede y debe adaptarse a las circunstancias del proyecto: Puede documentar la visión del producto como "historias de usuario" con post-it's en una pared, como lista de desarrollo en un fichero Excel, etc.

Puede construir incrementos de producto en iteraciones de dos o de seis semanas, etc.,

Puede admitir cambios a mitad de un incremento (como Extreme Programming) o no (como Scrum).

Etcétera.

La forma debe ser un buen acomodo, apropiado a la organización y al proyecto, para que puedan trabajar los principios que hacen funcionar el desarrollo ágil:

1.- Operación preparada para responder al cambio, no para cumplir un plan.

2.- Reducción al mínimo indispensable de las especificaciones documentadas como textos. Preferencia por prototipos ligeros (dibujos o representaciones de interfaz, simulaciones) o pesados (prototipos operativos o partes del sistema ya terminadas).

3.- Implicación activa del cliente en el equipo de desarrollo.

4.- Valor del conocimiento tácito de las personas y de su interacción por encima del conocimiento de los procesos.

Estructura ágil de Scrum.

El patrón de ciclo de vida de un modelo de desarrollo ágil está compuesto de cinco etapas:

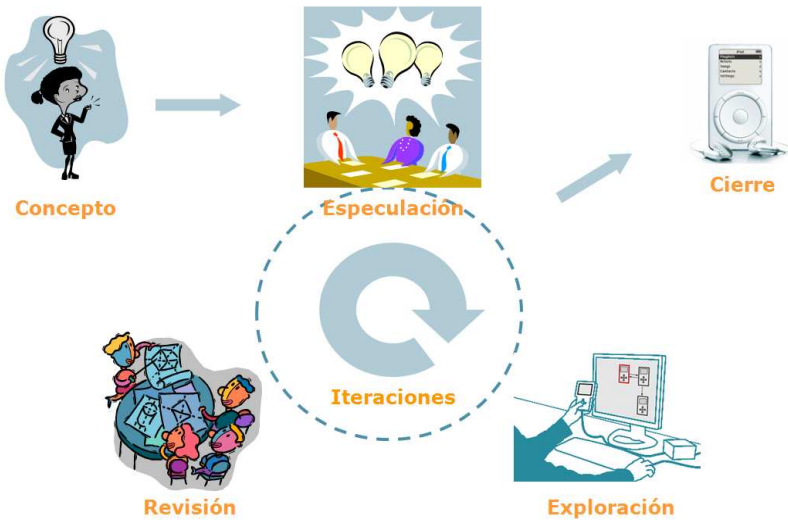
1.- Concepto

2.- Especulación

3.- Exploración

4.- Revisión

5.- Cierre



1.- Concepto

En la fase de concepto se crea la visión del producto o servicio que quiere obtener. Se decide y selecciona al equipo de personas que lo llevará a cabo.

Partir sin una visión determinada produce esfuerzo baldío. Igual que para una empresa, la visión es un factor crítico para el éxito del proyecto.

Se necesita tener la visión de lo que se quiere, y conocer el alcance del proyecto.

Esta información la deben compartir todos los integrantes del equipo.

2.- Especulación

Una vez que se dispone de la visión de lo que se quiere conseguir, el equipo especula y construye hipótesis sobre la información de la visión, que *per se* es muy general e insuficiente para determinar las implicaciones de un desarrollo (requisitos, diseño, costes…).

En esta etapa se determinan las limitaciones impuestas por el entorno de negocio (costes y agendas principalmente) y se determina la primera aproximación de lo que se puede producir.

La gestión ágil investiga y desarrolla tomando como partida la visión del producto. Durante el desarrollo confronta la realidad de lo que va

obteniendo. Su valor, posibilidades y la situación de negocio del entorno en cada momento.

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

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

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

3.- Exploración

Desarrollo de las funcionalidades que para generar el siguiente incremento de producto, ha determinado el equipo en la etapa anterior.

4.- Revisión

El equipo y los usuarios revisan las funcionalidades construidas hasta ese momento.

Trabajan y operan con el producto real para determinar su alineación y dirección con el objetivo

5.- Cierre

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

Posiblemente éste seguirá en el mercado, y si se emplea gestión ágil es presumible que se trate de un producto que necesita versiones y mejoras frecuentes para no quedar obsoleto. No quiere decir necesariamente que se ha terminado el proyecto.

Lo que para en un ciclo de desarrollo secuencial sería "mantenimiento," en un entorno ágil es la continuidad del proyecto en ciclos incrementales hacia la siguiente versión para ir acercándose a la visión del producto, que también es posible que vaya evolucionando con en el tiempo, al ritmo de su entorno tecnológico y de negocio.

Gestión, ¿predictiva o ágil?

Introducción: Hay dos formas de viajar

Forma 1:
Salimos el sábado. Con el AVE de las 12 vamos a Madrid, y desde allá en avión hasta el aeropuerto de Túnez. Un autocar nos lleva al hotel en Hammamet. El domingo día de playa. El lunes empezamos una ruta turística por el desierto de 3 días...

Forma 2:
Quiero pasar las vacaciones por Italia. La idea es salir en la primera semana de junio y estar entre 15 y 20 días allá. Tengo un presupuesto de 3.500 Euros. Quiero comenzar en Roma, pero luego no sé si ir hacia el norte e intentar abarcar Florencia, Bolonia y Venecia; o hacia el sur para conocer Nápoles y dar una vuelta por Sicilia. Bueno, lo decidiré sobre la marcha.

El primer viajero elabora un plan detallado. Cuando todo lo tiene encajado y conoce el coste, la duración, etc... empieza el viaje. Si durante el mismo algo se tuerce, pierde un vuelo... tomará medidas para intentar mantener el plan inicial, y si no es posible trazará uno nuevo.

El segundo viajero comienza el viaje sin un desarrollo detallado. Tiene una visión general del objetivo. El devenir de los acontecimientos y la información de cada momento irán escribiendo los detalles del viaje.



PLANIFICACIÓN



ADAPTACIÓN

¿Una es la forma buena y otra la mala?

Hay dos formas de gestionar proyectos: una predictiva o clásica y otra adaptativa o ágil.

La gestión predictiva parte de un plan detallado. Sabe exactamente qué es lo que va a hacer y conoce fechas y costes. Durante el desarrollo gestiona riesgos, evalúa el impacto que cada modificación supone sobre el plan inicial, toma decisiones frente a los imprevistos para seguir su cumplimiento; y si no queda más remedio, para replantearlo.

La gestión adaptable parte desde una visión general del objetivo y va dando pequeños pasos hacia él a través de un ciclo de construcción incremental, y de forma evolutiva contrasta y va descubriendo el detalle que resulta más adecuado para el producto con los usuarios y demás participantes, que pueden "tocar" o usar las partes construidas.

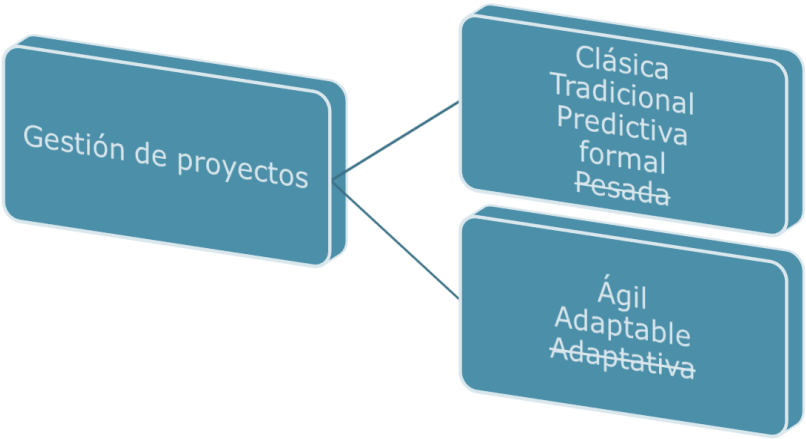
Hay un modelo que quiere conseguir previsibilidad: construir lo previsto, en el tiempo previsto y con el coste previsto.
Hay un modelo que quiere conseguir valor competitivo en entornos rápidos de la economía actual: innovación y agilidad.
Hay gestores que sólo trabajan de forma predictiva por la creencia fundamentalista de que es "la buena", y descalifican a los enfoques ágiles o adaptativos.
Hay gestores que sólo trabajan de forma adaptativa por la creencia fundamentalista de que es "la buena", y descalifican a los enfoques clásicos o predictivos.

El viaje de negocios de un ejecutivo necesita ajuste y previsión de agendas. Se debe gestionar con un patrón predictivo.
Un viaje de turismo innovador para descubrir rutas poco frecuentes, capaz de cambiar rápidamente si empeora el tiempo en la zona, o sacar ventaja de la pérdida de un avión se debe gestionar con un patrón adaptable.

¿Ágil, clásica, predictiva…?

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

Aquella se autodenominó ágil, y se hizo necesario dar otro "apellido" para la gestión de proyectos que hasta entonces, por única, no lo había necesitado.



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

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

Premisas de la gestión de proyectos predictiva

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

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

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

Características de la gestión de proyectos predictiva

Como consecuencia de las dos premisas anteriores, sus características son:

1.- Universalidad.

Los proyectos, pese a su diversidad, comparten patrones comunes de ejecución y regularidad.

Las prácticas de gestión trabajan sobre esos patrones comunes y resultan válidas para cualquier tipo de proyecto.

2.- Carácter predictivo.

La gestión predictiva define con detalle cuál es el producto previsto y elabora un plan de desarrollo sobre el que calcula costes y fechas.

Durante la ejecución realiza actividades de seguimiento y vigilancia para evitar desviaciones sobre lo planificado.



Hay otras premisas

La necesidad de innovación y agilidad ha cuestionado la validez de las dos premisas que cimientan el desarrollo de la gestión de proyectos:

- El objetivo es producir el producto definido, en costes y fechas
- Todos los proyectos comparten los mismos patrones de ejecución.

1.- ¿El objetivo de cualquier proyecto siempre es: producto, costes y fechas planificadas?

¿Y si las necesidades de innovación y agilidad pesan más que las de previsibilidad?

¿Y si el cliente no sabe o no quiere dar un dibujo cerrado del producto; no quiere que le cierren los requisitos, y prefiere incluir modificaciones y cambios de forma continua?

¿Y si no se trata de acotar el marco temporal de desarrollo, sino de tener el producto en continuo desarrollo, dándole valor de forma constante?

2.- ¿Todos los proyectos comparten los mismos patrones de ejecución?

¿Es adecuado emplear los mismos principios en dos proyectos tan diferentes, como el desarrollo de un nuevo servicio web 2.0 y la construcción de un puente?

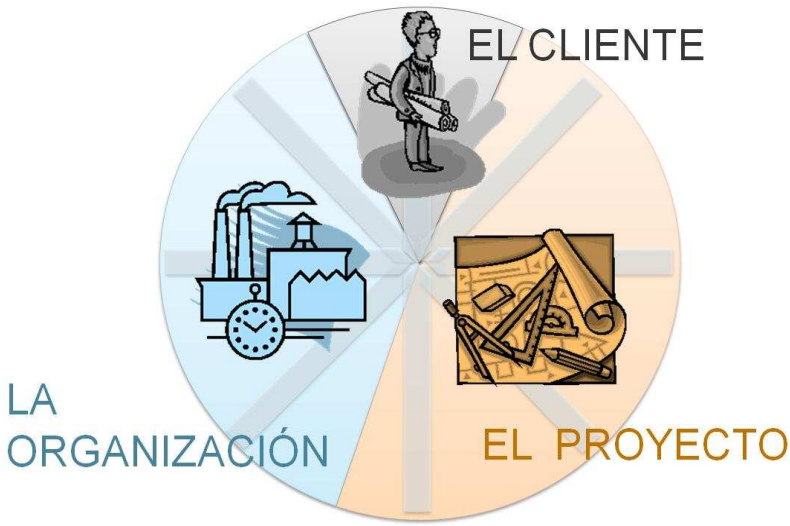
Es cierto que muchas características que diferencian unos proyectos de otros son superficiales y resultan indiferentes para el modelo de gestión; pero hay otras que permiten adoptar prácticas de gestión muy distintas.

Hay características relevantes que piden estrategias de gestión de proyectos diferentes.

Pueden ser:

- Por las necesidades del cliente
- Por las características del proyecto
- Por las características de la organización que desarrolla el sistema

Características determinantes



Cliente	Prioridad de negocio
Proyecto	Estabilidad de los requisitos
	Maleabilidad y coste de la materia prima
	Criticidad del sistema que se debe construir
	Coste de prototipado
	Tamaño del equipo
Organización	Cultura de la organización
	Nivel técnico del equipo
	Estrategia de desarrollo: procesos / personas

Por las circunstancias de negocio del cliente



Prioridad de negocio

¿Cuál es la prioridad más relevante para los intereses de negocio del cliente?

¿Qué tiene más importancia: el cumplimiento de agendas y fechas o el valor innovador del producto?

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

La gestión adaptable es un modelo especializado en dotar al producto del mayor valor innovador posible.

Por supuesto los dos objetivos son deseables, pero hay que determinar un nivel de equilibrio porque son excluyentes.

No se pueden hacer diagramas de Gantt detallados o trazar rutas críticas sobre una visión general.

Cuanto más se detallen los requisitos y el plan de ejecución, menores serán las posibilidades de modificación, y mayor resultará la previsibilidad de la ejecución del trabajo.

Por las circunstancias del proyecto



Estabilidad de los requisitos

Aparte de la necesidad o deseo del cliente, ¿Se puede obtener una descripción de requisitos detallada al inicio del proyecto?, y ésta, ¿se mantendrá estable durante el desarrollo?

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



Maleabilidad de la materia prima

¿Cómo de fácil es modificar el producto?

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

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



Coste de prototipado

El coste de prototipar podría verse como una característica independiente de la maleabilidad, pero suelen estar en estrecha relación, y en el caso del software, ambas son prácticamente la misma cosa.

La extrema maleabilidad de nuestra materia prima es la razón por la que también, resulta fácil desarrollar prototipos, no sólo ligeros (pantallas, dibujos, simulaciones), sino también pesados (código funcional y operativo).

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

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

A medida que el equipo va "tocando" y "probando" el sistema, surgen funcionalidades y posibilidades nuevas que aportan mayor valor al concepto inicial.

En este sentido, el argumento:"la forma más eficiente de desarrollar un trabajo es hacerlo bien a la primera", que se emplea con frecuencia para defender la validez de la gestión predictiva en cualquier proyecto, resulta tendencioso.

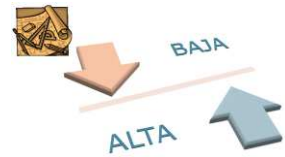
La afirmación "*per se*" es obviamente cierta; pero también son ciertas dos circunstancias relacionadas:

Se puede hacer "bien a la primera" cuando es posible conocer con detalle el resultado sin necesidad de realizar pruebas e investigación durante el desarrollo.

Las posibilidades al hacer un trabajo no son sólo "bien" o "mal".

Bien es un término amplio. Puede ser aceptable o suficientemente bien, o lo mejor posible.

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



Criticidad del sistema

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

Considerando por análisis de criticidad:

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

O lo que es lo mismo:

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

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

Si el sistema falla las consecuencias serán:

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

En este criterio hay diversidad de opiniones.

Hay quien considera que los modelos ágiles resultan poco adecuados para desarrollar sistemas críticos, y opiniones como la de Ken Schwaber, impulsor del uso de Scrum para desarrollo de software, que considera que sí que sería posible si se incluyeran unos requerimientos de conformidad

como entregables de cada iteración del desarrollo, junto con las funcionalidades a las que afectan.

Las críticas señalan que si se hiciera esto, se perdería el principio ágil de centrar el esfuerzo en partes funcionales, no en procesos de conformidad con procedimientos.

Con una posición rígida de "o procesos o agilidad", y considerando a los modelos (XP, Scrum, CMMI…) como patrones definidos, cuyas formas hay que respetar, las consideraciones de si esto que hago es ágil o no, interfieren y no aportan nada.

En este, como en todos los factores que estamos analizando, se trata de una cuestión de flexibilidad que hay que situar de forma adecuada a las características del proyecto.

A mayor nivel de criticidad del proyecto, mayor esfuerzo en tareas de validación y verificación del producto.

Considerar que no podemos incorporar prácticas de validación y verificación apropiadas para un sistema crítico, porque nuestro modelo es XP o Scrum es caer en la trampa de adoptar las formas de los modelos de forma ríquda.

O entrar en dudas sobre qué estamos empleando, ¿agilidad o procesos?, en un proyecto, en el que por requerirlo, empleamos prácticas de validación y verificación basadas en procesos…



Tamaño del equipo

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

Los grandes proyectos implican equipos numerosos y en ocasiones físicamente distantes, circunstancias que dificultan la comunicación directa.

No obstante hay desarrollos incipientes de prácticas ágiles que implantan esquemas de agrupamiento y comunicación directa en estructuras celulares de equipos de hasta 6 personas.

En este criterio hay diversidad de opiniones. Así mientras algunos textos opinan que el tamaño o la criticidad del sistema son aspectos muy relevantes, hay opiniones autorizadas en sentido contrario:

En la "International Conference on Complex Systems 2006", (Sutherland, Viktorov, & Blount, 2006)"Adaptive Engineering of Large Software Projects with Distributed / Outsourced Teams " se expusieron los buenos resultados obtenidos con prácticas de gestión ágil en un desarrollo de grandes dimensiones: un millón de líneas de código Java, y un equipo de 50 personas distribuidos en dos empresas distantes en países distintos.

Las prácticas de Scrum desarrolladas por Jeff Sutherland y Ken Schwaber para Software contemplan el desarrollo de sistemas críticos, incluyendo los requerimientos de conformidad como entregables en las iteraciones junto con las funcionalidades a las que afectan.

Por las circunstancias de la organización suministradora

Los elementos empleados por las organizaciones para el desarrollo de sus proyectos son: personas, procesos y tecnología.

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

Las personas tienen características propias:

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

Adoptar modelos de desarrollo ágil no consiste sólo en realizar las prácticas formales: equipo único, reuniones periódicas, desarrollo evolutivo de los requisitos, etc.

Si la organización mantiene un modelo de desarrollo basado en procesos y no en personas, y no tiene alineadas con los principios ágiles la cultura y estructura organizativa no obtiene los resultados propios del desarrollo ágil.



Cultura organizativa

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

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



Nivel profesional

"En el mundo del diseño informático, los mejores lo hacen entre 50 y 100 veces mejor que el promedio, y la cifra aumenta, conforme se incrementa la complejidad de la tecnología" (Jericó, 2001)

"La diferencia entre los promedios y los mejores ya no es de 1:2, como en el pasado. Es 1:100 o incluso 1:1000"[4]

Si el proyecto, más que innovación lo que requiere es ejecución controlada de un plan detallado, posiblemente sean los procesos de la organización los garantes del resultado, y con un modelo de gestión predictiva, el factor relevante es más la capacidad de los procesos empleados que un elevado nivel profesional de las personas del equipo.

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

[4] Nathan Myhrvold (Ex-director de I+D de Mircosoft)



Modelo de de desarrollo

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

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

MANAGEMENT EN EL DESARROLLO DE SOFTWARE

Modelos y metodologías: El mapa del bosque

¿Qué hacer con la crisis del software?

En 1968 (Bauer & Helms, 1968) se identificó al desarrollo del software como una actividad caótica en la construcción de grandes sistemas. Se acuñó el término "crisis del software" para definir la situación, y se acordó la necesidad de establecer procesos de ingeniería para el desarrollo de software.

Fue la primera vez que se habló de "Ingeniería del Software"

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

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

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

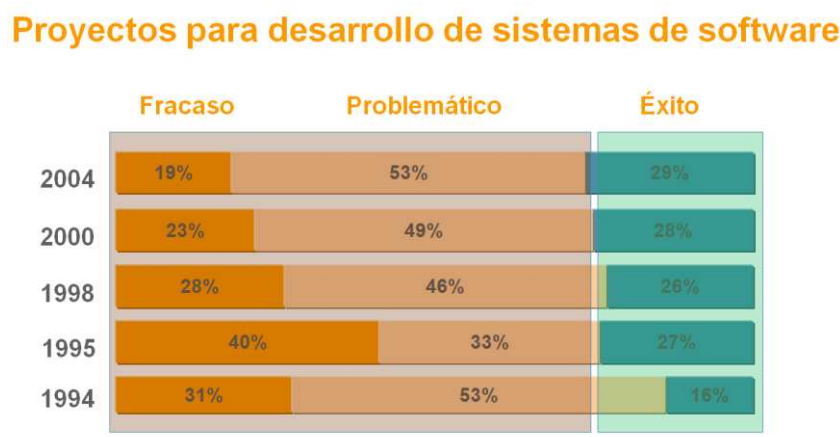
Para comprender mejor cuáles eran los conocimientos estables para desarrollar software en las fechas de la conferencia de la NATO, (Bauer & Helms, 1968) estas son algunas referencias útiles:

- En 1962 se publicó el primer algoritmo para búsquedas binarias.
- C. Böhm y G. Jacopini publicaron en 1966 el documento que creaba una fundación para la eliminación de "GoTo" y la creación de la programación estructurada.
- En 1968 los programadores se debatían entre el uso de la sentencia GoTo, y la nueva idea de programación estructurada;

ese era el caldo de cultivo en el que Edsger Dijkstra escribió su famosa carta "GoTo Statement Considered Harmful" en 1968. (Dijkstra, Marzo 1968)

- La primera publicación sobre programación estructurada no vio la luz hasta 1974, publicada por Larry Constantine, Glenford Myers y Wayne Stevens (Constantine, Myers, & Stevens, 1974).
- El primer libro sobre métrica de software fue publicado en 1977 (Gilb, 1977).
- Los primeros libros sobre análisis de requisitos aparecieron en 1979

Se puede decir que hasta bien entrados los 80 el uso de ordenadores era escaso, y ninguno el conocimiento desarrollado para gestionar proyectos de software.



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

- Se ejecuta en el tiempo planificado.
- Sin desbordar el presupuesto estimado.
- Satisficiendo las necesidades del cliente
  - Realiza las funcionalidades que necesita.
  - Las realiza correctamente y sin errores.

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

Sólo en uno de cada tres proyectos de software se cumple el plan inicial: el sistema realiza las funcionalidades inicialmente previstas, y se desarrolla sin desbordar ni agenda ni presupuesto.

¿El modelo predictivo es el único posible?.

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

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

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

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

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

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

Propuesta clásica.

En las fechas que se planteaba la necesidad de crear un conocimiento profesional para la gestión de proyectos, (v. origen de la gestión de proyectos pág. 21) también se estaba en la necesidad de crear una "ingeniería del software" para ofrecer garantías de calidad y previsibilidad al desarrollar programas.

Simultáneamente la teoría de *management,* o gestión empresarial, descubría las ventajas de la producción basada en procesos.



Ingeniería del software



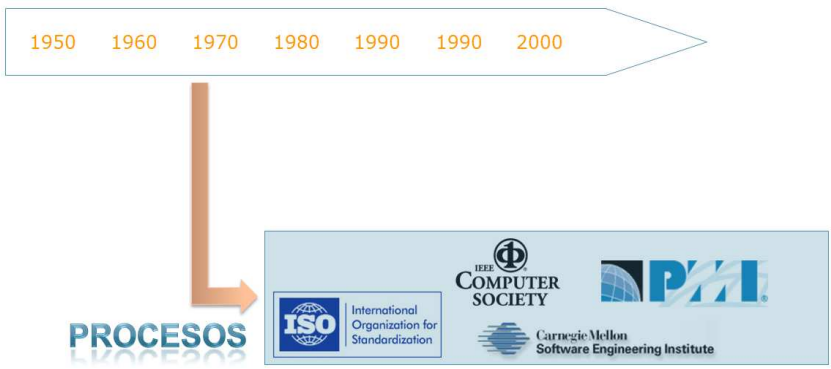
Gestión de proyectos predictiva



Producción basada en procesos

Fueron los tres ingredientes de la fórmula anti-crisis del software:

- Aplicar el conocimiento de gestión de proyectos que estaba cristalizando en esas mismas fechas.
- Desarrollo de conocimiento y pautas de ingeniería para lograr garantías de previsibilidad y calidad a su desarrollo.
- Producción basada en procesos.



Las principales organizaciones que trabajan en el desarrollo de pautas de ingeniería y gestión para el desarrollo de sistemas de software son:

- Instituto de Ingeniería del Software (SEI). Promovido por el Congreso Americano, y participado por el Departamento de Defensa y la Universidad Carneige Mellon para desarrollar métodos que permitan evaluar la capacidad de los desarrolladores de software.
- El organismo internacional de estandarización ISO.
- La organización profesional IEEE Computer Society

Las principales organizaciones que trabajan en el desarrollo de modelos predictivos de gestión de proyectos son las vistas en el tema Gestión de proyectos predictiva (pág. 23).

Aunque son muchas las líneas de trabajo e investigación, los principales referentes se han producido a partir de 1990 y son:

Por parte de ISO:

- ISO 9000-3 como modelo de calidad para desarrollo de software.
- ISO/IEC 12207 como modelo de procesos que intervienen en el desarrollo mantenimiento y operación de sistemas de software durante todo el ciclo de vida.
- ISO/IEC 15504 como modelo para la evaluación y mejora de los procesos de desarrollo y mantenimiento de software.



ISO 9000-3
ISO/IEC 12207
ISO/IEC 15504



CMM-SW
CMMI



PMBOK



ESTÁNDARES PARA LA INGENIERÍA DEL SOFTWARE

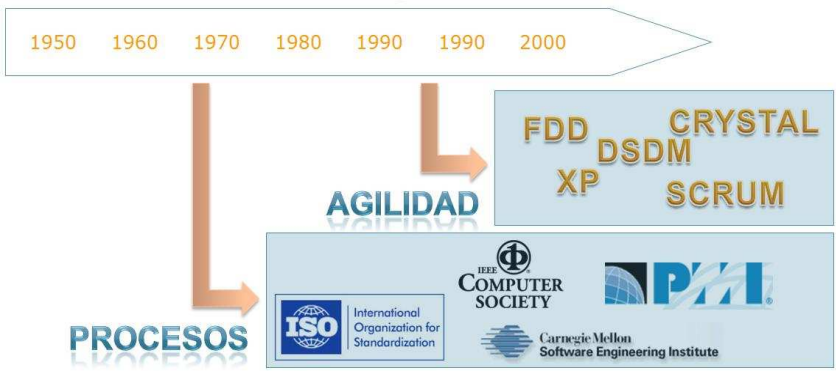
Por parte de SEI, la línea de modelos CMM para la evaluación y mejora de procesos, que comenzó con el desarrollo de CMM-SW (CMM for software) y ha evolucionado en estos 15 años en los actuales modelos CMMI (CMMI Models, Modules, and Reports).

La Computer Society (IEEE) ha desarrollado los estándares más conocidos para diferentes prácticas y procesos de la ingeniería del software.

El principal referente en gestión de proyectos clásica es el PMBOK (Project Management Body of Knowledge) desarrollado por el PMI (Project Management Institute).

Propuesta ágil.

A finales de los 90, reputados profesionales con predicamento en diferentes foros técnicos, comenzaron a cuestionar las metodologías formales, que representadas por CMM e ISO 15504, y respaldadas por la autoridad y los medios de sus respectivas organizaciones, estaban asentando una ingeniería del software basada en procesos.



En marzo de 2001, 17 críticos de estos modelos, convocados por Kent Beck, que acababa de definir una nueva metodología denominada *Extreme Programming*, se reunieron en Salt Lake City para discutir sobre los modelos de desarrollo de software (v .Manifiesto Ágil pág. 48).

En la reunión se acuñó el término "**Métodos Ágiles**" para definir a aquellos que estaban surgiendo como alternativa a las metodologías formales, a las que consideraban excesivamente "pesadas" y rígidas por su carácter normativo y fuerte dependencia de planificaciones detalladas, previas al desarrollo

El manifiesto ágil surgió con espíritu desafiante y beligerante contra los modelos edificados sobre procesos.

Los propios integrantes del manifiesto se auto-califican como "anarquistas organizacionales", y en los últimos años, desde uno y otro lado se han lanzado argumentos punzantes buscando, quizá más la descalificación ajena, que la justificación propia.

DSDM

Es la metodología más veterana de las auto-denominadas ágiles. Surgió en 1994 de los trabajos de Jennifer Stapleton, la actual directora del DSDM Consortium.

DSDM es la metodología ágil más próxima a los métodos formales; de hecho la implantación de un modelo DSDM en una organización le permite alcanzar lo que CMM consideraría un nivel 2 de madurez.

Sin embargo los aspectos que DSDM reprocha a CMM son:

- Aunque es cierto que se ha desarrollado con éxito en algunas organizaciones, lo que funciona bien en unos entornos no tiene por qué servir para todos.
- CMM no le da al diseño la importancia que debería tener.
- CMM plantea un foco excesivo en los procesos, olvidando la importancia que en nuestra industria tiene el talento de las personas.
- El tener procesos claramente definidos no es sinónimo de tener buenos procesos.

Extreme Programming

Es la metodología ágil más popular, y posiblemente también la más transgresora de la ortodoxia basada en procesos.

Su creador fue Kent Beck, impulsor del Manifiesto Ágil.

Extreme Programming (XP) se irgue sobre la suposición de que es posible obtener software de gran calidad a pesar, o incluso como consecuencia del cambio continuo. Su principal asunción es que con un poco de planificación, un poco de codificación y unas pocas pruebas, se puede decidir si se está siguiendo un camino acertado o equivocado, evitando tener que echar marcha atrás demasiado tarde.

Scrum

Scrum es una metodología ágil de desarrollo de proyectos que toma su nombre y principios de los estudios realizados sobre nuevas prácticas de producción por Hirotaka Takeuchi e Ikujijo Nonaka a mediados de los 80 (v. pág.37)

Aunque surgió como modelo en el desarrollo de productos tecnológicos, sus principios son válidos para entornos que trabajan con requisitos inestables, y necesitan agilidad: situaciones frecuentes en el desarrollo de determinados sistemas de software.

Jeff Sutherland aplicó los principios de los campos de Scrum al desarrollo

de software en 1993 en Easel Corporation (Empresa que en los macro-juegos de compras y fusiones se integraría en VMARK, luego en Informix y finalmente en Ascential Software Corporation). En 1996 presentó junto con Ken Schwaber, las prácticas que empleaba como válidas para gestionar el desarrollo de software OOPSLA 96 (Schwaber & Sutherland, 1996)

Otros modelos o prácticas ágiles.

DSDM, XP y Scrum son, posiblemente, las prácticas ágiles más conocidas, y como sobre los principios del manifiesto ágil, son muchas las que se pueden combinar y diseñar para responder mejor a las particularidades de cada empresa y proyecto, son también muchas las propuestas diseñadas y contrastadas por profesionales en entornos que requieren innovación y velocidad:

- Crystal: Adaptative Software Development (ASD)
- Pragmatic Programming (PP)
- Agile Modeling (AM)
- Feature Driven Development (FDD)
- Etc.

Estas metodologías ágiles no deben considerarse como modelos de procesos completos.

CMMI o ISO 155504 sí que lo son. Cubren, basándose en procesos, todas las actividades implicadas en el ciclo de vida de un sistema de software.

Cada metodología ágil surge de la difusión de una o unas "buenas prácticas" diseñada y utilizada con éxito por su autor, y contrastado éste también por los equipos que la van incorporando.

Por eso, en cada caso, son conjuntos de prácticas focalizadas sobre un área de conocimiento, más o menos delimitado, de la Ingeniería del Software. Las prácticas de Agile Modeling cubren básicamente tareas técnicas de diseño; Las de Scrum diseñadas por Sutherland y Schwaber se centran en la gestión del desarrollo; Extreme Programming cubre las actividades que desde el plano completo de la ISO 12207 pertenecería al desarrollo, etc.

Por esta razón es frecuente combinar varias de estas prácticas (ej. XP + Scrum + FDD) para dar cobertura ágil a un ámbito más amplio del ciclo de vida.

Software, personas y procesos

INTRODUCCION

Según como afrontan las organizaciones el desarrollo del software, éste puede comportarse como factor de riesgo o amenaza para el negocio; o por el contrario como una poderosa oportunidad de negocio.

Todas las empresas quieren producir más rápido, mejor y con menores costes, y es posible, porque la naturaleza del software, antes que causa de riesgos y problemas, es una fuente de oportunidades.

Cada vez más directivos están comprendiendo que el modo de gestionar el desarrollo de software, puede hacer de la materia prima de su negocio un material arisco de resultados impredecibles, o una ventaja competitiva.

La evolución hacia entornos de ingeniería del software requiere cambios severos en la organización, así como el convencimiento, implicación y empuje de la dirección. Pero sobre todo el diseño de un modelo de producción propio, que sepa aprovechar la personalidad de la organización, y responder a las particularidades de su negocio.

El software ha estado repitiendo los mismos problemas en los últimos 40 años, y quien no cambia la forma de hacer las cosas, sigue tropezando en ellos.

El problema que pueden encontrar quienes deciden implantar métodos más adecuados es caer en la desorientación ante el abanico de modelos de calidad, de procesos y de técnicas de trabajo desplegado en la última década, o abrazar al primero que se presenta en la puerta de la organización como “solución” de eficiencia y calidad.

La madurez de los procesos

Las organizaciones que desarrollan o mantienen software pueden optar por trabajar "a la buena de Dios", o por seguir una metodología. Hacerlo a la buena de Dios no es tan raro. Es la primera forma que se empleó para desarrollar programas:
"Aquí tenemos unos ordenadores, aquí unos señores a los que les encanta leer los manuales de programación, y se trata de conseguir que estas

máquinas terminen imprimiendo facturas (o haciendo lo que sea)".

En realidad, cuando en la segunda mitad del siglo pasado la industria del hardware puso en la calle máquinas que se podían programar, poco más se podía hacer; y los pioneros de nuestra profesión, atraídos por el encanto de diseñar y construir artefactos útiles, y verlos funcionar, fueron los primeros en remangarse frente al teclado y decir a su cliente con una sonrisa:"en un par de meses esto estará funcionando".
Fueron también los primeros en pasar noches en vela programando, prometiendo una y otra vez que "tan sólo es cuestión de un par de días más."

Aunque las cinco décadas de vida del software ya han sido suficientes para experimentar los excesos y los errores de la infancia y la juventud, la resistencia al cambio es el mejor aliado de la inercia, y produce una cierta impermeabilidad a la experiencia.
En cualquier caso es una opción: trabajar sin ninguna metodología.

SEI, (Software Engineering Institute) por el rigor de su documentación y la circunspección de su imagen no puede llamar a esta forma de producir como "a la buena de Dios", y en su lugar afirma que es la forma propia de organizaciones "poco maduras".

En la escala, que de 1 a 5 emplea para determinar el grado de madurez de una organización, y en consecuencia el nivel de garantía que ofrece en cuanto a calidad, predictibilidad en los resultados, y eficiencia en el desarrollo de software, este tipo de organizaciones se quedan, lógicamente, en el 1.
Que, ¡ojo cuidado!, no quiere decir que necesariamente van a producir mal software, de forma ineficiente y con retraso, sino que las probabilidades de cumplir las tres facetas es baja.
Hay equipos que lo consiguen, pero son pocos. La razón es sencilla, los resultados en estos casos son tan buenos como las personas que los desarrollan, y lo cierto es que los buenos programadores escasean.
A este modo de producción se le ha venido llamando "programación heroica", porque todo el peso del resultado descansa sobre el "saber hacer" de los programadores.
El éxito o fracaso de las organizaciones que trabajan sin metodologías depende del conocimiento tácito de su personal, pero teniendo en cuenta que se trata del conocimiento que cada uno traía ya de la calle, o del que adquiere *motu proprio*, porque estamos hablando de "ninguna metodología", lo que implica que como mucho los procesos de formación de la empresa llegan al "ahí tienes manuales y libros, por si hubiera alguna cosa que no sabes".

Este es sin duda el modelo con el que empiezan muchas *start-up*: un equipo de emprendedores con talento, capaces de construir sistemas de software con calidad.
Los resultados serán tan buenos como ellos los sepan hacer. El cumplimiento de agendas dependerá de su capacidad de previsión y organización. Pero no hay que engañarse; en este caso no se trata de empresas que saben hacer software, sino de personas que saben hacer software.
Y esta situación dibuja el guión común a todas las pequeñas empresas de programación que surgen de cero, impulsadas tan sólo por el empuje de sus emprendedores: pueden llegar todo lo lejos que la combinación del talento y de la capacidad de marketing de sus creadores les permitan.
O sea, en ocasiones acabarán cerrando, y en otras alcanzarán un nivel de estabilidad tan mediocre o tan brillante como dicha combinación les permita. Y si una vez logrado ese nivel de equilibrio desean proyectar mayor crecimiento a su organización se enfrentan a un reto importante:

Pasar de un grupo de personas que saben hacer software a une empresa que sabe hacer software.

Salto que supone que no van a ser ya ellos, sino su empresa quien deberá producir de forma eficiente y repetible en todos sus proyectos, resultados de calidad. Y esta es otra aventura en la que muchos fracasan teniendo que regresar a la casilla de salida, o si el "roto" del intento ha sido muy grave, terminar cerrando, o como mal menor dejándose adquirir por algún competidor más o menos grande del sector.

La teoría de la gestión empresarial recomienda para esta travesía tomar como medio de transporte la gestión por procesos.
Sí, no es nada nuevo, y leerlo a estas alturas puede dar la sensación de estar desayunando con el periódico del día anterior.
Aparte de las excelencias recogidas en el manual del consultor sobre la transversalidad de los procesos, que atravesando a toda la organización dibuja modelos organizacionales sin rigideces departamentales, y alinea con sus flujogramas a toda la empresa en el mismo sentido con visión centrada en el cliente, y bla, bla, bla.

Lo cierto es que la gestión por procesos encierra cuatro factores que hacen posible pasar de grupo de emprendedores a empresa:

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

No sólo son procesos

Los procesos marcan pautas para realizar el trabajo, pero sin las personas y las herramientas (tecnología) necesarias, lo que se puede producir con ellos es más bien poco.
Las únicas combinaciones válidas para formar sistemas capaces de producir resultados son:

Personas + tecnología	Producción heroica
Personas + procesos + tecnología	Producción basada en procesos

Un ejemplo que seguro todos hemos realizado alguna vez, y que ilustra las diferencias entre los dos modos de producción es el montaje de un mueble en *kit*.

Para enfrentarnos a esta tarea tenemos, por un lado a las personas: nosotros mismos; y por otro la tecnología: el destornillador (manual o eléctrico), los tornillos, pasadores, la cola de montaje, etc. Y si dejamos el proceso a un lado (el manual con las instrucciones de cómo proceder) tendremos un sistema perfecto de producción heroica: personas + tecnología; del que cabrá esperar los resultados propios de estos entornos.

La productividad y la calidad no son homogéneas. Hay quien ensambla el mueble, en cuestión de minutos, y quien necesita toda una tarde. Algunos obtienen estanterías robustas y perfectas, y otros terminan el montaje con peor fortuna, afirmando que está bien montada pero las piezas venían mal cortadas de fábrica, o que quien diseñó ese mueble era un perfecto inútil, o que los tornillos son de mala calidad; o quizá sin ni siquiera percatarse de los defectos con los que ha dejado el mueble.

En definitiva esta forma de trabajo produce resultados, y puede resultar más o menos apropiada para realizar hobbys o bricolajes domésticos, pero en una empresa de montaje de muebles no resultaría viable un sistema de producción que no pudiera garantizar eficiencia en los costes, y resultados con una calidad homogénea.

Hay dos formas de evitar estos problemas:

- Introducir un tercer elemento en el sistema de producción: los procesos y explicitar en ellos el conocimiento necesario para obtener el resultado con el grado de calidad necesario.

- Si el conocimiento necesario se centra en las personas y la tecnología, emplear sólo a los mejores ebanistas con las mejores herramientas.

Como veremos a continuación generalmente la mejor solución no suele consistir en adoptar una u otra opción, sino una mezcla de ambas, con mayor o menor peso específico en una u otra, en función de las características del negocio.

Al añadir procesos al sistema se consigue que todas las personas ejecuten el trabajo siguiendo las mismas pautas, y que los parámetros de ejecución y los resultados puedan ser medidos de forma homogénea en toda la organización.

De esta forma se reduce drásticamente la heterogeneidad de los resultados, tanto en la eficiencia de los tiempos invertidos, como en la calidad obtenida, que empieza a ser proporcional a la capacidad de los procesos que se hayan diseñado.
Pero no hay que dejarse deslumbrar por los procesos y olvidarse de los otros componentes.
Basar una empresa sobre un sistema de producción de dos elementos: personas y tecnología plantea limitaciones; pero basarlo en un entorno exclusivo de procesos, o de procesos y tecnología, simplemente no es posible.

La realidad de los procesos es cierta, pero en el triángulo personas - procesos - tecnología cada elemento actúa con un peso específico, diferente en función del tipo de producción, e incluso de las características de personalidad de cada empresa.

Como veremos más adelante, la perspectiva de la flexibilidad reconsidera el modelo clásico de tres elementos, porque razona que "personas" y "procesos" no son elementos simples, sino que cada uno de ellos tiene dos componentes.

Llegados a este punto, el ámbito de la teoría generalista se acaba, y cada sector y cada empresa, más que importar un modelo estándar sacado del manual del perfecto consultor, debe comenzar por el *gnosce te ipsum*, y a partir de ahí analizar y descubrir la potencia de cada uno de los tres elementos de la producción para diseñar "su" sistema de producción.

Los gestores o consultores "estándar" no existen; bueno, perdón, sí, pero obtienen los resultados estándar que todos conocemos.

Relevancia del capital estructural y relevancia del capital humano

El capital estructural lo forman los bienes que quedan en la empresa cuando las personas se van a sus casas: patentes, licencias, cartera de clientes, equipos, maquinaria, vehículos, etc.

El capital humano es el compuesto por el valor que emplea en su negocio y que resulta inseparable de las personas.

Todas las industrias necesitan ambos tipos de capitales para elaborar los productos o servicios de su negocio, pero la relevancia con que cada uno puede influir en el resultado final es muy diferente de unas empresas a otras.
Es frecuente que en las dedicadas a la fabricación de productos, el componente estructural sea más crítico que en las de servicios, donde el elemento humano tiene más relevancia.

Pero éste no es un principio universal, y dentro de la misma industria o del mismo sector, la estrategia y la táctica de cada empresa también influyen en los niveles de relevancia de cada tipo de capital.

Veámoslo con un ejemplo, comparando el peso de cada uno en dos empresas del mismo sector: hostelería.

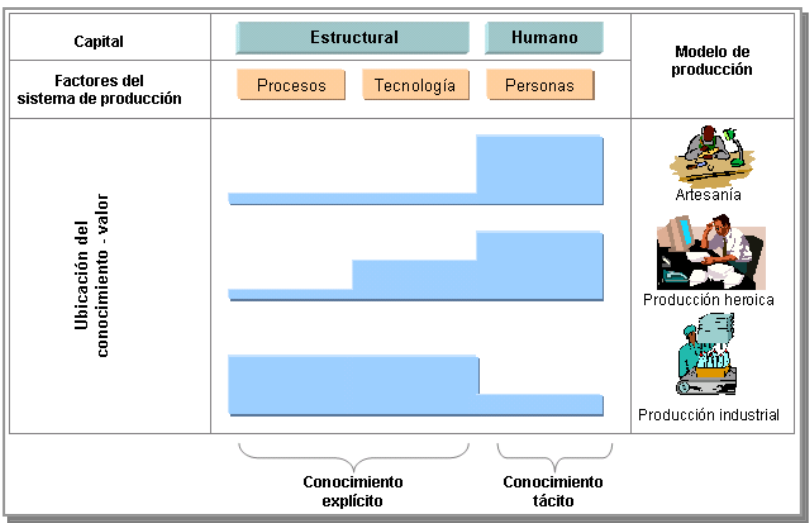
Por un lado pensemos en un exquisito restaurante de cocina vasca (por ejemplo), y por el otro en un establecimiento de *PhonoPizza* (también por ejemplo).

Al margen de consideraciones gastronómicas, ambos negocios tienen perfectamente claros su identidad, personalidad y segmento; ambos tienen también su propio patrón de atributos de calidad.

Los procedimientos de PhonoPizza son los que garantizan de forma continuada la calidad y repetitividad de sus resultados, y éstos tan apenas dependen el conocimiento tácito de sus cocineros.

Los hornos regulan automáticamente el tiempo y la temperatura, los ingredientes se producen también en base a procesos, y se distribuyen a todos los establecimientos en estuches, ambientes frigoríficos y medios de transporte ejecutados por procesos que son los principales responsables de los resultados.

Por esta razón resulta mucho más fácil, o en lenguaje empresarial, menos costoso, reemplazar al personal de cocina de un establecimiento de *PhonoPizza* que al de un restaurante de cocina vasca.



Las barras de color azul de la figura representan el valor, que para una empresa determinada, aporta cada elemento a su sistema de producción.

Las combinaciones posibles para cada negocio son muchas, y factores como la innovación en la tecnología, en los procesos o la formación del personal pueden abrir la horquilla de potencial de cada elemento.

La homogeneidad y calidad de los resultados, y la eficiencia del sistema en su conjunto serán consecuencia del equilibrio logrado.

Por supuesto una franquicia de hamburguesas y comida rápida podría incluir, además de tecnología y procesos eficientes, la contratación o formación de gourmets de alta cocina; pero no es una buena forma de aprovechar la naturaleza de su producto como ventaja competitiva.

La obsesión por la excelencia mal entendida, o la desorientación sobre las funciones del propio puesto lleva a muchos gestores a centrar sus esfuerzos en la incorporación de modelos o técnicas, bien de procesos, bien de recursos humanos o de innovación tecnológica; o de todos a la vez, por razones tan poco solventes como "ser lo último en management", o lo estudiado en el último seminario, o lo que con tanta vehemencia le ha "vendido" el último pseudo-consultor que visitó su empresa...

Muchas veces, esta es la razón del escepticismo y las críticas que finalmente verterán sobre modelos, métodos o técnicas contrastadas y eficaces; cuya debilidad no radica en ellas sino en interpretaciones tergiversadas, implementaciones incorrectas, o en sistemas donde no resultan apropiados.

Conseguir que el sistema de procesos, tecnología y personas formen una ventaja competitiva frente a la competencia, y no un reto más del negocio, no es fácil, ni puede importarse con la implantación "prêt-à porter" de un modelo estándar.

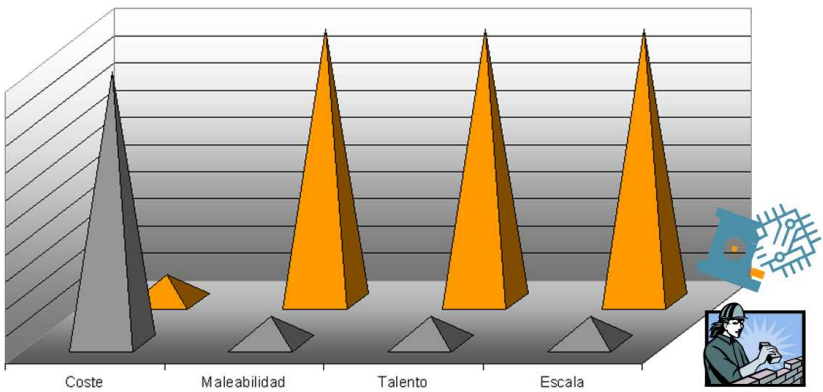
La clave del éxito es conseguir que cada factor aporte al conjunto hasta el límite de su mejor relación eficiencia/calidad.

En esta tarea nunca está dicha la última palabra, y la labor de innovación constante en procesos y tecnología puede ir ampliando los límites de valor a un factor, y dibujar un nuevo equilibrio con mejores parámetros de eficiencia y calidad en el sistema.

Las características del Software

Hay cuatro características clave, que hacen a las empresas de software muy diferentes a las de producción industrial, y a sus proyectos, de los de de ingenierías civiles:

- Coste de la materia prima.
- Maleabilidad del producto.
- Valor aportado por las personas.
- Factor de escala del producto.



Coste de la materia prima

Un ingeniero de software puede desarrollar con recursos muy escasos el sistema de sus sueños; sin embargo un arquitecto o un ingeniero naval no pueden hacer lo mismo.

Los recursos físicos necesarios para la construcción de sistemas, en ninguna ingeniería son despreciables, excepto en la ingeniería de software.

Esta característica tiene implicaciones importantes en la gestión, porque permite considerar opciones, que hasta la aparición del software, la gestión de proyectos ni se planteaba.

Sería absurdo considerar para la construcción de edificios, ir edificando en pequeños incrementos, revisar lo hecho sobre el terreno, y derribar lo que se considere erróneo o mejorable, para rehacerlo de una nueva forma.

Maleabilidad

Los materiales físicos no son muy maleables. Esta característica refuerza la necesidad de previsión antes de comenzar la construcción de un sistema, porque una vez desarrolladas cada una de las partes o módulos, no es posible su cambio, amoldando lo ya construido.

Pero el software no es físico, la maleabilidad puede ser muy elevada si se tiene en cuenta como opción de desarrollo, para acometer ampliaciones y modificaciones del sistema con técnicas de re-factorización.

Maleabilidad y coste permiten plantear ciclos de construcción ágil, a pequeños incrementos en iteraciones breves, durante las que se van refinando y descubriendo los requisitos.

Un modelo de gestión, que parecería absurdo, no lo es tanto si caemos en la cuenta de que las características de nuestra industria son diferentes a las del resto.

Valor aportado por las personas

La visión de gestión Scrum, ayuda a comprender los marcos de producción, porque no considera a "procesos" y "personas" como factores simples, sino que cada uno de ellos en realidad es una combinación de dos elementos.

A los "procesos" los llamaríamos mejor "procedimientos", y éstos pueden ser "procesos" (conocimiento) o "rutinas" (ayuda). En realidad, habitualmente son una combinación de ambos; y en cada caso en proporciones diferentes.

El factor "personas" también es una mezcla de "ejecución" y "talento"; y la eficiencia de cada situación reclama su propia proporción en la combinación.

Con este criterio, en algunas organizaciones puede tener sentido para gestionar el factor de producción "personas" un departamento de recursos humanos, pero en otras la gestión de este factor puede ser más adecuada con un departamento de gestión del conocimiento o del talento.

Cuando el grueso del conocimiento reside en los procesos, y las personas aportan capacidad de ejecución en forma de tareas mecánicas o supervisión, la diferencia de productividad o calidad si el trabajador es mejor o peor, no es muy significativa.

En la cocina de un Telepizza, que el operario tenga más o menos talento influye poco en el resultado.

Sin embargo, cuando el grueso del conocimiento reside en las personas, que aportan el talento, y los procedimientos en este caso son quienes ayudan a las personas ("rutinas"), las diferencias de los resultados entre los mediocres y los mejores son muy importantes.

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

Factor de escala

Algunos productos tienen factores de escala escasos: aumentar el número de unidades realizadas, supone incrementos de proporciones similares en el coste de fabricación.

El software tiene un factor de escala prácticamente infinito: cuesta lo mismo producir uno que mil.
Si mi tratador de textos, o mi solución de correo, o de seguridad o… es la mejor del mercado, venderé, cien, mil o diez mil veces más programas, con la misma inversión de desarrollo.

Algo que sin duda no le ocurre, por ejemplo a una empresa de construcción, o de joyería.

¿Cómo afecta esta característica a la ratio coste de prototipado / valor del producto? (pág. 73).

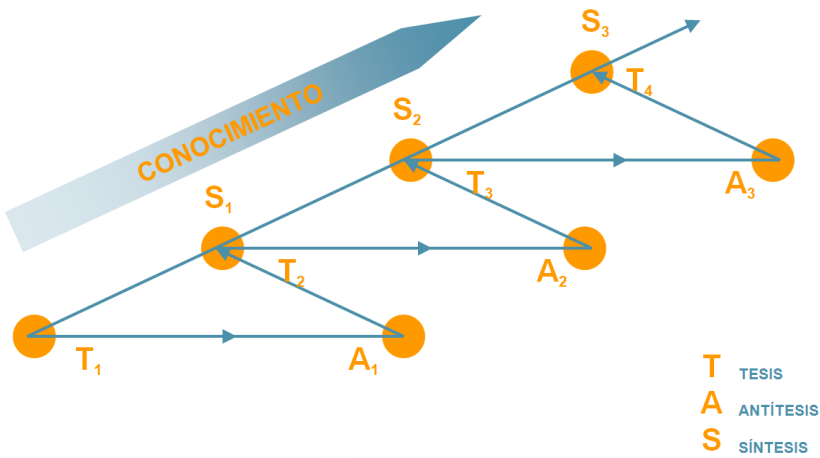
Si el producto necesita el máximo valor posible, un modelo de plan y requisitos cerrados, para hacerlo bien a la primera, y evitar costes de modificaciones; puede ser un ahorro miope. El cuestionamiento, los requisitos siembre abiertos, la apertura al cambio, y la inyección de valor continuo durante todo el desarrollo es una gran inversión.

Scrum Management: Síntesis, flexibilidad y gestión sistémica

El cuestionamiento de lo conocido es el motor de la evolución del conocimiento.

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

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



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

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

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

En los 90, llega la difusión y aplicación de este conocimiento. En algunos ámbitos da buenos resultados, y en otros genera la réplica "dialéctica": El Manifiesto Ágil, que cuestiona la validez de los modelos basados en procesos y la gestión predictiva para el desarrollo de software.

Se radicalizan las posturas entre ambas líneas y se genera (y se está generando) la tensión entre contrarios que mueve la evolución dialéctica del conocimiento.

Algunos ejemplos de esta tensión:

"*La diferencia entre un atracador de bancos y un teórico de CMM es que con el atracador se puede negociar*"…

"La evaluación en CMM depende más de una buena presentación en papel que da la calidad real del producto de software. Tiene que ver más con el seguimiento a ciegas de una metodología que con el desarrollo y puesta en producción de un sistema en el panorama tecnológico".

Ken Orr , CMM versus Agile Development: Religious wars and software development .

"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".

Richard Turner y Apurva Jain, Agile meets CMMI: Culture Clash or Common Cause .

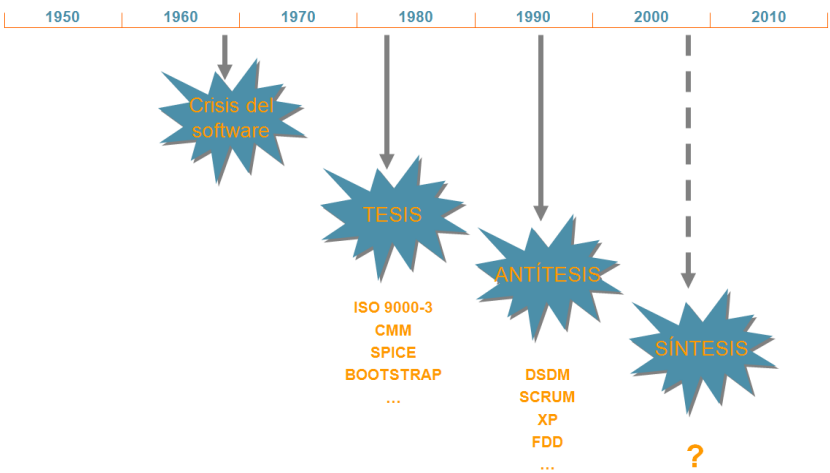
En los últimos años se apuntan ya las tendencias de la evolución hacia la síntesis:

En estos momentos autoridades de la Ingeniería del Software como Barry Boehm y Richard Turner hablan de balancear la agilidad y la disciplina (Boehm & Turner, 2004)

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

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

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



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

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

La contradicción produce desconcierto, pero además en nuestro caso, la velocidad de comunicación facilitada por Internet, y el apresuramiento general del entorno, hace que se solapen las tres tendencias del ciclo.

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

Estrategia de Scrum management: Síntesis + Flexibilidad + Gestión Sistémica

SÍNTESIS

¿Qué es mejor, el yin o el yang? ¿La tesis o la antítesis? ¿Los procesos o la agilidad?

Dar la espalda a la gestión predictiva es trabajar con un fondo de conocimiento incompleto; tan incompleto como emplear la gestión predictiva dando la espalda a los valores y prácticas de los modelos ágiles.

FLEXIBILIDAD

Cada proyecto tiene sus propias circunstancias: estabilidad del entorno, componente de innovación, grado de criticidad del producto que debe construir, cultura de la organización que lo desarrolla, etc.
No hay dos proyectos iguales, ni dos empresas iguales; ni por tanto, una "talla única" de gestión de proyectos.

La cuestión no es determinar si el modelo “bueno” para el software es PMI o Scrum, CMMI o Extreme Programming.

La Flexibilidad se consigue al conocer ambos y emplear la forma más adecuada a las circunstancias de cada proyecto o de cada empresa.

Se consigue al combinar y emplear modelos y prácticas desde el conocimiento de su "fondo" además del de su forma, para lograr la mejor "talla" para la empresa.

> ***Una Scrum Management no se queda en el "cómo" de las prácticas, sino que trabaja desde el "porqué" para descartar, modificar o incorporar según las características de la empresa y los proyectos.***
>
> ***"***

Un "gestor flexible", un "ScrumManager" tiene experiencia de trabajo en su industria, y además una visión sintética de la tesis y la antítesis; y cuanto mayores sean ambos mejor gestor será.

GESTIÓN SISTÉMICA

La empresa es un sistema inter-relacionado, y es importante que tenga una identidad definida con una visión y misión concretas, porque de esta forma será el punto para dar coherencia y alineación a todos los departamentos y personas.

Cuanto más nítida sea la visión, misión, estrategia segmento de mercado y objetivos de la organización, con mayor tino y precisión se podrán combinar los componentes del sistema, y orientar la gestión de su funcionamiento.

Aplicar una gestión sistemática: que al actuar considere a la organización como un todo relacionado, y no sólo el departamento o área en la que cada momento actúe.

Las áreas comerciales, de programación, de recursos humanos, de marketing, etc. no deben actuar de forma aislada. El tipo de producto o servicio que se desarrolla, los criterios para contratación de personal, el modelo de calidad, las prácticas de programación, los procesos de presupuesto y contratación con clientes forman parte del mismo sistema, y la falta de consistencia entre ellos genera empresas fragmentadas.

La implantación de un modelo de producción Scrum, Extreme Programming o CMMI, no es una cuestión del responsable de programación, ajena al resto de la empresa.

Las técnicas empleadas, los modelos de calidad, la tecnología, la cultura de la empresa, la gestión de las personas… deben guardar coherencia, para que se potencien, y no se contrarresten.

Un "*know-how*" adecuado, basado en el conocimiento tácito de las personas, compaginado con equipos desmotivados no es una buena combinación.

Procesos y personas en la nueva estrategia de gestión.

Tanto los procesos como la agilidad, se basan en la premisa de que son tres los elementos en un entorno de producción: Las personas, los procesos y la tecnología.

Las teorías de producción basadas en procesos funcionan bien en entornos de producción industrial, y de ahí los tomaron los pioneros de la ingeniería del software.

Estos modelos trabajan sobre la premisa de que la calidad y la eficiencia de la producción se deben sobre todo a los procesos empleados.



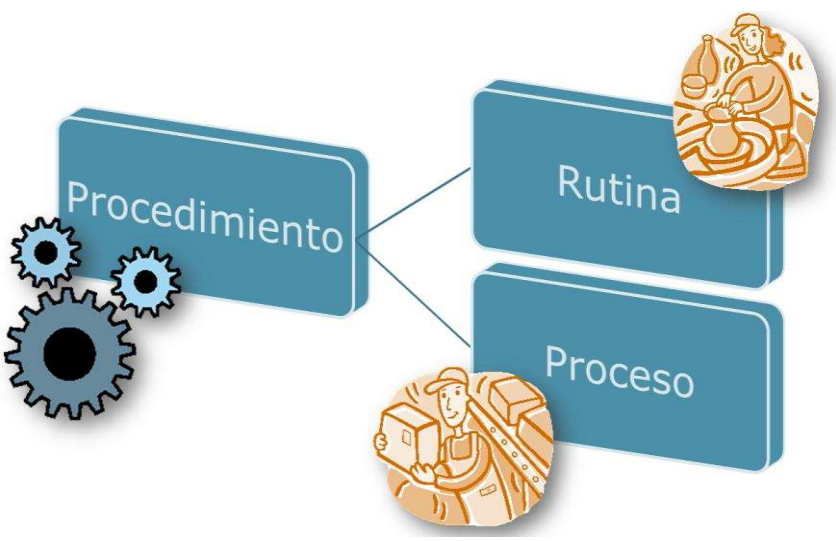
Este es el principio sobre el que Watts Humphrey desarrolla las bases de los modelos de madurez CMM - CMMI.

El Manifiesto Ágil, sin embargo, afirma en su primer enunciado que el protagonismo deben tenerlo las personas, y no los procesos.



Es posible flexibilizar las prácticas y modelos conocidos a la identidad de nuestra empresa, si consideramos que no todo lo etiquetado como procesos es la misma cosa.

En unos casos las personas ayudan al proceso, y en otros son los procesos los que ayudan a las personas.



En el primer caso el proceso es el protagonista; el que sabe cómo hacer el trabajo y la persona se integra en el sistema como instrumento, como operario de apoyo. En el segundo el artífice es la persona y el proceso una

ayuda, una herramienta que simplifica aspectos rutinarios para que pueda lograr más eficiencia y no diluir el esfuerzo en rutinas mecánicas.

La principal diferencia entre unos y otros es el tipo de conocimiento con el que trabajan. La clasificación entre explícito (contenido en los procesos y la tecnología), y tácito (contenido en la persona) (Ikukiro & Hirotaka, 1995) .

Desde nuestro punto de vista llamamos "procesos" a los que operan en sistemas de conocimiento explícito, y "rutinas" a los que ayudan a las personas en sistemas de conocimiento tácito.

Cuando el conocimiento, necesario para producir el resultado radica (en su mayor parte) en la tecnología empleada y el procedimiento; éste es un "proceso", y las personas no aportan "conocimiento", sino "ejecución". La persona ayuda al proceso.

Como ya apuntábamos en el ejemplo de la pizzería (pág. 97) el conocimiento para que la comida se haga en su punto, está en la tecnología y los procesos de trabajo. Las personas no tienen por qué saber hacer pizzas, sino ejecutar el proceso.

En un restaurante de alta cocina el conocimiento de cómo hacer las recetas está en las personas. En este caso, la tecnología y los procedimientos (rutinas) son los que les ayudan a ellos.

Esta es una de las conclusiones de la síntesis entre procesos y disciplina (tesis) por un lado, y agilidad (antítesis) por otro.

Para los primeros, el valor de los resultados es consecuencia principalmente de los procesos empleados.

Para los segundos, la variable más influyente en el valor del resultado son las personas.

¿Quién tiene razón: la tesis (CMM, ISO 15504, PMI...) o la antítesis (XP, Scrum, DSDM...)?
Ambos la tienen, y lo que su síntesis revela es que los procesos y las personas, que cada uno gestiona de forma diferente, encierran dos aspectos diferentes en cada caso, y no uno sólo.

Los llamados "procesos" en realidad pueden ser "procesos" o "rutinas" de trabajo.
El valor que cada sistema de desarrollo necesita de la persona puede ser trabajo o conocimiento.

Desde esta perspectiva surgen los principales criterios para diseñar y gestionar marcos de trabajo adecuados a las circunstancias, y se explica por qué y cuándo es adecuado diseñar un modelo de procesos con trabajo, y cuándo trabajar con conocimiento y rutinas; y porqué chirrían las prácticas conocidas al combinar en un sistema rutinas con trabajo, o procesos con conocimiento.

Y esta es una simplificación en blanco y negro de una realidad en color. Son raros los sistemas que encierran todo el conocimiento sólo en procesos y tecnología, o sólo en las personas, pero por eso mismo también son minoría los entornos de producción en los que puede encajar como solución de talla única un modelo basado en procesos o un modelo ágil "tal cual".

Scrum Management

Con este nombre definimos la estrategia de gestión que trabaja con la síntesis del conocimiento desarrollado por las teorías de procesos y las de agilidad.

De ésta toma el patrón de Campo de Scrum (pág. 40) como entorno para el desarrollo de los proyectos.

De aquella toma los beneficios de la mejora continua y la institucionalización de los procedimientos (que no de los procesos).

De la síntesis extrae que personas y procesos no son elementos simples, sino combinaciones "ejecución-talento" en el primer caso, y de "proceso-rutina" en el segundo; y con este criterio base flexibiliza los procedimientos y las estrategias de gestión, adoptando el grado más adecuado entre agilidad y disciplina para cada sub-sistema de la organización, tras analizar cuanto tiene el factor "personas" de ejecución y cuánto de trabajo; y qué grado de proceso y de rutina tiene el factor "procedimiento".

Es una forma de gestionar que no implanta procedimientos y prácticas de talla única pre-diseñada.

Los criterios de síntesis, y por tanto de referencia son:

- Beneficio del trabajo en equipos pequeños (productividad, comunicación directa)
- Desarrollo incremental e iterativo (producción frecuente de partes del producto que puede evaluar el cliente, integración y pruebas tempranas)
- Diseño de procesos o rutinas en función de la principal necesidad del proyecto: previsibilidad o creatividad e innovación.
- Grado de institucionalización de los procedimientos (procesos o rutinas) adecuado al tamaño y previsión de crecimiento de la organización.
- Gestión sistémica de la organización.

Scrum Management es un modelo de gestión de "conocimiento tácito". El gestor no aporta "ejecución" para implantar un modelo, sino "talento" para desarrollar el propio.

No consiste por tanto en aplicar un modelo, a modo de proceso, sino que son los gestores los que lo construyen con su conocimiento, sobre estas bases.

Los criterios de la gestión flexible en el Software

Introducción

La tesis (ISO 12207, 15504, CMMI…) ha identificado y definido QUÉ cosas son las que hay que hacer.
La antítesis (XP, Scrum, FDD...) ha mostrado formas de trabajar: CÓMO hacer determinados "qués".

Posiblemente la síntesis, la moraleja de ambos sea cuestionar los "POR QUÉs".

¿Qué hacer para que los proyectos de software salgan bien? La Tesis:

"Bueno..., tenga en cuenta que no sólo se trata de programar de forma adecuada. Hay que hacer de forma adecuada otras muchas tareas que tienen lugar desde que a alguien se le ocurre la idea de hacer algo, hasta que finalmente, y después de hecho y empleado, se deja de usar."

Esto es lo que dice ISO con su estándar internacional ISO/IEC 12207 (ISO/IEC, 2004): Las cosas QUE intervienen en el desarrollo de un proyecto de software.

QUÉ tareas tiene que hacer el cliente al adquirir el sistema (5.1 Acquisition), QUÉ tareas debe hacer el suministrador para responder a la adquisición (5.2 Suply), QUÉ tareas para desarrollar el sistema (5.3 Development) etc.

ISO 12207, CMMI o ISO 15504 son modelos. Los modelos dicen el QUÉ hay qué hacer, pero no CÓMO hacerlo. Los modelos le dicen: hay que hacer un contrato, hay que mirar las opciones del mercado, hay que hacer requisitos, plan de proyecto, análisis, codificar, probar, etc...

Ni CMMI ni ISO prescriben formas sobre CÓMO hacer la gestión de la configuración, la planificación del proyecto o la gestión de los requisitos. CMMI dice: hágalo como usted quiera, siempre y cuando alcance el fin del área de proceso que venga al caso. O sea, siempre y cuando usted

tenga control de las versiones, sepa cuál es el plan de proyecto, o los requisitos de lo que debe hacer.
Que usa gráficos de Gantt, o "post-it" en pizarras… allá usted. Que si documentos Word con el estándar de requisitos IEEE 830, o historias de usuario... Usted sabrá.

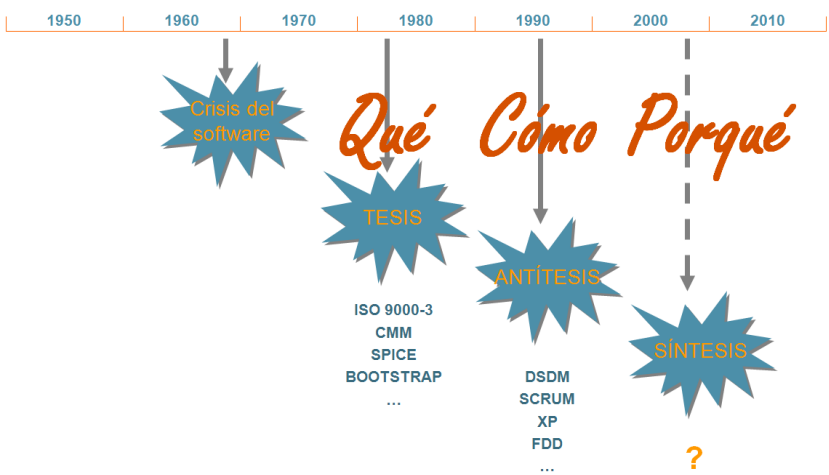
Usted lo hará bien, si consigue el objetivo del área de proceso. Bueno, otra cosa será si quiere que yo lo certifique, porque entonces me lo tendrá que demostrar, y eso le complicará un poco la vida, pero... bueno, allá usted.

¿Qué hacer para que los proyectos de software salgan bien?: La antítesis

Bueno... tiene que hacer las cosas de esta manera: ponga a los programadores por parejas, desarrolle trozos pequeños en 2 ó 4 semanas cada uno. Haga una reunión de una jornada antes de empezar a programar cada trozo, y organícela en dos partes: en la primera cierre los requisitos con todo el equipo, en la segunda...
Todos los días el equipo debe reunirse durante 5 minutos y responder a tres preguntas...

Este es el tipo de respuesta de la antítesis: de los modelos ágiles, que en rigor no serían "modelos" sino "prácticas", porque no dicen QUÉ hay que hacer, sino CÓMO hay que hacer.

Y posiblemente la síntesis está en el PORQUÉ.: ¿Por qué la descripción del sistema o ConOps con el estándar IEEE 1362? o, ¿por qué en un Product Backlog?



Criterios para cuestionar los “por qués”

La talla única no resulta válida, y las prácticas más indicadas para unos, pueden resultar bien demasiado pesadas, bien insuficientes para otros. Los responsables de la gestión en cada empresa adoptan las decisiones para cada área de proceso sobre el conocimiento del grado y formas que necesita su empresa en tres áreas:

- Equilibrio agilidad – procesos
- Formas y modos de institucionalización
- Formas y modos de ingeniería de procesos.

Y este conocimiento es consecuencia del análisis del tipo de trabajo, y de las características de la organización.



Además de determinar un nivel de equilibrio entre agilidad y procesos, adecuado al tipo de trabajo que se realiza, hay otros dos factores importantes en el diseño y gestión del marco de trabajo de la empresa: la institucionalización y la ingeniería de procesos.

Los modelos para la mejora de procesos como CMMI tienen una fortaleza importante, que a la vez es debilidad de las prácticas ágiles: se trata de las prácticas genéricas, que están presentes en todas las áreas de procesos para que en todas se consiga:

- "Institucionalizar" las formas de trabajo
- Implantar prácticas de ingeniería de procesos.

Tal y como lo define el glosario de CMMI, ***institucionalizar*** las prácticas de trabajo es incrustarlas en la cultura corporativa de la empresa de manera que se realicen de forma rutinaria.

> ***La institucionalización se refiere a las ventajas de documentar los procedimientos que emplea la empresa, y disponerlos de forma accesible a todos los interesados; a la necesidad de formar al personal para que las conozca y sepa emplearlas en el trabajo.***

Claro que CMMI se refiere a la institucionalización de sus prácticas, pero lo cierto es que resulta útil tanto para los procesos como para las rutinas de trabajo ágiles.

Los criterios de rigor de documentación, comunicación, formación y ámbito dependen de las características de la organización y no de agilidad o procesos.

A un grupo de emprendedores, que desarrollan en una "start-up" un innovador producto de software, aplicar procesos pesados para la institucionalización, les aporta más problemas que ventajas.

Una gran empresa, o una pequeña que quiere asentar principios para dar el salto: de personas que saben programar a empresa que sabe programar, debe incluir procesos para explicitar e institucionalizar su saber hacer, independientemente de que sean procesos o agilidad.

Y, en segundo lugar, las prácticas de ingeniería de procesos, son las relacionadas con la medición y mejora de las propias prácticas y procesos empleados en el trabajo.

Ingeniería de procesos:

1.- Análisis del entorno y el trabajo

2. Criterios y diseño de procesos y/o rutinas de trabajo

3.- Medición y análisis

4.- Mejora

Igual que para la institucionalización, es irrelevante que se trate de agilidad o de procesos. Las características de la organización determinarán para cada área "por qué" resulta conveniente o no, emplear modelos tipo PDCA[5], IDEAL[6], las métricas, los criterios y las formas apropiadas.

La organización

Las características de la organización que se deben tener en cuenta son:

- Nº de personas
- Previsión / estrategia de evolución

Una empresa puede ser un reducido grupo de técnicos, o una gran organización con departamentos de programación en diferentes países; y las no muy grandes pueden tener estrategias de crecimiento más o menos ambiciosas; de forma que aunque pudieran resultar poco aconsejables prácticas de institucionalización por el tamaño, sí que pueden ser necesarias para permitir el crecimiento, o la homogeneidad del trabajo a pesar de la dispersión geográfica.

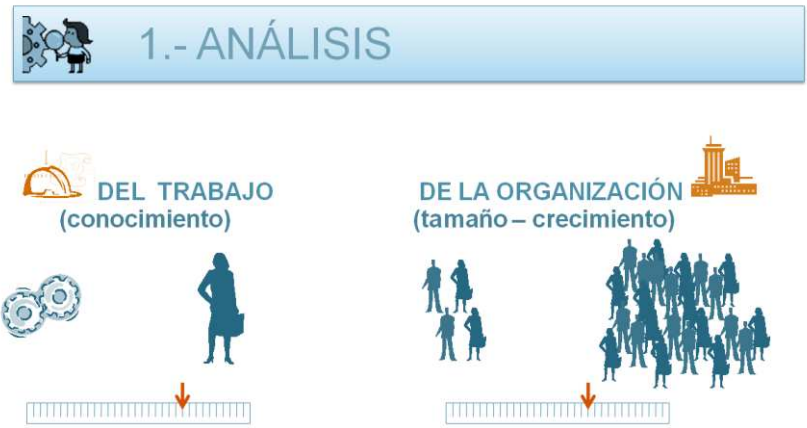
El esfuerzo de institucionalización tiene que ser coherente con estas circunstancias, porque lograr que siempre se trabaje de la misma forma puede no necesitar ninguna práctica adicional, en el primer caso, o requerir importantes recursos.

---

[5] Ciclo "Plan, Do, Check, Act (Planificar, Hacer, Verificar, Actuar), o círculo de Deming.

[6] Modelo IDEAL (Initiating, Diagnosing, Establishing, Acting & Learning) para la mejora organizacional, definido y utilizado por CMMI.

En la actualidad, por no tener en cuenta estas consideraciones, pequeñas organizaciones incorporan procesos que sólo les aportan incomodidades y burocracia, y algunas organizaciones ágiles no pueden abordar la escalabilidad y repetitividad de su "saber hacer".



El trabajo

También las características del trabajo son relevantes para cuestionar los "porqués"; que como apunta la introducción del libro, decir software es decir mucho y no decir nada. Hay empresas para sistemas muy diferentes. Unas se pueden dedicar, por ejemplo, a la configuración e integración de productos estándar (CRM's, ERP's…) y otras al desarrollo de artefactos novedosos para servicios vanguardistas.

Las primeras pueden "explicitar" la mayor parte del conocimiento necesario (integración y configuración de sistemas) en procesos y tecnología; y trabajar con personal de menor conocimiento "tácito", porque la calidad y homogeneidad de los resultados, depende más de los procesos que de las personas.

En las segundas ocurre lo contrario.

A las primeras aprovecha más funcionar en marcos de modelos de procesos.

A las segundas interesa acercarse a los principios de la agilidad.



Las singularidades de la organización y de sus proyectos dibujan el patrón personal de cada empresa; la "talla" de prácticas y modelos que mejor le sientan.

Y una vez conocido el patrón, se puede tomar como mapa de referencia para analizar qué procesos y tareas resultan convenientes o no, y en qué forma: el estándar ISO/IEC 12207, cuestionando en cada caso:

- Su conveniencia.
- Si debe orientarse como procesos, o como rutinas de trabajo.
- Conveniencia y forma de institucionalización.
- La conveniencia y forma de su inclusión en prácticas de ingeniería de procesos, y con qué nivel y rigor de medición: cualitativa o cuantitativa y sobre qué métricas.

Las líneas generales[7] de análisis para las principales áreas son:

ADQUSICIÓN – SUMINISTRO.

De adquisición son las tareas responsabilidad del cliente: cosas como saber qué necesita, de qué forma y con qué criterios va a seleccionar al proveedor, forma de contrato, seguimiento del desarrollo y validación de lo entregado.

---

[7] Con el objetivo de introducción y planteamiento global, a continuación describen criterios generales para los procesos primarios del estándar ISO 12207, a excepción del de operación.

Y de suministro son los procesos y tareas que realiza el proveedor para responder a las actividades de adquisición.

Tanto si se trabaja de forma ágil como no, en ambos casos la responsabilidad del proceso de adquisición es del cliente; y bien él directamente, o asistido por servicios o profesionales, debería conducirlo de forma consecuente, solicitando que el desarrollo se realice de forma predictiva o ágil según sean:

- El valor innovador que se espera del sistema.
- La incertidumbre del entorno de negocio del cliente.
- La inestabilidad prevista de los requisitos.

Estos criterios marcan si para describir el sistema es más apropiado completar una descripción tipo IEEE STD 1362 (IEEE, 1998) , antes de empezar el desarrollo, o a través de un Product Backlog en constante evolución durante la ejecución del proyecto.

También determinan si resulta mejor un modelo de contrato por obra, con descripción y plazos cerrados, o un modelo de contrato de servicio para el desarrollo iterativo del sistema.

Las prácticas de validación del sistema también pueden ser continuas durante el desarrollo sobre cada funcionalidad o cada iteración, o a la entrega, tomando como referencia los requisitos del proyecto.

DESARROLLO

Los procesos de desarrollo cubren la gestión, los requisitos del software, análisis, codificación, integración y pruebas.

Una vez conocidas y analizadas las características del proyecto resulta fácil decidir "porqué" es mejor gestionar el desarrollo con un ciclo en cascada o iterativo e incremental.

Por qué es mejor disponer de una descripción cerrada de requisitos del software, o trabajar con las historias de usuario o el *sprint backlog* de cada iteración.

Por qué es más apropiado integrar cada parte en las fechas y según prevé el Gantt del proyecto, o trabajar con integración y pruebas sistemáticas continuas.

Si por los requisitos de integridad del sistema, interesa emplear planes de verificación del tipo de un estándar IEEE 1012 con independencia clásica (equipos diferentes, presupuestos diferentes, gestión diferente), o basta con una independencia doméstica, del propio equipo e integrada en las pruebas.

MANTENIMIENTO

En los sistemas desarrollados con patrones predictivos el mantenimiento deberá adoptar modelos formales tipo IEEE Std. 1219. Cada petición de cambio tendrá que evaluarse, catalogarse, analizarse… de forma adecuada al tamaño y características del proyecto.

En los sistemas desarrollados de forma ágil, los procesos de mantenimiento pueden incluso no tener sentido, puesto que el sistema está en desarrollo continuo, y desde su beta inicial va incrementando y mejorando continuamente su funcionalidad, y reparando los problemas que se detectan.

Por la propia definición de mantenimiento como "modificación de un producto de software, después de su entrega…" en los ciclos de vida ágil para productos en continua evolución, y por tanto en continua modificación, puede no tener sentido aplicar estos procesos, ya que las prácticas ágiles contemplan el cambio continuo.

PROCESOS ORGANIZACIONALES

El diseño y la gestión de cada área de procesos, de las personas, de la calidad y la mejora continua, la cultura e institucionalización de las formas de trabajo; en definitiva: la gestión de la organización, debe estar alineada y ser coherente entre todos los procedimientos; y en conjunto coherente en el sistema lógico e inter-relacionado que es la organización.

Aquí, y posiblemente más que en ningún otra área, encajan mal las tallas únicas. No es realista emplear modelos o prácticas generales de gestión, y el saber hacer y la capacidad de los gestores es mucho más importante que el de los procesos, precisamente para determinar cuáles o qué prácticas deben emplearse.

ISO 12207 contempla estos procesos (organizacionales) en el ámbito del desarrollo de software, (gestión, formación infraestructura, mejora continua) pero con la perspectiva de organización como sistema, la propuesta de Scrum Management es gestionar de forma global las áreas

que tienen relaciones significativas, para evitar situaciones de incoherencia en la empresa.

Cada empresa puede estar distribuida en departamentos de: recursos humanos, desarrollo, comercial, calidad, diseño… según el tipo de sistemas de software que desarrolle o integre, el tamaño y distribución de la organización, etc.

Algunas diseñan y desarrollan su propio catálogo de productos, otras trabajan como factorías de software para otras, las hay que integran sistemas de terceros, que ofrecen servicios de outsourcing, etc.

En unos casos se trata de grandes sistemas para banca, por ejemplo, y en otros, páginas web para un pequeño comercio electrónico. En unos casos los trabajos de programación son prácticamente rutinarios, sin necesitar tan apenas innovación, como por ejemplo añadir nuevos listados a un sistema de gestión; y en otros casos se trata de desarrollar productos sorprendentes.

En esta, que es nuestra industria, ¿cuáles son los mejores criterios y prácticas?

¿Pueden estar desconectadas de los modelos de desarrollo de software?

- ¿La selección de personal?
- ¿La gestión de recursos humanos?
- ¿Los procesos comerciales?
- ¿Los procesos de calidad?
- ¿La formación (de todo el personal de la empresa)?
- ¿I+D+i?
- …

Los principios son simples: Análisis de las características de la empresa, del tipo de software, gestión sistémica, alineación y coherencia; y como las realidades pueden ser muchas, la mejor guía son las cualidades humanas de los gestores: criterio y conocimiento.

.

LA PRÁCTICA DE SCRUM

El modelo Scrum

El origen

Scrum es una metodología ágil para gestionar proyectos de software, que toma su nombre y principios de los estudios realizados sobre nuevas prácticas de producción por Hirotaka Takeuchi e Ikujijo Nonaka a mediados de los 80 (Ikujiro & Takeuchi, 1986).

Aunque surgió como práctica en el desarrollo de productos tecnológicos, resulta válido en los entornos que trabajan con requisitos inestables, y necesitan rapidez y flexibilidad; situaciones habituales en el desarrollo de algunos sistemas de software.

Scrum para software

En 1993, Jeff Sutherland aplicó el modelo Scrum al desarrollo de software en Easel Corporation (Empresa que en los macro-juegos de compras y fusiones se integraría en VMARK, luego en Informix y finalmente en Ascential Software Corporation). En 1996 presentó, junto con Ken Schwaber, las prácticas que empleaba como proceso formal, para gestión del desarrollo de software en OOPSLA 96 (Schwaber & Sutherland, 1996).

En 2001 formaron parte de los firmantes del Manifiesto Ágil. Las prácticas diseñadas por Schwaber y Sutherland para gestionar el desarrollo de software están incluidas en la lista de modelos ágiles de Agile Alliance.

Introducción al modelo

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

Scrum es una metodología ágil:

- Es un modo de desarrollo de carácter adaptable.
- Orientado a las personas antes que a los procesos.
- Emplea desarrollo ágil: iterativo e incremental.

El desarrollo se inicia desde la visión general de producto, dando detalle solo a las funcionalidades que, por ser las de mayor prioridad para el negocio, se van a desarrollar en primer lugar, y pueden llevarse a cabo en un periodo de tiempo breve (entre 15 y 60 días).

Cada uno de los ciclos de desarrollo es una iteración (sprint) que produce un incremento terminado y operativo del producto.

Estas iteraciones son la base del desarrollo ágil, y Scrum gestiona su evolución a través de reuniones breves de seguimiento en las que todo el equipo revisa el trabajo realizado desde la reunión anterior y el previsto hasta la reunión siguiente.

El protocolo de Scrum para Software definido por Jeff Sutherland y Ken Schwaber prescribe que las reuniones de seguimiento del sprint (iteración) sean diarias.

Control de la evolución del proyecto

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

Revisión de las Iteraciones

Al final de cada sprint o iteración, se realiza una revisión con todas las personas implicadas en el proyecto. Este es el periodo máximo que se puede tardar en reconducir una desviación del proyecto o de las circunstancias del producto

Desarrollo incremental

En el proyecto, no se trabaja con diseños o abstracciones.

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

Desarrollo evolutivo

Como modelo ágil, es útil en entornos con incertidumbre e inestabilidad de requisitos.

Intentar predecir en las fases iniciales cómo será el resultado final, y sobre dicha predicción desarrollar el diseño y la estructura del producto no es realista, porque las circunstancias obligarán a remodelarlo muchas veces.

¿Para qué predecir los estados finales de la estructura, arquitectura o diseño si van a estar cambiando? Scrum toma a la inestabilidad como premisa; por eso el protocolo de las prácticas de trabajo que se diseñen tiene que permitir la evolución continua sin degradar la calidad de la arquitectura, que se irá generando durante el desarrollo.

Con Scrum, el diseño y la estructura del resultado se construyen de forma evolutiva. No se considera que la descripción detallada del producto, del servicio, de la estrategia o de la arquitectura del software (según el caso) deban realizarse en la primera "fase" del proyecto.

El desarrollo ágil no es un desarrollo por fases.

En la aplicación de Scrum para software, para evitar los problemas de degradación del sistema o de la arquitectura por la evolución continua del producto se deben incluir prácticas de refactorización en las tareas de diseño y codificación.

Auto-organización

Durante el desarrollo de un proyecto surgen circunstancias impredecibles en todas las áreas y niveles. La gestión predictiva confía la responsabilidad de su resolución al gestor de proyectos.

En Scrum los equipos son auto-organizados, con margen de decisión suficiente para tomar las decisiones que consideren oportunas.

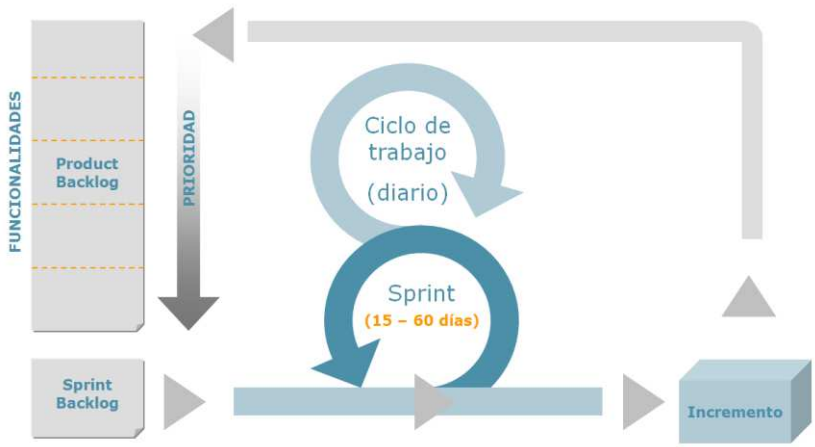
Colaboración

Las prácticas y el entorno de trabajo ágiles facilitan la colaboración del equipo, que es necesaria y debe basarse en la colaboración abierta entre todos según los conocimientos y capacidades de cada persona, y no según su rol o puesto.

Visión general del proceso

El resultado final se construye de forma iterativa e incremental.

Al comenzar cada iteración ("sprint") se determina qué partes se van a construir, tomando como criterios la prioridad para el negocio, y la cantidad de trabajo que se podrá abordar durante la iteración.



Los componentes y conceptos empleados en Scrum son:

Las reuniones

- **Planificación del sprint**: Jornada de trabajo previa al inicio de cada sprint en la que se determina cuál es el trabajo y los objetivos que se deben cubrir con esa iteración.
  Esta reunión genera la "sprint backlog" o lista de tareas que se van a realizar, y en ella también se determina el "objetivo del sprint": lema que define la finalidad de negocio que se va a lograr.

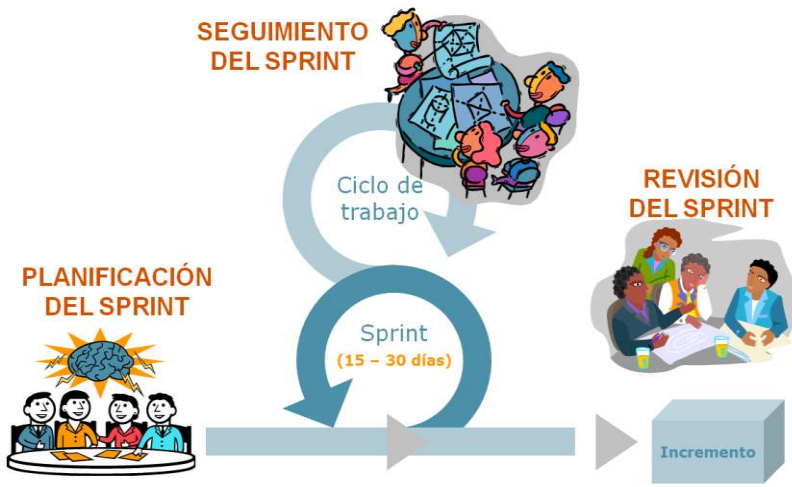
- **Seguimiento del sprint**: Breve reunión diaria para dar repaso al avance de cada tarea, y al trabajo previsto para la jornada.
  Sólo interviene el equipo, y cada miembro responde a tres preguntas:

  1.- Trabajo realizado desde la reunión anterior.

2.- Trabajo que se va a realizar hasta la próxima reunión de seguimiento.

3.- Impedimentos que se deben solventar para que pueda realizar el trabajo.

- **Revisión de sprint**: Análisis y revisión del incremento generado. Esta reunión no debe tomarse como un "acontecimiento especial", sino como la presentación normal de los resultados.



Los elementos

- **Product backlog**: Requisitos del sistema. Se parte de la visión del resultado que se desea obtener; y evoluciona durante el desarrollo.
  Es el inventario de características que el propietario del producto desea obtener, ordenado por orden de prioridad.

  Es un documento "vivo", en constante evolución.

  Es accesible a todas las personas que intervienen en el desarrollo.

  Todos pueden contribuir y aportar sugerencias.

  El responsable del product backlog es una única persona y se le denomina: propietario del producto.

- **Sprint Backlog**: Lista de los trabajos que realizará el equipo durante el sprint para generar el incremento previsto.
  El equipo asume el compromiso de la ejecución.

  Las tareas están asignadas a personas, y tienen estimados el tiempo y los recursos necesarios.

- **Incremento**: Resultado de cada sprint.
  Se trata de un resultado completamente terminado y en condiciones de ser usado.

Los roles o responsabilidades

El grado de funcionamiento de Scrum en la organización depende directamente de estas tres condiciones:

- Características del entorno (organización y proyecto) adecuadas para desarrollo ágil.
- Conocimiento de la metodología de trabajo en todas las personas de la organización y las implicadas del cliente.
- Asignación de responsabilidades:
  - Del producto.
  - Del desarrollo.
  - Del funcionamiento de Scrum

Responsabilidad del producto: El propietario del producto

En el proyecto hay una persona, y sólo una, conocedora del entorno de negocio del cliente y de la visión del producto. Representa a todos los interesados en el producto final y es el responsable del Product Backlog.

Se le suele denominar "propietario del producto" y es el responsable de obtener el resultado de mayor valor posible para los usuarios o clientes.

Es responsable de la financiación necesaria para el proyecto, de decidir cómo debe ser el resultado final, del lanzamiento y del retorno de la inversión.

En desarrollos internos puede ser el product manager, o responsable de marketing… quien asume este rol.

En desarrollos para clientes externos lo más aconsejable es que sea el responsable del proceso de adquisición del cliente.

Responsabilidad del desarrollo: El equipo

Todo el equipo de desarrollo, incluido el propietario del producto conoce la metodología Scrum, y son los auténticos responsables del resultado.

Es un equipo multidisciplinar que cubre todas las habilidades necesarias para generar el resultado.

Se auto-gestiona y auto-organiza, y dispone de atribuciones suficientes en la organización para tomar decisiones sobre cómo realizar su trabajo.

Responsabilidad del funcionamiento de Scrum (scrum manager)

La organización debe garantizar el funcionamiento de los procesos y metodologías que emplea, y en este aspecto Scrum no es una excepción.

En el modelo de Scrum definido por Jeff Sutherland, esta responsabilidad se garantiza integrando en el equipo una persona con el rol de ScrumMaster.

Considerando que las realidades de unas y otras empresas pueden ser muy diferentes, y que siempre que sea posible es mejor optar por adaptar las prácticas de trabajo a la empresa, y no al revés, en ocasiones puede resultar más aconsejable:

- Que en lugar de una persona con la función de "ScrumMaster", sean las personas y puestos más adecuados en cada organización los que reciban la formación adecuada y asuman las funciones correspondientes para cubrir esta responsabilidad.
- Que al compromiso de funcionamiento del proceso se sume también la dirección de la empresa, con el conocimiento de gestión y desarrollo ágil; y facilitando los recursos necesarios.

Scrum Manager designa por tanto, más que al rol, a la responsabilidad de funcionamiento del modelo. Puede ser a nivel de proyecto o a nivel de la organización; y en algunos casos resultará más apropiado un rol exclusivo (tipo ScrumMaster) y en otros, puede ser mejor que la responsabilidades de funcionamiento las asuman los responsables del departamento de calidad o procesos, o del área de gestión de proyectos…

Herramientas

Gráfico Burn-Up

Herramienta de gestión y seguimiento para el propietario del producto. Presenta de un vistazo las versiones de producto previstas, las funcionalidades de cada una, velocidad estimada, fechas probables para cada versión, margen de error previsto en las estimaciones, y avance real.

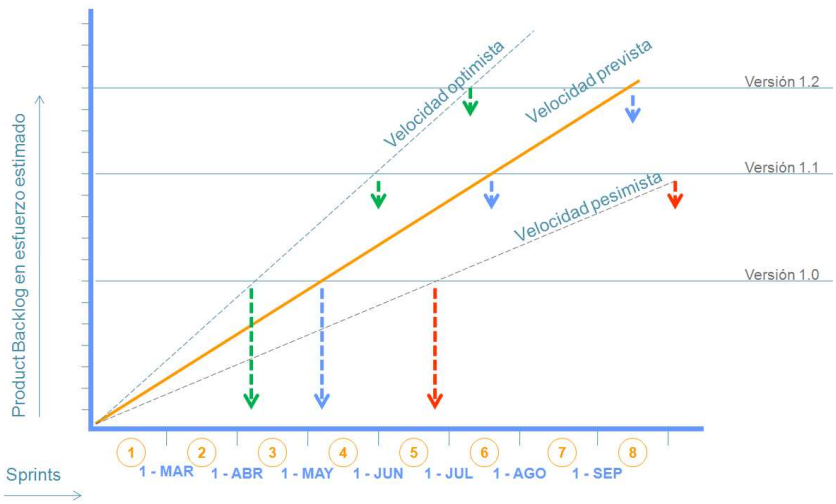


Gráfico Burn-Down

Herramienta del equipo para gestionar y seguir el trabajo de cada sprint.

Representación gráfica del avance del sprint.

Juegos y protocolos de decisión

Estimación de póker: Juego para agilizar y conducir la estimación de las tareas en la reunión de inicio del sprint.

Estimación a los chinos: Otro protocolo con formato de juego para realizar estimaciones en equipo.

Conceptos y métricas

Tiempo real o tiempo de trabajo.

Tiempo efectivo para realizar un trabajo. Se suele medir en horas o días.

Tiempo teórico o tiempo de tarea

Tiempo que sería necesario para realizar un trabajo en "condiciones ideales": si no se produjera ninguna interrupción, llamadas telefónicas, descansos, reuniones, etc.

Puntos de función o puntos de funcionalidad

Unidad de medida relativa para determinar la cantidad de trabajo necesaria para construir una funcionalidad o historia de usuario del product backlog.

Estimaciones

Cálculo del esfuerzo que se prevé necesario para desarrollar una funcionalidad.

Las estimaciones se pueden calcular en unidades relativas (puntos de función) o en unidades absolutas (tiempo teórico).

Velocidad absoluta

Cantidad de producto construido en un sprint. Se expresa en la misma unidad en la que se realizan las estimaciones (puntos de función, horas o días reales o teóricos).

Velocidad relativa

Cantidad de producto construido en una unidad de tiempo de trabajo.

P. ej.: puntos de función / semana de trabajo real; o horas teóricas / día de trabajo real…

Valores

Las prácticas de Scrum son una "carrocería" que permite trabajar con los principios ágiles, que son el motor del desarrollo. Son una ayuda para organizar a las personas y el flujo de trabajo; como lo pueden ser otras propuestas de formas de trabajo ágil: Cristal, DSDM, etc.

La carrocería sin motor, sin los valores que den sentido al desarrollo ágil, no funciona.

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

Scrum: Los elementos

Introducción

Los principales elementos de Scrum son:

- Product Backlog. Lista de las funcionalidades que necesita el cliente, priorizada según las prioridades que él determina.
- Sprint Backlog: Lista de tareas que se van a realizar en un sprint.
- Incremento: Parte del sistema desarrollada en un sprint.

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

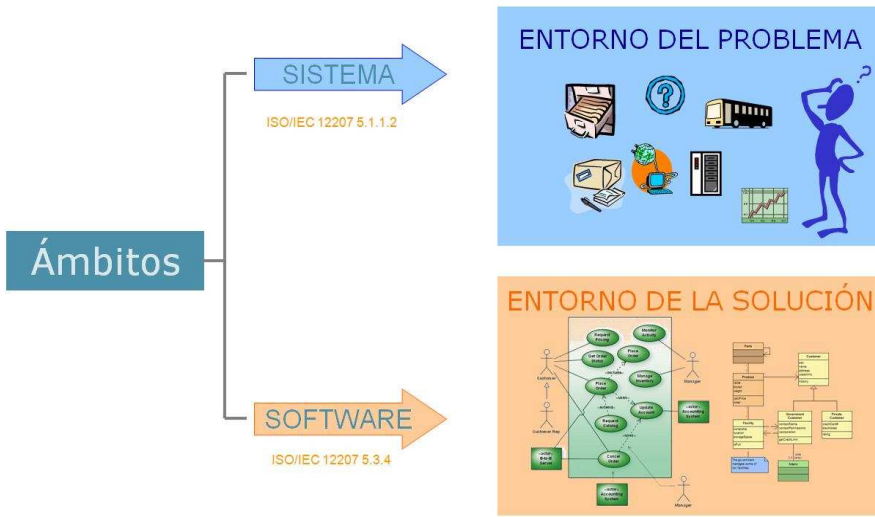
El incremento, como valor real para el cliente, no se trata de un prototipo, o de módulos o subrutinas a falta de pruebas o integración, sino de una parte del producto final, completamente operativa que podría entregarse tal cual al cliente.

Los requisitos en el desarrollo ágil

La ingeniería del software clásica diferencia dos áreas de requisitos

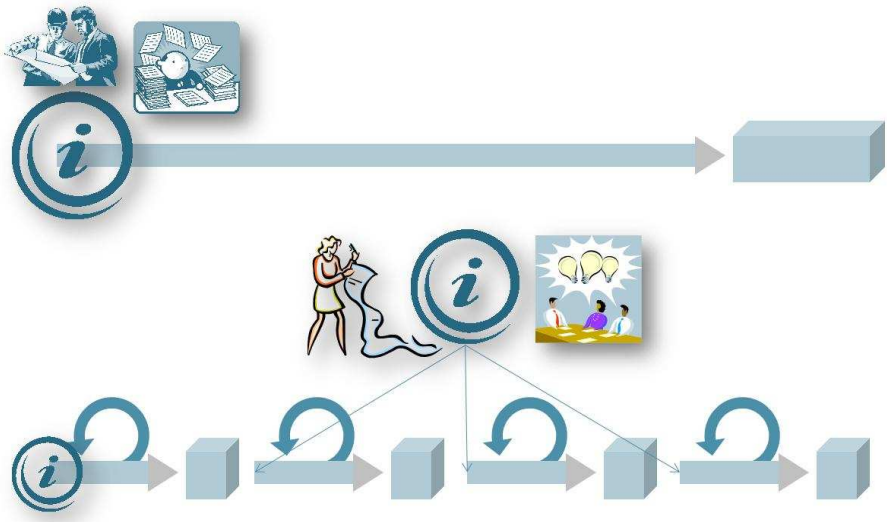
- Requisitos del sistema
- Requisitos del software

A los primeros los sitúa en el proceso de adquisición (ISO/IEC, 2004), haciendo por tanto al cliente responsable de definir cuál es su problema y qué funcionalidades tiene que aportar la solución.



No importa si es gestión tradicional o ágil. La descripción del sistema es responsabilidad del cliente, aunque la forma en la que la debe abordar es diferente en cada caso.

- En los proyectos predictivos, los requisitos del sistema suelen especificarse en documentos formales; y el product backlog de los proyectos ágiles toma la forma de lista de historias de usuario.
- Los requisitos del sistema formales se especifican por completo al inicio del proyecto, y el product backlog es un documento vivo, que evoluciona durante todo el desarrollo de forma concurrente con el resto de actividades.
- Los requisitos del sistema los desarrolla una persona o equipo especializado en ingeniería de requisitos a través del proceso de obtención (elicitación) con el cliente. En Scrum la visión del cliente es conocida por todo el equipo (el cliente forma parte del equipo") y el product backlog se realiza y evoluciona de forma continua con las aportaciones de todo el equipo.



Pero la responsabilidad es del cliente, del "propietario del producto" en el caso de Scrum ", que tiene la responsabilidad de decidir lo que se incluye en el product backlog, y el orden de prioridad.

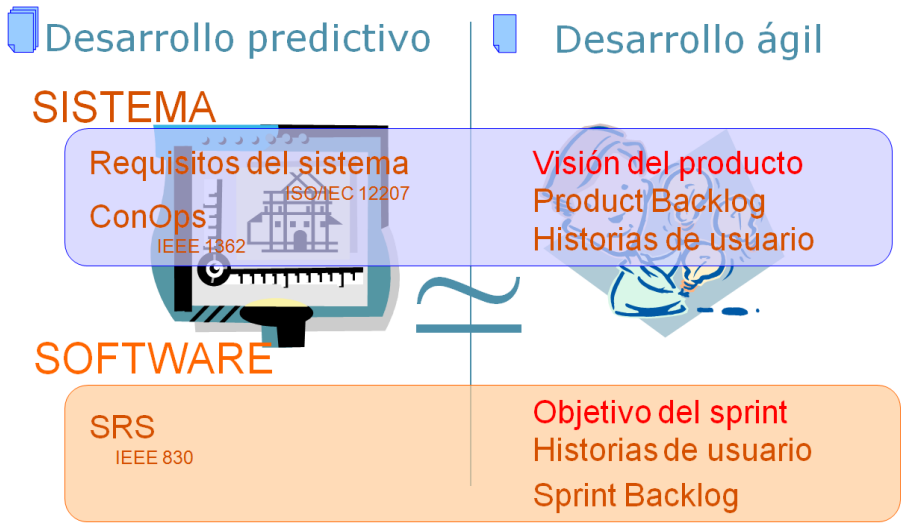
Requisitos y visión del producto

Scrum para software emplea dos formatos para el registro y comunicación de los requisitos:

- Product Backlog
- Sprint Backlog

El product Backlog se sitúa en el área de requisitos o necesidades de negocio desde el punto de vista del cliente. Área que en la ingeniería del software tradicional cubren los requisitos del sistema o ConOps (Concept of Operations).

El Sprint Backlog se sitúa en el área de especificación de los requisitos de software necesarios para dar respuesta a las funcionalidades esperadas por el cliente.



Para Scrum Management, los requisitos y sus modelos de especificación: Product Backlog o Sprint Backlog, se sitúan en la zona de la "forma", y para que la implantación de Scrum alcance niveles de capacidad elevados tiene que responder a una visión clara, conocida y compartida por todo el equipo, tanto a nivel de producto en general (visión del producto) como del sprint en el que se está trabajando (objetivo del sprint).

Product Backlog: los requisitos del cliente

El product backlog es el inventario de funcionalidades, mejoras, tecnología y corrección de errores que deben incorporarse al producto a través de las sucesivas iteraciones de desarrollo.

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

Estos son algunos ejemplos de posibles entradas de un backlog:

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

A diferencia de un documento de requisitos del sistema, el product backlog nunca se da por completo; está en continuo crecimiento y evolución.

Habitualmente se comienza a elaborar con el resultado de una reunión de "fertilización cruzada" o brainstorming; o un proceso de "Exploración" (Extreme Programming, pág. 87) donde colabora todo el equipo partiendo de la visión del propietario del producto.

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

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



El product backlog evolucionará de forma continua mientras el producto esté en el mercado, para darle valor de forma continua del mayor, y mantenerlo útil y competitivo.

Para comenzar el desarrollo se necesita una visión de los objetivos que se quieren conseguir con el producto, comprendida y conocida por todo el equipo, y elementos suficientes en el product backlog para llevar a cabo el primer sprint.

Formato del product backlog

El desarrollo ágil prefiere la comunicación directa, antes que a través de documentos. El product backlog no es un documento de requisitos, sino una herramienta de referencia para el equipo.

Es recomendable el formato de lista que incluya al menos la siguiente información para cada línea:

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

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

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

Es aconsejable no tomar ningún protocolo de trabajo de forma rígida. El formato del product backlog no es cerrado.
Los resultados de Scrum no dependen de la rigidez en la aplicación del protocolo, sino de la institucionalización de sus principios y la implementación en un "formato" adecuado a las características de la empresa y del proyecto.

Id	Orden	Est.	Descripción	Criterio validación	Obs.
1	10	30	Plataforma tecnológica	Se tiene el diagrama de la arquitectura, validado por xxx	La arquitectura debe permitir escalabilidad por clusterización de
2	20	40	Prototipos interfaz usuario	Todas las pantallas de interfaz están dibujadas y se puede recorrer toda la func	Debe estar interfaz para las funcionalidades de la pila a fecha
3	30	40	Diseño de datos	Diagrama BB.DD. Realizado, validado por xxx	
4	40	60	El operador define el flujo y textos de un expediente	Definir completamente un expediente con la funcionalidad programada	
5	50	999.	Etc...	Etc...	

Sprint backlog

El sprint backlog es la lista que descompone las funcionalidades del product backlog en las tareas necesarias para construir un incremento: una parte completa y operativa del producto.

En el sprint backlog se asigna a cada tarea la persona que la va a llevar a cabo, y se indica el tiempo de trabajo que se estima, aún falta para terminarla.

Es útil porque descompone el proyecto en tareas de tamaño adecuado para determinar el avance a diario; e identificar riesgos y problemas sin necesidad de procesos complejos de gestión.

Es también una herramienta de soporte para la comunicación directa del equipo.

Condiciones

- Realizado de forma conjunta por todos los miembros del equipo.
- Cubre todas las tareas identificadas por el equipo para conseguir el objetivo del sprint.
- Sólo el equipo lo puede modificar durante el sprint.
- El tamaño de cada tarea está en un rango de 4 a 16 horas de trabajo.
- Es visible para todo el equipo. Idealmente en una pizarra o pared en el mismo espacio físico donde trabaja el equipo.

Formato y soporte

Hay tres opciones:

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

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

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

Ejemplos

SPRINT	INICIO	DURACIÓN
1	1-feb-06	12

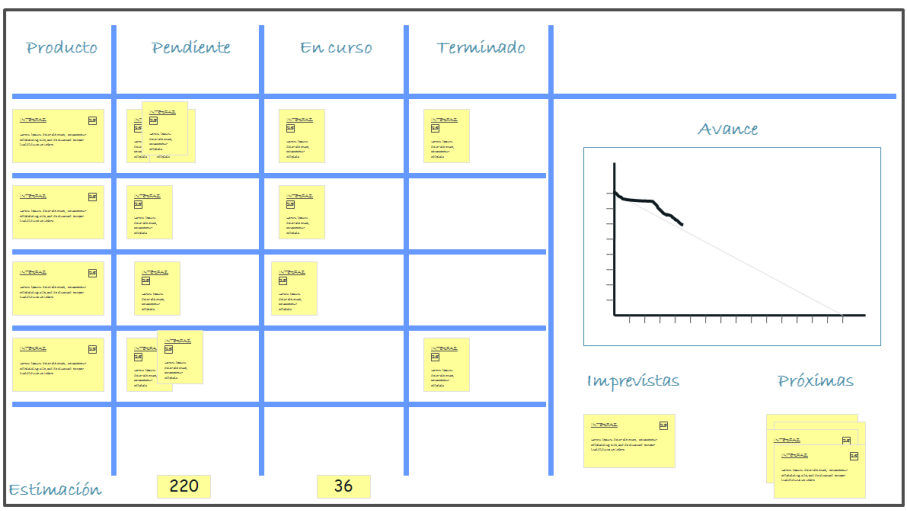
	X
	1-feb
Tareas pendientes	23
Horas de trabajo pendientes	276

PILA DEL SPRINT					
Backlog	Tarea	Tipo	Estad	Responsal	
1	Descripción de la tarea 1	Análisis	Terminada	Luis	16
1	Descripción de la tarea 2	Prototipado	Terminada	Luis	12
1	Descripción de la tarea 3	Pruebas	Terminada	Luis	4
1	Descripción de la tarea 4	Codificación	Terminada	Elena	8
1	Descripción de la tarea 5	Codificación	Terminada	Elena	16
2	Descripción de la tarea 6	Pruebas	Terminada	Elena	6
2	Descripción de la tarea 7	Codificación	Terminada	Antonio	16
2	Descripción de la tarea 8	Codificación	Terminada	Antonio	16
3	Descripción de la tarea 9	Pruebas	Terminada	Antonio	12
3	Descripción de la tarea 10	Pruebas	En curso	Luis	12
3	Descripción de la tarea 11	Codificación	Pendiente	Luis	8

SPRINT	INICIO	DURACIÓN
1	1-mar-07	12

	J
	1-mar
	23
	276

SPRINT BACKLOG			
Tarea	Estado	Responsable	
Descripción de la tarea 1	Terminada	Luis	16
Descripción de la tarea 2	Terminada	Luis	12
Descripción de la tarea 3	Terminada	Luis	4
Descripción de la tarea 4	Terminada	Elena	8
Descripción de la tarea 5	Terminada	Elena	16
Descripción de la tarea 6	Terminada	Elena	6
Descripción de la tarea 7	Terminada	Antonio	16
Descripción de la tarea 8	Terminada	Antonio	16
Descripción de la tarea 9	Terminada	Antonio	12
Descripción de la tarea 10	En curso	Luis	12
Descripción de la tarea 11	Pendiente	Luis	8



El Incremento

Incremento es la parte de producto desarrollada en un sprint.

El incremento es la parte de producto producida en un sprint, y tiene como características que está completamente terminada y operativa: en condiciones de ser entregada al cliente final.

No se trata por tanto de módulos o partes a falta de pruebas, o documentación o…

Idealmente en el desarrollo ágil:

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

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

Teniendo en cuenta esta excepción habitual, Incremento es:

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

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

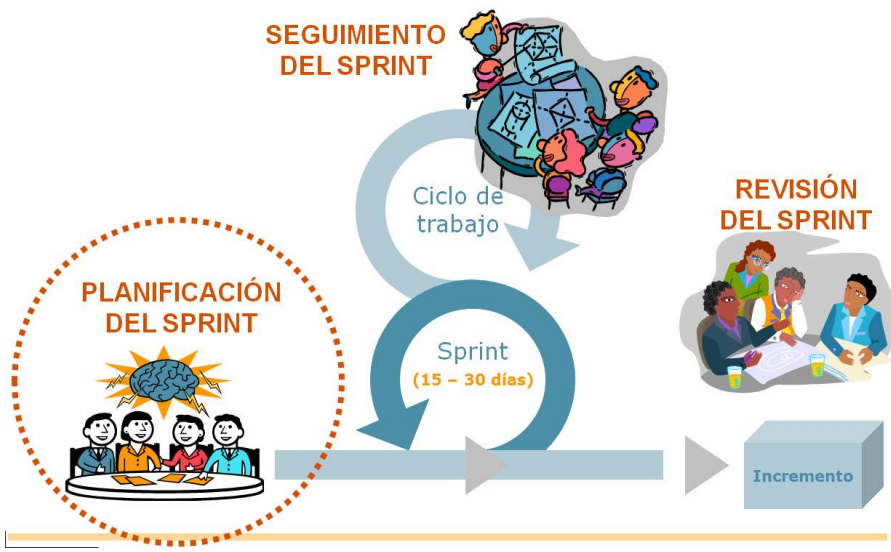
Las reuniones

Introducción

En el trabajo con Scrum, el seguimiento y la gestión del proyecto se basa en la información de trabajo de las tres reuniones que forman parte del modelo:

- Planificación del sprint
- Monitorización del sprint
- Revisión del sprint

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



Planificación del Sprint

Descripción general

En esta reunión, tomando como base las prioridades y necesidades de negocio del cliente, se determinan cuáles y cómo van a ser las

funcionalidades que se van a incorporar al producto con el próximo sprint.

En realidad esta reunión consiste en dos: En la primera, que puede tener una duración de una a cuatro horas, se decide qué elementos del product backlog se van a desarrollar.

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

La planificación del sprint no debe durar más de un día.

Las características de la reunión son:

Pre-condiciones:

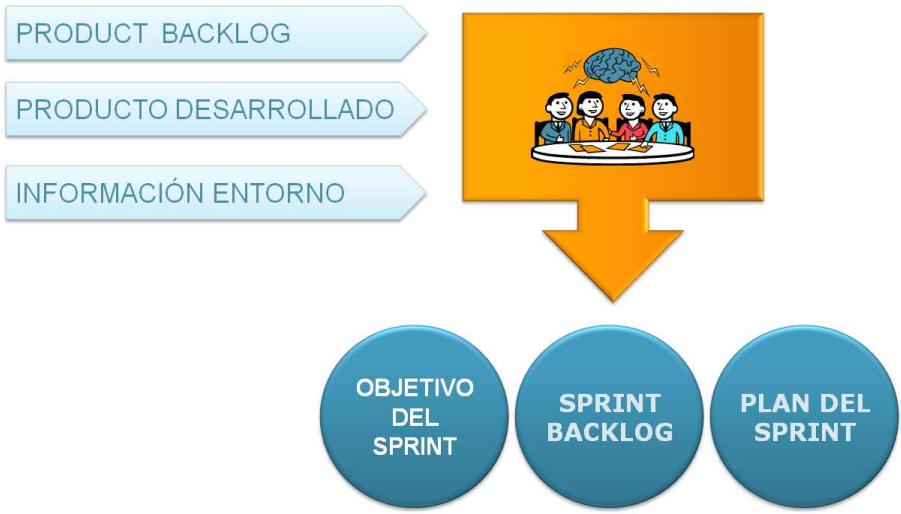
- La organización tiene determinados los recursos posibles para llevar a cabo el sprint.
- El propietario del producto tiene preparado el backlog del producto: con su criterio de prioridad para el negocio, y un nº suficiente de elementos para desarrollar en el sprint.
- Siempre que sea posible el propietario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

Entradas:

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

Resultados:

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



Es una reunión conducida por el responsable del funcionamiento de Scrum (v. pág. 132), a la que deben asistir el propietario del producto y el equipo al completo, y a la que también pueden asistir otros implicados en el proyecto.

La reunión comienza con la presentación del propietario del producto del backlog, en la que expone los resultados que por orden de prioridad necesita; especialmente los que prevé que se podrán desarrollar en el siguiente sprint.

Si el product backlog ha tenido cambios significativos desde la anterior reunión; explica también las causas que los han ocasionado.

El objetivo es que todo el equipo conozca las razones y los detalles con el nivel necesario para poder estimar el trabajo necesario.

Formato de la reunión

Esta reunión marca el inicio de cada sprint. Una persona con la responsabilidad de procesos en la organización (Scrum Manager) es el responsable de su organización y gestión.

Duración máxima: un día.

Deben asistir: el propietario del producto, el equipo y el Scrum Manager.

Pueden asistir: es una reunión abierta a todos los que puedan aportar información útil.

Consta de dos partes separadas por una pausa de café o comida, según la duración.

Primera parte:

Duración de 1 a 4 horas.

Propietario del producto:

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

El equipo

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

Las aportaciones del equipo pueden suponer modificaciones en el backlog. De hecho no es que "puedan" es que "deben" suponerlas.

Esta reunión es un punto caliente del protocolo de Scrum para favorecer la fertilización cruzada de ideas en equipo y añadir valor a la visión del producto.

Tras reordenar y replantear las funcionalidades de la pila del producto, el equipo define el "objetivo del sprint" o fase que define de forma sintética cuál es el valor que se le aportará al producto.

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

Segunda parte:

En la segunda parte, que puede alargarse hasta el final de la jornada:

El equipo desglosa cada funcionalidad en tareas, y estima el tiempo para cada una de ellas, determinando de esta forma los elementos del sprint backlog.

En este desglose el equipo tiene en cuenta los elementos de diseño y arquitectura que deberá incorporar el sistema.

Los miembros del equipo se auto-asignan las diferentes tareas teniendo como criterios sus conocimientos, intereses y distribución homogénea del trabajo.

Esta segunda parte debe considerarse como una "reunión del equipo", en la que deben estar todos sus miembros y ser ellos quienes descomponen el trabajo en tareas, las asignan y estiman.

El papel del propietario del producto en esta parte es atender a dudas y comprobar que el equipo comprende y comparte su objetivo.

El Scrum Manager actúa de conductor o moderador de la reunión.



Funciones del rol de Scrum Manager

El Scrum Manager es responsable y garante de:

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

2.-Que antes de la reunión que el propietario del producto disponga de un backlog adecuado y suficiente para realizar el sprint.

3.- Que el diálogo principal de la reunión se realice entre el propietario del producto y el equipo. Otros asistentes pueden participar, pero su colaboración no puede implicar toma de decisiones ni limitar el diálogo principal.

4.- Que la reunión sea un trabajo de colaboración activa entre los dos protagonistas: cliente y equipo, y concluyen con un acuerdo sobre el incremento de producto que van a realizar en el sprint.

5.- Que el equipo comprende la visión y necesidades de negocio del cliente.

6.- Que el equipo ha realizado una descomposición y estimación del trabajo realistas y ha considerado las posibles tareas necesarias de análisis, investigación o apoyo.

7.- Que al final de la reunión están objetivamente determinados:

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

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

Pizarra de trabajo

Es recomendable, que el propietario del producto emplee una hoja de cálculo, alguna herramienta similar, o el soporte de una intranet, para guardar en formato digital la pila del producto.

Pero no es aconsejable emplearla como base para trabajar sobre ella en la reunión proyectándola sobre la pantalla de la sala.

Es mucho mejor trabajar y manipular elementos físicos; y usar una pizarra y fichas removibles (adhesivas, con chinchetas o magnéticas).



La comunicación es más rica y fluida, y el grado de implicación que se logra mucho mayor.

Además un interfaz físico resulta más cómodo y visual. Para cambiar la prioridad de las tareas basta con moverlas de sitio.

Es posible tener sobre la mesa varias tareas simultáneamente, etc.

Un ejemplo de pizarra.

La pizarra es una herramienta para facilitar la comunicación y el trabajo de la reunión.

Al final de la reunión el propietario del producto registrará en la hoja de cálculo, o en la herramienta que emplee, el estado y las modificaciones en la pila del producto.

El equipo hará lo mismo con backlog del sprint.

Según la distribución y espacio de la oficina de trabajo quizá se reutilice la pizarra o las notas para el seguimiento del sprint; o quizá no.

Este es un ejemplo, pero la pizarra, y el resto de las formas, son técnicas que ayudan a trabajar de forma ágil; no reglas estrictas.

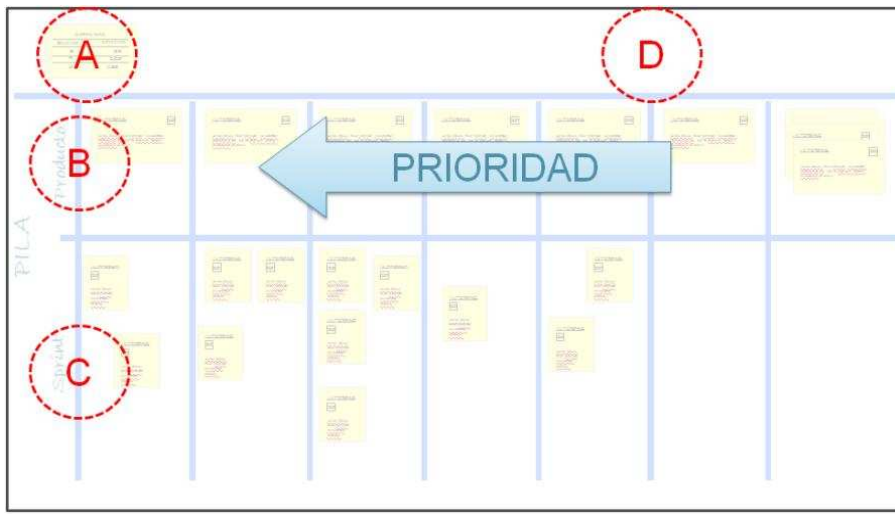
En cada caso se pueden ajustar o modificar según las características de la organización.

Algunos soportes que se suelen emplear:

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

Se puede conseguir una solución práctica y económica empleando fichas adhesivas ("Post-it") y usando como pizarra cartón pluma blanco de 5mm. fijado con puntas directamente sobre la pared.

El cartón pluma es un material ligero, de acabado satinado que puede adquirirse en tiendas de materiales para bellas artes y manualidades.

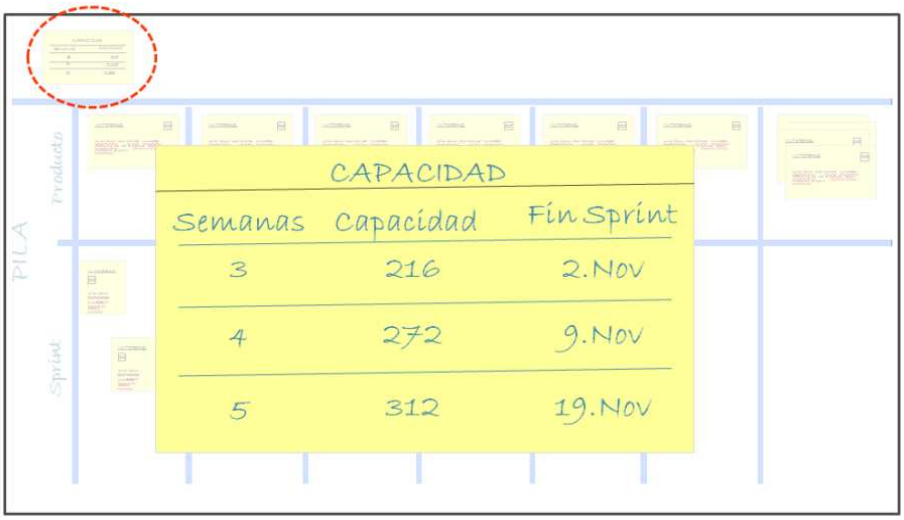


Con cinta adhesiva removible se marcan se delimita:

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

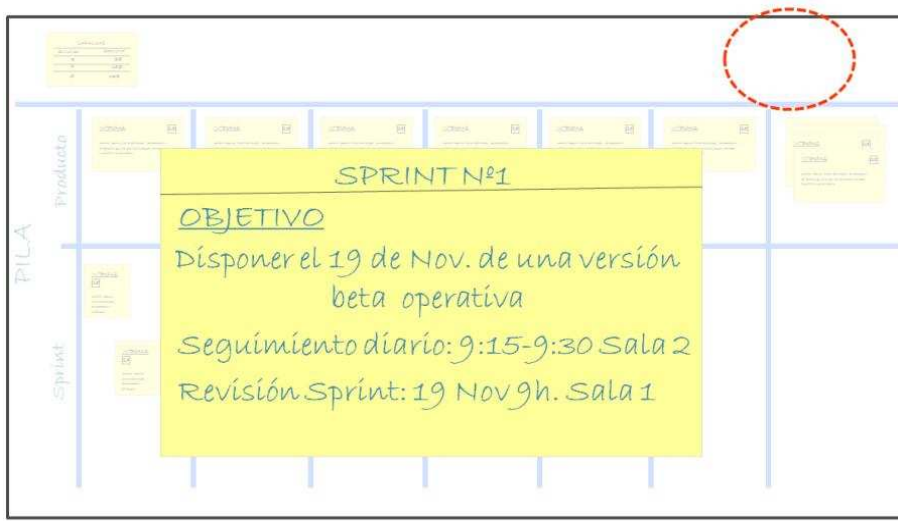
En cada ficha se refleja la información básica para las decisiones de la reunión: priorización, estimación, descomposición y asignación a los miembros del equipo.

Las siguientes imágenes muestran un ejemplo de uso:









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

Monitorización del Sprint

Descripción

Reunión diaria breve, de no más de 15 minutos en la que todos los miembros del equipo dicen las tareas en las que están trabajando, si se han encontrado o prevén encontrarse con algún impedimento y actualizan sobre el sprint backlog las tareas ya terminadas o los tiempos de trabajo que les quedan.

Pre-condiciones

- Disponibilidad de un lugar físico en la organización para realizar diariamente la reunión.
- Sprint backlog actualizado en el soporte que emplee el equipo (dibujado en pizarra, con post-it's, sobre hoja de cálculo…)
- Asiste todo el equipo
- Asiste un responsable con rol de Scrum Manager de la organización.
- Un miembro del equipo (team leader)conduce y garantiza el protocolo, formato y tiempos de la reunión.

Entradas

Sprint Backlog y gráfico Burn-down (página. 164) actualizados con la información de la reunión anterior.

Información de las tareas realizadas por cada componente del equipo

Resultados

Backlog y gráfico de avance (Burn-down) actualizados.

Identificación de necesidades e impedimentos.

Formato de la reunión

Se recomienda realizarla de pie y emplear un formato de backlog o lista de tareas en una pizarra o en la pared, para que todo el equipo pueda verlo, anotar o mover las tareas, junto con el gráfico de avance del sprint.

En la reunión está presente todo el equipo, y pueden asistir también otras personas relacionadas con el proyecto o la organización, pero éstas últimas no pueden intervenir.

Uno por uno, los miembros del equipo exponen estas tres cuestiones:

1.- Tarea en la que trabajaron ayer.

2.- Tarea o tareas en las que trabajarán hoy.

3.- Si van a necesitar alguna cosa especial o prevén algún impedimento para realizar su trabajo.

Y actualizan sobre el sprint backlog el tiempo de trabajo que queda pendiente a las tareas en las que están trabajando, o marcan las que hayan podido completar.

Al final de la reunión:

- Con las estimaciones de tiempos actualizadas por el equipo, el team leader actualiza el gráfico de avance del sprint.
- El responsable de la gestión de procesos de la organización (Scrum Manager) comienza a gestionar las posibles necesidades e impedimentos identificados.

Revisión del Sprint

Descripción

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

Objetivos:

- El propietario del producto obtiene una revisión del progreso del sistema. Esta reunión le ofrece a intervalos regulares el ritmo de construcción del sistema y la trayectoria que va tomando la visión del producto.
- Al ver el incremento funcionando, el propietario del producto, y el equipo en general obtienen feedback clave para evolucionar y dar valor al product backlog.

- Otros ingenieros y programadores de la empresa también pueden asistir para ver cómo trabaja la tecnología empleada.
- El responsable de procesos o calidad de la organización (Scrum manager) obtiene feedback sobre buenas prácticas y problemas durante el sprint, necesaria para las prácticas que se empleen de ingeniería de procesos y mejora continua (v. pág. 116).

Reunión que se realiza al final de cada sprint en la que el equipo muestra el incremento construido, y se genera retro-información entre todos los participantes para preparar el product backlog para el inicio del siguiente sprint.

Pre-condiciones

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

Entradas

- Incremento terminado.

Resultados

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

Formato de la reunión

Es una reunión informal. El objetivo es ver el incremento, trabajar en el entorno del cliente. Están prohibidas las presentaciones gráficas y "powerpoints".

El equipo no debe invertir más de una hora en preparar la reunión, y lo que se muestra es el resultado final: terminado, probado y operando en el entorno del cliente (incremento)

Según las características del proyecto puede incluir también documentación de usuario, o técnica.

Es una reunión informativa. NO TIENE UNA MISIÓN ORIENTADA A TOMAR DECISIONES, NI A CRITICAR EL INCREMENTO. Con la información generada en la preparación del siguiente sprint se expondrán y tratarán las posibles modificaciones sobre la visión del producto.

Un protocolo recomendado:

1.- El team leader expone el objetivo del sprint, la lista de funcionalidades que se incluían y las que se han desarrollado.

2.- El equipo hace una introducción general del sprint y demuestra el funcionamiento de las partes construidas.

3.- Se abre un turno de preguntas y sugerencias sobre lo visto. Esta parte genera información muy valiosa para que el propietario del producto, y para el equipo en general puedan mejorar el valor de la visión del producto.

4.- El responsable del proceso (Scrum Manager), de acuerdo con las agendas del propietario del producto y el equipo cierra la fecha para la reunión de preparación del siguiente sprint.

Las herramientas

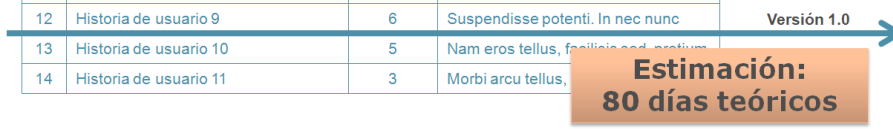
Gráfico Burn-Up

Es una herramienta de planificación y seguimiento del propietario del producto, que muestra en un gráfico muy simple el plan general de desarrollo del producto, y la traza de su evolución.

Se confecciona con:

- La estimación de esfuerzo prevista en el product backlog. (v. formato del product backlog pág. 142)
- La velocidad del equipo (v. velocidad absoluta pág. 134).

Id	Descripción	Esfuerzo	Criterio de validación
1	Historia de usuario 1	6	Lorem ipsum dolor sit amet
2	Historia de usuario 2	8	consectetuer adipiscing elit
3	Historia de usuario 3	3	Aliquam vehicula accumsan tortor
4	Historia de usuario 4	6	Pellentesque turpis
5	Historia de usuario 5	12	Phasellus purus orci
6	Historia de usuario 6	5	penatibus et magnis dis parturient
7	Historia de usuario 7	7	Quisque volutpat ante sit amet velit
8	Historia de usuario 8	8	Cras iaculis pede eu tellus
9	Historia de usuario 9	10	Vestibulum vel diam sed pede blandit
10	Historia de usuario 10	5	Suspendisse aliquam felis et turpis
11	Historia de usuario 11	4	Nullam imperdiet lorem vitae justo
12	Historia de usuario 9	6	Suspendisse potenti. In nec nunc
13	Historia de usuario 10	5	Nam eros tellus,
14	Historia de usuario 11	3	Morbi arcu tellus,

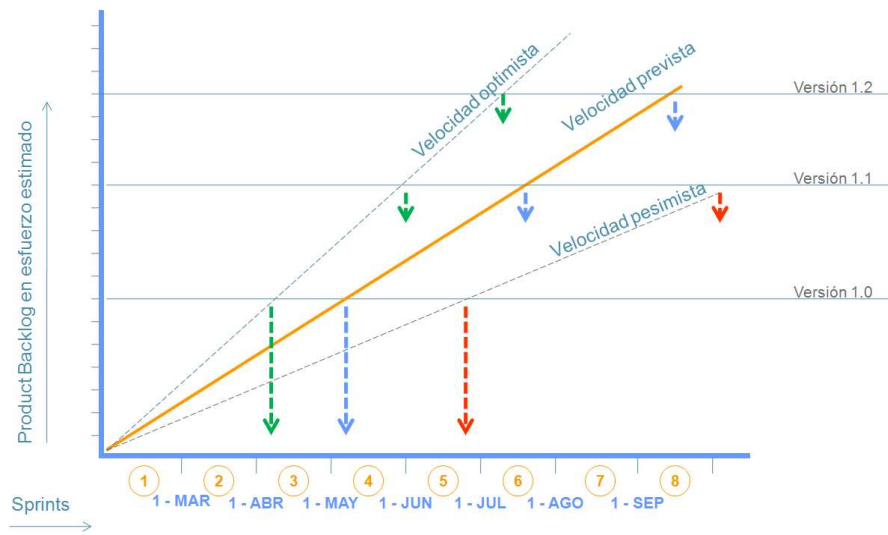


Estimación:
80 días teóricos

1.- Estimación del esfuerzo en el product backlog



El eje Y representa el esfuerzo, y sobre él se marcan los hitos de versiones previstas en el backlog.



El eje X representa el tiempo de desarrollo con las fechas de los sprints previstos.

En el área del gráfico se proyecta la línea que representa la velocidad de desarrollo del equipo.

Este dato se obtiene sobre el histórico de velocidad desarrollada por el mismo equipo en proyectos o sprints anteriores.

Si no se tiene información histórica, un buen dato para comenzar es utilizar "tiempo real" como unidad para el esfuerzo y la velocidad (horas o días reales) y suponer como velocidad del equipo un tercio del tiempo disponible de trabajo (v. pág. 134).

Ejemplo: Par un equipo de 3 personas y sprints de 20 días laborables, el tiempo disponible es de: 3 * 20 = 60 días disponibles.

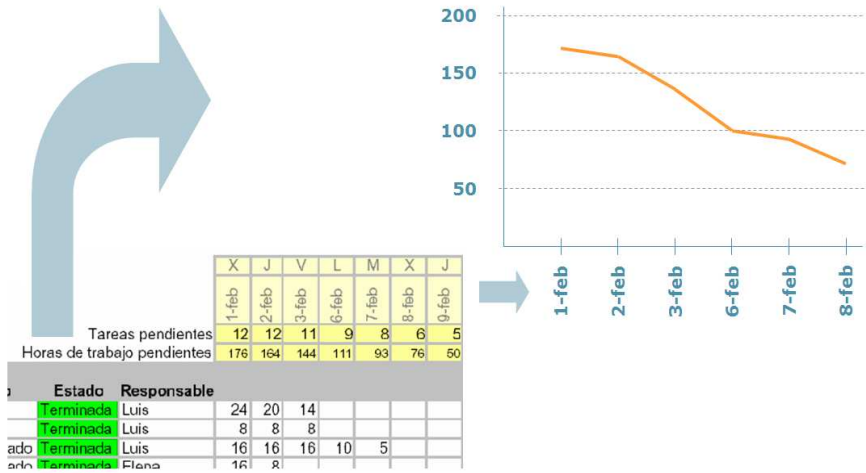
Velocidad previsible: 20 (60/3).

La intersección de los hitos en Y del esfuerzo previsto para una versión, con la línea de velocidad prevista, proyecta sobre X la fecha en la que previsiblemente estarán desarrolla la versión.

Si las estimaciones se realizan considerando valores optimistas y pesimistas de velocidad, o de esfuerzo necesario, se pueden obtener valores de rango de fechas de probabilidad.

Gráfico Burn-Down

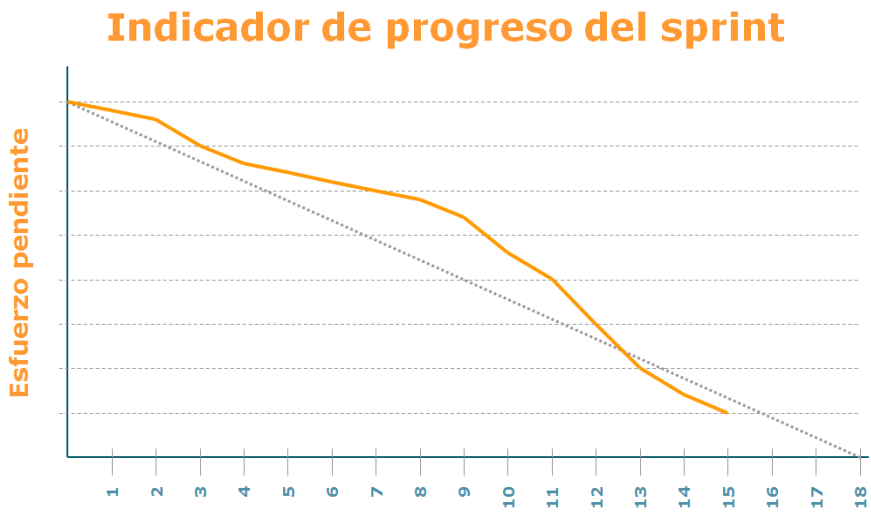
Herramienta de seguimiento para el equipo, que muestra el avance del sprint día a día y revela de forma temprana posibles desviaciones.



Es un gráfico cartesiano que representa en el eje x los días laborables del sprint, y en el y la cantidad de esfuerzo estimada.

En la reunión diaria cada miembro del equipo al referirse al trabajo que realizó ayer, y el que tiene previsto hacer hoy, actualiza en el sprint backlog si ha terminado alguna de las tareas en las que ha trabajado, o cuánto esfuerzo estima que les quedan.

De esta forma al final de la reunión la columna del día del sprint backlog muestra el esfuerzo que según el equipo falta para terminar el sprint, y el equipo marca en el gráfico el punto que tiene como ordenada ese valor, y como abscisa la fecha del día.



La evolución ideal del sprint se representaría por la línea punteada en gris de la figura.

La línea naranja muestra la evolución real diaria.

El recorrido sobre la diagonal es síntoma de problemas o sub-estimación del sprint backlog.

El recorrido bajo la diagonal es síntoma de sobre-estimación del backlog.

Juegos y protocolos de decisión

Estimación de póquer

Es un protocolo de trabajo ágil, que resulta útil en las reuniones en las que el equipo debe estimar el esfuerzo de las tareas con criterio de "juicio de expertos".

En estos casos es frecuente entrar en dilatadas exposiciones que no llegan a conclusiones concretas.

Para evitarlas, y ayudar a conducir la reunión, James Grening ideó este juego de planificación en el que cada participante dispone de 8 cartas con los números necesarios para representar la cifra de días que estima para la tarea.



Se ideó para las estimaciones de versión en Extreme Programming, modelo ágil que suele emplear como unidad de esfuerzo: días de trabajo de cada par de programadores; y tareas de una dimensión mínima de medio día, y máxima de 10.

Las tareas de mayor dimensión se dividen en otras menores.

Por esta razón el modelo original emplea las ocho cartas de la figura: con los siete números se puede representar cualquier estimación entre 10 días y medio día; y el infinito se emplea para indicar que la tarea necesitaría más de 10 días de trabajo, y por tanto debe dividirse en otras menores.

El número de cartas que debe tener cada participante, y los números que representan dependerán de la unidad de estimación empleada por el equipo: puntos de funcionalidad (story points), días u horas teóricas o reales (v. métricas, pág. 134).

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

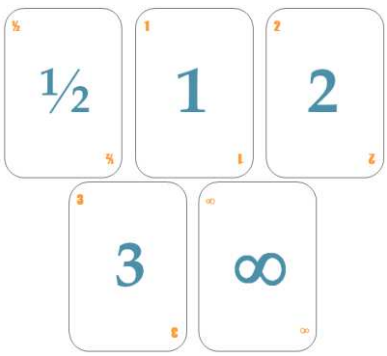
- No definir tareas demasiado grandes, porque dificulta la precisión de las estimaciones y la identificación de riesgos. Un criterio válido, si el equipo no dispone de experiencia previa, es descomponer en otras menores las que tengan una duración mayor de 7 días ideales de trabajo.
- No definir tareas con duraciones inferiores a medio día ideal de trabajo.
- Utilizar al misma unidad de medida (story points, días, horas…) en todos los gráficos y backlogs.

Ejemplo de uso en la reunión de planificación del sprint

Criterios empleados por el equipo:

- Unidad de estimación: días teóricos de trabajo.
- Tamaño máximo de tarea: 5 días teóricos de trabajo.
- Tamaño mínimo de tarea: ½ día teórico de trabajo.

Modelo de cartas empleado:



- Para cada funcionalidad del product backlog, una vez expuesta y comprendida, cada participante selecciona, sin mostrarlas, las cartas que suman el nº de días de trabajo teóricos que estima necesarios.
- Si calcula más de 5, selecciona la carta “infinito”.
- Cuando todos tienen preparada la estimación, se ponen boca arriba sobre la mesa.
- Si no hay gran diferencia entre las estimaciones de cada participante, se toma como resultado la media.
- Si la estimación resulta "infinito", la funcionalidad debe dividirse en sub-tareas.

- Si las estimaciones resultan muy dispares el team leader, con su criterio de gestión y basándose en las características del proyecto, equipo, reunión, nº de funcionalidades pendientes de evaluar. puede optar por:
  - Preguntar a los que han presentado las estimaciones extremas: ¿Por qué crees que es necesario tanto tiempo?, y ¿Por qué crees que es necesario tan poco tiempo? Tras oír las razones, repetir la estimación.
  - Dejar a un lado la estimación de esa funcionalidad y retomar al final o en otro momento las funcionalidades que no alcanzan consenso.
  - Pedir al propietario del producto que descomponga la funcionalidad y valorar cada una de las nuevas funcionalidades resultantes.
  - Tomar la estimación menor, mayor o la media.

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

Scrum Management: Responsabilidades

Introducción

No es lo mismo "adoptar" formas ágiles, que "trabajar" de forma ágil.

Algunas organizaciones abordan el cambio en la capa de las formas de trabajo: incorporan backlogs, reuniones de Scrum, gráficos de seguimiento y desarrollo en intervalos cortos.

En estos casos las mejoras, si se producen, son relativamente pequeñas, y pueden no compensar a las tensiones que genera incorporar modos de trabajo sobre culturas, tipos de proyectos o modelos organizativos en los que no encajan.

Otras saben que no sólo implica cambios en la forma de hacer las cosas, que no afecta sólo a los programadores y que también necesita una cultura adecuada en la organización.

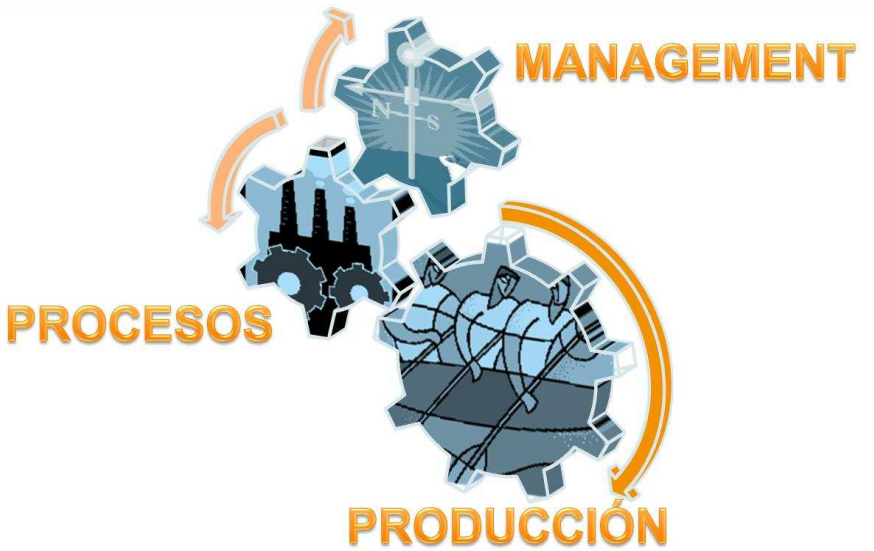
No debe plantearse como objetivo "adoptar" prácticas más o menos ágiles…

Pero atención, que el objetivo tampoco es trabajar de forma ágil…

El objetivo es trabajar de la forma más adecuada a las características de la empresa y del tipo de trabajos que desarrolla.

El éxito en la adopción de un modelo de management Scrum, no depende del nivel profesional y de responsabilidad de un profesional incorporado como ScrumMaster .

La adopción de una scrum management será exitosa en la medida del nivel de competencia y responsabilidad que ofrezca la organización a su implantación en tres áreas: management o gestión de la organización, calidad o procesos, y producción



Responsabilidades de Management

Equilibrio sistémico de la empresa

La organización tiene una visión, misión y valores reales y coherentes. Las personas, cultura, procesos y métodos de las distintas áreas de la empresa son adecuados están alineados con la visión y misión.

Coherencia del modelo

La decisión de implantar un modelo ágil de trabajo responde a la coherencia de los principios ágiles con las características de la empresa y sus proyectos.

Medios y formación

La dirección provee de los recursos necesarios para la puesta en marcha y funcionamiento de las prácticas ágiles, y la formación adecuada para las personas.

Responsabilidades de Procesos

Configuración de Scrum

Se analizan las características y medios de la organización, y se diseñan las formas y prácticas ágiles más adecuadas.

Mejora continua

De forma regular se toma información sobre el funcionamiento del modelo, se analiza y se determinan acciones para mejorar la configuración de Scrum.

Garantía de funcionamiento de Scrum

De forma continua se monitoriza la implantación y funcionamiento de Scrum en los proyectos y se identifican:

- Impedimentos en los proyectos para que el equipo pueda llevar a cabo el objetivo del sprint.
- Prácticas o decisiones en la organización que impiden o dificultan la metodología Scrum.

Responsabilidades de producción

Visión del producto

El cliente sabe cuáles son sus necesidades y el resultado que desea obtener; así como las restricciones que impone el negocio sobre el desarrollo y en base a ello define las funcionalidades que necesita y la prioridad en la que deben desarrollarse.

Auto-organización

El equipo del proyecto (técnicos y cliente) trabaja de forma auto-gestionada: prioridades, estimación, asignación de tareas, decisiones técnicas…

Tecnología ágil

El equipo técnico conoce y emplea modelos y medios de desarrollo ágil: integración continua, pruebas automáticas, refactorización…



¿Responsabilidades o roles?

Scrum Management, aborda la implantación de principios ágiles en la empresa desde la visión de la organización en conjunto; como un sistema inter-relacionado y diseñado para alcanzar el mejor equilibrio y alineación hacia la visión y misión de la empresa.

Por eso, además de las responsabilidades "directas" del equipo técnico y del propietario del producto, también se consideran las "indirectas" de la dirección de la organización, que normalmente son tan o más importantes para obtener el mayor beneficio de la implantación de modos de trabajo ágiles.

¿Por qué integrar principios ágiles?

¿Para la misión de la empresa y el tipo de proyectos que realiza, resulta mejor abordarlos con patrones predictivos o ágiles?

¿Es adecuada la estructura organizativa de la empresa? ¿La cultura?

¿Qué acciones de gestión se deben realizar en paralelo a la adopción de formas ágiles para armonizarlas?

¿Las personas tienen niveles de empowerment suficientes para trabajar en equipos realmente auto-gestionados?

Estas cuestiones son muy importantes para implantar nuevos principios de desarrollo en una empresa, por eso la implantación de modos ágiles implica a más "roles" que los exclusivamente técnicos.

Cada empresa tiene configurada su estructura, departamentalización y procesos de la forma más adecuada sus características de tamaño, distribución física, cultura, producto, etc.

Resulta más difícil integrar Scrum en la empresa, si el modelo prescribe roles fijos que suelen suponer modificaciones en la estructura organizativa.

Se consigue una mejor integración considerando como premisa que la razón para lograr trabajo en campos de scrum son las responsabilidades, y no los roles, y que por tanto el grado de beneficio es proporcional a la medida y eficiencia con la que se cubren aquellas.

Para lograr trabajo de equipo en campos de Scrum es más importante pensar en las responsabilidades de toda la organización que en los roles del equipo.

En definitiva se trata de lograr la incorporación de los principios ágiles, construyendo un modelo adecuado para la empresa, y no al contrario.

Para enfocar la adopción de un modo de trabajo ágil de forma global y flexible, hay que considerar que se trata de cubrir con eficiencia las nueve responsabilidades siguientes; mejor que pensar en roles (propietario de producto y gestor de Scrum).

Responsabilidad	Asignaciones más comunes
1.- Equilibrio sistémico	*Dirección*
2.- Coherencia del modelo	*a) Dirección* *b) Área de calidad / procesos* *c) Asesoría / consultoría externa*
3.- Medios y formación	*a) Dirección* *b) Dirección y RR.HH.*
4.- Configuración de Scrum	*a) Área de calidad / procesos* *b) Dirección técnica* *b) Asesoría / consultoría externa*
5.- Mejora continua	*a) Área de calidad / procesos* *b) Dirección técnica* *c) Asesoría / consultoría externa* *d) Oficina de proyectos*
6.- Garantía de funcionamiento	*a) Área de calidad / procesos* *b) Oficina de proyectos* *d) Dirección técnica* *e) Team leader*
7.-Visión del producto	*a) Área comercial* *b) Product manager* *c) Cliente*
8.- Auto-organización	*a) Oficina de proyectos* *b) Team leader* *c) Equipo*
9.- Tecnología ágil	*a) Dirección técnica* *b) Equipo*

Los 9 factores clave para una Scrum Management

Es la forma del modelo y de las prácticas las que deben facilitar la ejecución de los principios ágiles, adaptándose a las características de la empresa, y no al contrario.

El grado de beneficio que obtendrá la organización con la adopción de un modelo Scrum está en relación directa con la medida de nivel y competencia con la que cubra las responsabilidades apuntadas:

El resultado no depende tanto de quién cubre las responsabilidades, sino de que éstas estén asignadas a los roles o puestos más apropiados de la organización, y que éstos tengan el nivel de capacitación suficiente.

Si la organización dispone de una persona o departamento responsable de calidad, o procesos… es el mejor candidato para asumir la responsabilidades de este grupo.

Si se trata de una organización más pequeña que no cuenta con un departamento de calidad pero sí con uno de gestión de proyectos, éstas personas son las mejores candidatas para asumir el conocimiento y la responsabilidades de garantía de Scrum, configuración..

Es posible que la de configuración se asigne a una consultoría externa o que la asuman los profesionales del departamento de procesos…

Es posible que….

No hay talla única porque no hay dos empresas iguales.

Trabajos citados

April, A., Laporte, C. Y., & Renault, A. (2006). Applying ISO/IEC Software Engineering Standards in Small Settings: Historical Perspectives and Initial Achievements. *Proceedings of SPICE Conference*. Luxembourg: Department of Software and IT Engineering, Canada.

Bauer, F., & Helms, F. (1968). NATO SCIENCE COMMITEE. *Software Engineering*. Garmisch: Peter Naur & Brian Randell.

Beck, K. (1999). *extreme Programming explained Embrace Change*. Addison-Wesley.

Beck, K., Beedle, M., Bennekum, A. v., Cockburn, A., Cunningham, W., Fowler, M., y otros. (Marzo de 2001). *Manifiesto for Agile Software Development*. Recuperado el 01 de 09 de 2007, de http://www.agilemanifesto.org

Beth Chrissis, M., Konrad, M., & Shrum, S. (2003). *CMMI Guideliness for Process Integration and Product Improvement*. Boston: Addison Wesley.

Boehm, B., & Turner, R. (2004). *Balancing Agility and Discipline*. Addison Wesley.

Brassens, G. (Compositor). (1952). La Mauvaise Reputation.

CMMI Models, Modules, and Reports. (s.f.). Obtenido de http://www.sei.cmu.edu/cmmi/models

Constantine, L., Myers, G., & Stevens, W. (1974). Structured Design. *IBM Systems Journal* , 115-139.

Dijkstra, E. W. (Marzo 1968). GoTo Statement Considered Harmful. *The Guide to Computing Literature* , 147-148.

Garcia, S., Graettinger, C., & Kost, K. (2006). Proceedings of the First International Research Workshop for Process Improvement in Small Settings. *First International Research Workshop for Process Improvement in Small Settings*. Pittsburgh: Carnegie Mellon, Software Engineering Institute.

Gilb, T. (1977). *Software Metrics.* Cambridge, Massachusetts: Winthrop Publisher, Inc.

Group, T. S. (1994 - 2004). *Chaos Report.* Boston.

IEEE Computer Society. (2000). *Guide to the Software Engineering Body of Knowledge SWEBOK.* Los Alamitos, California: IEEE.

IEEE. (1998). IEEE Std 1362-1998 IEEE Guide for Information Technology System Definition Concept of Operations (ConOps) Document Description. IEEE Standards Association.

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

Ikukiro, N., & Hirotaka, T. (1995). *The Knowledge-Creating Company.* Oxford University Press.

Imai, H., Nonaka, I., & Takeuchi, H. (1985). Managing the New Product Development Process: How Japanese Companies Learn and Unlearn. *The uneasy alliance* , 337-375.

ISO/IEC. (2004). ISO/IEC 12207 . *International Standard ISO/IEC 12207 Information Technology Software Life Cycle Processes* . ISO.

Jericó, P. (2001). *La gestión del talento.* Madrid: Prentice Hall.

Laporte, C. Y., & April, A. (2005). Applying Software Engineering Standards in Small Settings: Recent historical Perspectives. *International Research Workshop for Process Improvement in Small Settings.* Pittsburg: Department of Software and IT Engineering, Canada.

Norden, P. (Julio 1958). Curve Fitting for a Model of Applied Research and Development Scheduling. *IBM Journal* , 232-248.

PMI. (2005). *PMBOK.* Philadelphia: Project Management Institute.

Schwaber, K., & Sutherland, J. (1996). *Controlled Chaos: Living on the Edge.* San José, California: OOPSLA.

Sutherland, J., Viktorov, A., & Blount, J. (2006). *Adaptive Engineering of Large Software Projects with Distributed/Outsourced Teams.* Boston: International Conference on Complex Systems 2006.

Takeuchi, H., & Nonaka, I. (2004). *Hitotsubashi on Knowledge Management.* Singapore: WILEY.

Wujec, T., & Muscat, S. (2002). *Return on Imagination: Realizing the Powr of Ideas.* London: Prentice Hall.

Índice

A

adquisición · 119

Agile Modeling · 88

agilidad · 59

  etapas de un modelo ágil · 61

  principios - fondo · 61

auto-gestión · 42

auto-organización · 42

C

campos de Scrum · 40

  características · 41

capital estructural · 96

capital humano · 96

cierre · 63

CMM - CMMI · 85, 104

CMMI · 113

concepto · 62

crisis del software · 81

Crystal · 88

cultura · 77

cultura de cumplimiento · 29, 30

D

desarrollo secuencial · 38, 39

dialéctica · 103

DSDM · 86

E

ejecución · 100

especulación · 62

estimación · 134

estimación de póker · 133, 165

exploración · 63

Extreme Programming · 87, 106

F

FDD · 88

flexibilidad · 59

G

gestión de proyectos

origen · 21

patrones comunes · 23

gestión de proyectos predictiva

ámbito · 26

características · 68

definición · 20

errores frecuentes · 26

premisas · 67

principios · 24

gestión sistémica · 107

gráfico burn-down · 133, 164

gráfico burn-up · 133, 161

I

incertidumbre · 34, 41

incremento · 131, 146, 159

ingeniería concurrente · 57

fertilización cruzada · 58

ingeniería de procesos · 116

innovación · 57

sistemática y continua (kaizen) · 59

institucionalización · 116

IPMA · 22, 104

ISO 9000-3 · 85, 104

ISO/IEC 12207 · 85, 113, 119

ISO/IEC 15504 · 85, 106, 113

K

kaizen · 59

M

maleabilidad · 72

manifiesto ágil · 47

mantenimiento · 121

métricas · 134, 166

moprosoft · 105

motivación · 52

P

personas · 100, 109

conocimiento · 110

ejecución · 110

pizarra de trabajo · 152

PMBOK · 85

PMI · 22, 26, 104

Pragmatic Programming · 88

Prince2 · 22

procedimientos · 100

procesos · 100, 108

crítica · 96

fortalezas · 93

madurez · 91

organizacionales · 121

product backlog · 120, 130, 138, 140, 142

propietario del producto · 130, 131, 139

prototipado · 72

proyecto

diferencia con operaciones · 19

puntos de funcionalidad · 134

R

refactorización · 59, 127

requisitos · 138

reuniones Scrum

planificación del sprint · 129, 147

revisión del sprint · 130, 158

seguimiento del sprint · 129, 157

revisión · 63

rutinas · 100

S

Scrum · 87, 106, 125

responsabilidades · 131

valores · 135

Scrum Management · 107, 111

criterios de gestión · 115

principios · 112

responsabilidades · 169

scum manager · 132

SEI · 84

síntesis · 103, 110, 113, 114

software

coste · 99

factor de escala · 101

maleabilidad · 100

talento · 100

solapamiento · 38, 41, 43

sprint · 126, 129

sprint backlog · 129, 131, 139, 143, 151

suministro · 119

SWEBOK · 104

T

talento · 77, 100

tiempo

real o de trabajo · 134

teórico · 134

Time to market · 58

V

valor · 56

velocidad · 31

velocidad absoluta · 134

velocidad relativa · 134

visión · 62