Los Fundamentos de Agile Scrum

Nader K. Rad y Frank Turley

Van Haren PUBLISHING

Other publications by Van Haren Publishing

Van Haren Publishing (VHP) specializes in titles on Best Practices, methods and standards within four domains:

- IT and IT Management
- Architecture (Enterprise and IT)
- Business Management and
- Project Management

Van Haren Publishing is also publishing on behalf of leading organizations and companies: ASLBiSL Foundation, BRMI, CA, Centre Henri Tudor, Gaming Works, IACCM, IAOP, IFDC, Innovation Value Institute, IPMA-NL, ITSqc, NAF, KNVI, PMI-NL, PON, The Open Group, The SOX Institute.

Topics are (per domain):

IT and IT Management

ABC of ICT
ASL®
CATS CM®
CMMI®
COBIT®
e-CF
ISO/IEC 20000
ISO/IEC 27001/27002
ISPL
IT4IT®
IT-CMF™
IT Service CMM
ITIL®
MOF
MSF
SABSA
SAF
SIAM™
TRIM
VeriSM™

Enterprise Architecture

ArchiMate®
GEA®
Novius Architectuur
Methode
TOGAF®

Business Management

BABOK® Guide
BiSL® and BiSL® Next
BRMBOK™
BTF
EFQM
eSCM
IACCM
ISA-95
ISO 9000/9001
OPBOK
SixSigma
SOX
SqEME®

Project Management

A4-Projectmanagement
DSDM/Atern
ICB / NCB
ISO 21500
MINCE®
M_o_R®
MSP®
P3O®
PMBOK® Guide
Praxis®
PRINCE2®

For the latest information on VHP publications, visit our website: www.vanharen.net.

Los Fundamentos de Agile Scrum

Nader K. Rad

Frank Turley

Van Haren PUBLISHING

Colofón

Título:	Los Fundamentos de Agile Scrum
Autor:	Nader K. Rad y Frank Turley
Editorial:	Van Haren Publishing, 's-Hertogenbosch
ISBN Copia impresa:	9789401805346
Edición:	Primera impresión, primera edición, 1 de octubre de 2019
Diseño:	Van Haren Publishing, 's-Hertogenbosch
Derechos de autor:	© Van Haren Publishing 2019
	Para más información sobre Van Haren Publishing, pueden contactar por correo electrónico a: info@vanharen.net o visitar nuestra web: www.vanharen.net

Contenido

Nota sobre los Autores

Nader K. Rad es escritor, conferenciante y asesor de Gestión de Proyectos en Management Plaza. Su carrera comenzó en 1997 y ha estado involucrado en multitud de proyectos de distintas industrias. Ha diseñado varios cursos de gestión de proyectos, preparado diversos cursos de formación online y escrito más de 40 libros.

Más info sobre el autor: http://nader.pm
Página web del autor: https://mplaza.pm
Perfil LinkedIn del Autor: be.linkedin.com/in/naderkrad

Frank Turley ha sido Project manager durante más de 15 años. Es PRINCE2®Practitioner, Scrum Master y formador y coach de gestión de proyectos y PRINCE2. Ha escrito varios libros relacionados con PRINCE2® y gestión de proyectos y se le conoce en el mundo de PRINCE2 por la creación del material más popular de formación para autoaprendizaje de PRINCE2.

Más info sobre el autor: https://mplaza.pm/frank-turley/
Página web del autor: https://mplaza.pm
Perfil LinkedIn del Autor: http://linkedin.com/in/frankturley

1. EL CONCEPTO AGILE

Si su objetivo es aprender algo que le pueda beneficiar en sus proyectos, debe reflexionar sobre dos temas que a menudo se malinterpretan:

1. Puede que a menudo escuche la frase, "Agile es una forma de pensar". La verdad es que Agile requiere una determinada forma de pensar, como todo, pero no es correcto decir que es una forma de pensar. Decir "Agile es una forma de pensar", en la práctica, solo lleva a una cosa: poder trabajar como uno quiera, llamándolo Agile, sin aceptar críticas ni buscar mejoras reales.
2. Si tiene el más mínimo conocimiento de cómo funcionan los sistemas autoritarios, sabrá que siempre tiene que haber un *enemigo*. Este concepto cubre los agujeros que pueda tener su sistema y ayuda a controlar a las masas. Muchos profesionales de Agile usan la palabra "cascada" para referirse al enemigo; y mientras que el concepto "cascada" no se acaba de definir del todo, se insinúa que son los sistemas de gestión de proyectos ya establecidos y conocidos. Si su objetivo es tener éxito en proyectos, no necesita crear la ilusión de un enemigo externo. Y recuerde que todo sistema de éxito se construye sobre sistemas existentes, sin tener que empezar de cero. Y aunque la crítica es absolutamente necesaria, debe hacerse desde el respeto y el conocimiento.

Así pues, hablemos de la verdadera naturaleza de Agile.

METODOLOGÍA DE ENTREGA DE PROYECTO Y CICLO DE VIDA

Cuando se desarrolla software, de una manera u otra, se realizan los siguientes pasos, bien para funcionalidades individuales o para la solución completa:

- Analizar
- Diseñar
- Desarrollar
- Integrar
- Prueba (Test)

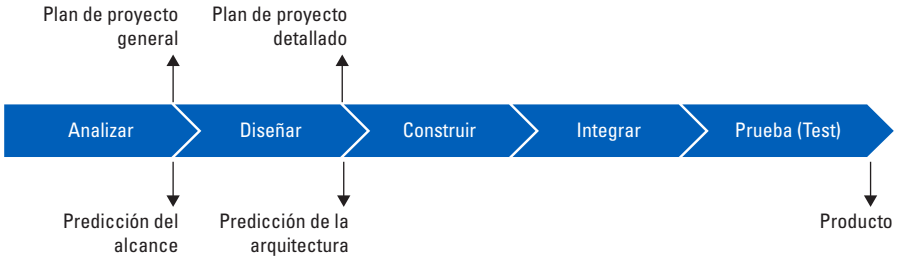
Por supuesto, se puede usar otra terminología para esos pasos, o agruparlos en menos pasos, o dividirlos en más; está bien. Estos pasos los podemos llamar *procesos de entrega*.

Ahora, la pregunta es, ¿Cómo vamos a gestionar y realizar estos procesos? Piense en algunas opciones antes de leer el resto de este capítulo.

¿En cuántas opciones ha pensado?

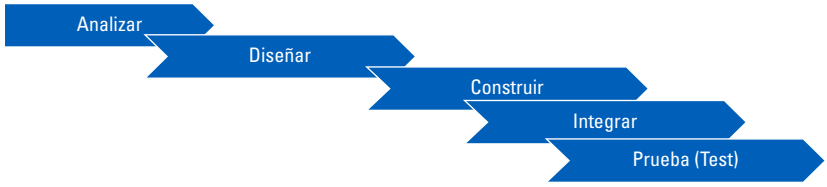
Puede que tenga muchas opciones en mente, pero todas deben pertenecer a una de las dos formas genéricas que hay. A propósito, estas opciones las podemos llamar el *ciclo de vida del desarrollo*.

Un ciclo de vida genérico es algo así:



En este ciclo de vida, cada proceso se debe completar antes de proceder al siguiente; es decir, analizamos por completo el requisito y decidimos qué queremos que contenga la solución. Entonces, diseñamos la arquitectura de la solución y averiguamos la mejor manera de dar forma a las características. Entonces, los programadores empiezan a trabajar en las distintas unidades y después las unidades se integran en una solución. Y esa solución se prueba.

Es obvio que los pasos se pueden solapar; por ejemplo, no es necesario esperar a que todas las unidades estén completas antes de integrarlas y probarlas. Su ciclo de vida puede tener el siguiente aspecto:



En esencia, no es distinto del ciclo de vida anterior; seguimos teniendo una secuencia de procesos de desarrollo como motor principal del ciclo de vida.
Como podrá observar, este tipo de ciclo de vida se basa en un esfuerzo inicial por entender qué es lo que vamos a producir. Tenemos una especificación por adelantado, un diseño por adelantado y, por consiguiente, un plan por adelantado. Por eso, a veces se le llama un desarrollo *dirigido por el plan*. También intentamos predecir qué es lo que queremos y cómo se puede producir, y por eso también se le suele llamar *predictivo*.

Un Ciclo de vida Predictivo es la manera habitual y apropiada de desarrollar muchos proyectos, como por ejemplo un proyecto de construcción. En primer lugar, se planifica y diseña, y luego se sigue ese plan y diseño. Sin embargo, esto no es cómodo para algunos tipos de proyectos.

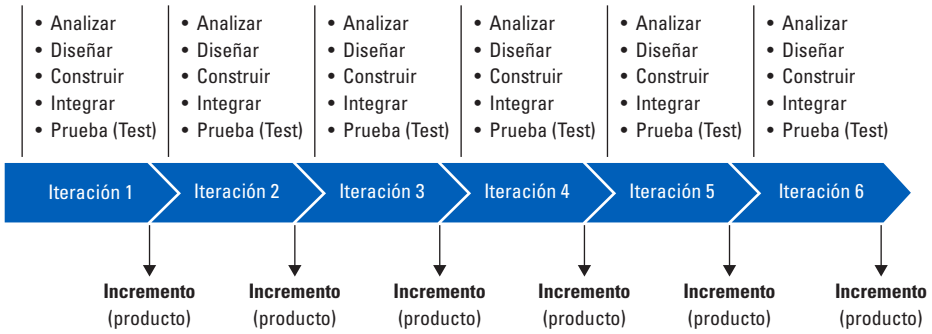
Piense en el típico proyecto de desarrollo de TI. Puede dedicarle mucho tiempo a la especificación y análisis de los requisitos, y basarlo todo en eso. ¿Qué ocurre después? ¡Que el cliente no estará contento cuando vea el resultado! Pedirá cambios, y los cambios son caros en este ciclo de vida porque es posible que haya que revisar todo el trabajo anterior.

Como se suele decir en este sector, el cliente no sabe lo que quiere hasta que ve el producto. ¿Cuándo ven el producto? Hacia el final del proyecto. En ese punto, el coste de cambiar es máximo.

Para superar este problema, podemos renunciar a la comodidad y a la estructura de un ciclo de vida predictivo y usar uno que cree el producto de forma *incremental*, es decir en múltiples versiones, cada vez con más características. Este es un lujo que tenemos en los proyectos de desarrollo de TI que no puede tener todo el mundo: múltiples versiones de software funcional, cada vez con más características. Piense en un proyecto de construcción, no hay incrementos significativos y el producto no se puede utilizar hasta el final.

Para ser justos, esta desventaja de un proyecto de construcción se compensa con el hecho de que si se tiene que empezar un proyecto para construir un hospital, el resultado final será un hospital, con independencia de la cantidad de cambios que haga, y no, por ejemplo, ¡un parque de atracciones! Sin embargo, en desarrollo de TI, se puede empezar un proyecto para crear algo parecido a un hospital y acabar con algo parecido a un parque de atracciones.

Por lo tanto, en los proyectos de desarrollo de TI, podemos tener entregas incrementales: aprovechemos esta oportunidad mediante un ciclo de vida como el siguiente:



No hay una predicción real en este ciclo de vida. En vez de predecir el producto y depender de esa predicción, tenemos pequeños periodos de tiempo durante los cuales creamos incrementos del producto. Mostraremos ese incremento (la última versión del producto) al cliente y a los usuarios finales, recibiremos su feedback (sus comentarios al respecto), y decidiremos qué hacer en el siguiente periodo de tiempo. Así que, en vez de basarnos en la predicción, seguimos con el proyecto y nos *adaptamos* al feedback. ¿Cómo queréis llamar a este ciclo de vida? "Adaptativo" es un buen nombre: ciclo de vida adaptativo.

Para crear cada incremento, necesitamos ejecutar todos los procesos de desarrollo durante ese periodo de tiempo. En el siguiente periodo, repetiremos los procesos: *iteramos*. Por eso, este método de desarrollo se llama a veces *desarrollo iterativo*. Los periodos de tiempo durante los cuales iteramos, se pueden llamar iteraciones. No es el único nombre que se utiliza para ello. Puede que ya conozca por lo menos un nombre más para las iteraciones. Volveremos pronto a este tema.

CICLOS DE VIDA PREDICTIVOS VS ADAPTATIVOS

Tanto el ciclo de vida adaptativo como el predictivo, tienen ventajas y desventajas. Que la selección del ciclo sea la correcta depende de muchos factores, pero el más importante es el tipo de producto.

Se pueden hacer dos preguntas esenciales antes de decidir el tipo de ciclo de vida que necesita para su proyecto:

1. ¿Necesito poder adaptarme? Porque si no, un ciclo de vida predictivo es…. ¡Pues más predecible! Es más fácil y está más estructurado. Se necesita un sistema adaptativo cuando existe el riesgo de empezar con la idea de crear un hospital y acabar con un parque de atracciones.
2. ¿Puedo adaptarme? Esta pregunta es todavía más importante. Para ser adaptativo, se debe tener la posibilidad de desarrollar de formar *iterativa* y de entregar de forma *incremental*. Pensemos de nuevo en un proyecto de construcción: ¿puede construir el edificio de forma iterativa? ¿Puede diseñar la

base sin diseñar el resto del edificio que determinará la carga que debe soportar la base? ¡La respuesta es sencillamente NO! No es posible usar el desarrollo iterativo en un proyecto de construcción. Y la entrega de forma incremental tampoco es posible, como ya hemos visto. Así que no podemos usar un ciclo de vida adaptativo para construir un edificio (no nos confundamos con el diseño interior y la decoración, o incluso unas reformas, para las cuales sí es posible que podamos usar un sistema adaptativo).

Lo que quiero transmitir es que Predictivo vs Adaptativo no es una cuestión que una cosa es buena y la otra mala.

Como pequeño ejercicio, piense en un proyecto de TI para actualizar los sistemas operativos de 300 ordenadores de una organización o en un proyecto de TI para crear una infraestructura de red para una organización enorme con seis oficinas. En su opinión, ¿qué ciclo de vida de desarrollo es mejor para estos dos proyectos?

AGILE VS MÉTODO EN CASCADA (WATERFALL)

"Agile" es el nombre más común para los sistemas que utilizan los ciclos de vida Adaptativos. Así es como se puede definir de verdad "Agile", en vez de decir "¡Agile es una mentalidad!"

Los "fans" de Agile utilizan el término Cascada para referirse a los Ciclos de vida Predictivos. La palabra Cascada se utiliza a menudo para referirse a los Ciclos de vida Predictivos usados en proyectos de TI; no se oye a nadie decir "Este edificio se construyó usando el método en Cascada".

Para asegurarse de que conoce a fondo la terminología, debe ser consciente de que decir método en Cascada es prácticamente una grosería hoy en día, y ¡tiene derecho a enfadarse y a ofenderse si alguien le dice que está usando el método en Cascada! Por eso sugiero que usemos el concepto más formal en este libro: Ciclo de vida Predictivo.

¿ES NUEVO AGILE?

Normalmente, Agile se anuncia como la novedad. Ciertamente, el uso del concepto Agile refiriéndose a los Ciclos de vida Adaptativos es nuevo, pero ¿el ciclo de vida en sí lo es?

No sé a los demás pero a mí me cuesta imaginar la larga historia del ser humano con tantos proyectos y programas que se habrá realizado sin que hubiera ningún tipo de ciclo de vida adaptativo. ¿Se le ocurre algún ejemplo?

Le propongo uno. Piense en una iniciativa (programa o proyecto) muy popular en los viejos tiempos: el ir a la guerra. ¿Se puede gestionar una guerra con un enfoque Predictivo? ¿Planifican y diseñan todo desde el comienzo? Desde luego que no. Puede que haya un plan inicial general que se parezca más a una estrategia que a un plan, y que se gestione la guerra de batalla (es decir, iteración) en batalla (o con varias a la vez), y en función del resultado de cada batalla, se adapta el resto de la iniciativa.

No es un ejemplo agradable pero es un ejemplo claro de que los Ciclos de vida Adaptativos no pueden ser nuevos.

Entonces, ¿cuál es la novedad?

En un momento dado, el enfoque de gestión conocido como el científico y el Taylorismo se convirtieron en la norma, tanto es así que cualquier otro enfoque se percibía como inferior e incluso equivocado. El Taylorismo se basaba plenamente en sistemas Predictivos. Por lo tanto, los sistemas Predictivos dominaban el mundo, por así decirlo.

Después, llegamos a un momento en el que se iniciaban más y más proyectos de desarrollo de TI y los Ciclos de vida Predictivos no eran la mejor manera de gestionar aquellos proyectos. Las personas intentaron aguantar, mientras que aumentaba la presión hasta que hubo manifestaciones y, finalmente, ¡la revolución! Como cualquier otra revolución, devoró a sus retoños pero ese es un tema para otro momento.

EL MANIFIESTO AGILE

Algunas personas comenzaron a usar sistemas Adaptativos para el desarrollo de TI y, poco a poco, los fueron organizando en procesos de gestión que se podían repetir. Un grupo de pioneros se juntó en el 2001 para formalizarlos, dándoles nombre y creando un manifiesto.

Empecemos por el nombre. La leyenda dice que las dos opciones al final era Agile (Ágil) y Adaptive (Adaptativo).

Lamentablemente, se quedaron con Agile. Adaptive habría sido mucho mejor porque describe la naturaleza del enfoque y evita muchos malentendidos.

Así pues, a continuación le mostramos el Manifiesto Agile. Está disponible en la página web AgileManifesto.org, muy moderna y avanzada.

> Estamos descubriendo formas mejores de desarrollar software tanto por nuestra propia experiencia como ayudando a terceros. A través de este trabajo hemos aprendido a valorar:
>
> | | | |
> |---|---|---|
> | Individuos e interacciones | sobre | procesos y herramientas |
> | Software funcionando | sobre | documentación extensiva |
> | Colaboración con el cliente | sobre | negociación contractual |
> | Respuesta ante el cambio | sobre | seguir un plan |
>
> Esto es, aunque valoramos los elementos de la derecha, valoramos más los de la izquierda.

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

Lamentablemente, este manifiesto en sí no ha sido adaptado en lo que tiene de vida.

La última frase normalmente se pasa por alto. Le invito a que lea de nuevo el manifiesto teniendo en cuenta la última frase.

Así pues, revisemos estas cuatro declaraciones.

Valor 1: Individuos e interacciones sobre procesos y herramientas

Restarle importancia a los individuos y a las interacciones es una manera muy rápida de fracasar. Al fin y al cabo, son las personas las que realizan el proyecto. Algunos miembros de dirección piensan que pueden superar problemas en este ámbito usando un "sistema" más sofisticado pero eso no funciona casi nunca, o nunca.

Muchos nos hemos visto decepcionados por el optimismo ingenuo de pensar que, al implementar una herramienta sofisticada, se iban a solucionar problemas causados por no tener en cuenta el aspecto humano, o incluso sus métodos. Aun así, los responsables se gastan cantidades enormes de dinero implementando y manteniendo herramientas, con la esperanza de que sean mágicas. El hecho es

que las herramientas solo pueden facilitar un sistema; no sustituyen la necesidad de tener un sistema en sí. El lado positivo es que estas herramientas son programas de software sofisticados que necesitan años de desarrollo y mantenimiento y crean muchos proyectos y empleos, y ¡nos ofrecen la posibilidad de invertir en pensar en cómo mejorar nuestra forma de realizar proyectos de TI!

La parte sobre procesos en esta declaración es un poco delicada. En realidad, habla de un tipo concreto de proceso, no de los procesos en general. Trata de aquellos procesos que han sido diseñados para sustituir la necesidad de la interacción humana y sus complejidades. Personalmente, conozco directores que creen que si tienen un mejor proceso, no tendrán que contratar a expertos profesionales. Mientras tanto, uno de los aspectos geniales de los sistemas Agile es que TODOS tienen integrados el aspecto humano en sus procesos, en vez de meterlo a la fuerza o incluso limitarse a comentar la importancia de la faceta humana, lo cual ocurre, lamentablemente, con los sistemas de gestión de proyectos ya establecidos.

Por lo tanto, para resumir, los procesos que intentan ignorar o reemplazar el aspecto humano son malos, y los procesos que incluyen estos aspectos y los integran como parte del sistema son buenos.

Valor 2: Software funcionando sobre documentación extensiva

Al contrario de la declaración anterior, que es correcta para todo tipo de proyectos, esta es específica a los sistemas Adaptativos. Se refiere al hecho de que, en vez de utilizar documentación por adelantado para predecir lo que tiene que ocurrir en un proyecto, simplemente, trabajamos sobre partes de software operativos (incrementos) y los utilizamos para adaptar la solución.

Valor 3: Colaboración con el cliente sobre negociación contractual

Cualquier proyecto tendría más éxito si tuviera un nivel de colaboración más alto del cliente. En los sistemas Adaptativos, es más que importante: es necesario. El cliente tiene que colaborar con nosotros todo el tiempo ya que estamos constantemente especificando nuevos requisitos y requiriéndole que compruebe los incrementos y que nos dé feedback. Si no lo hace, no podremos adaptar la solución.

Y a todos nos encanta la negociación contractual ☺ Un proyecto Agile ideal, con un contrato de tiempo y material y un cliente que no cree que los proveedores son delincuentes, no necesita mucha negociación contractual. Las dos partes trabajan conjuntamente para crear un producto de valor. Sin embargo, el ideal es tan solo el ideal, y los clientes siguen buscando contratos que delimitan el

ámbito y el precio, que suponen una contradicción fundamental con los métodos Adaptativos, lo cual es una fuente de negociaciones interminables de contrato similares a las de los proyectos Predictivos.

Valor 4: Respuesta ante el cambio sobre seguir un plan

Esta declaración, como la segunda declaración, es específica a los sistemas Adaptativos. En vez de tener un plan por adelantado, Predictivo, que nos muestra el camino, dependemos de la adaptación. A esto último, se le suele llamar "cambio" en Agile, probablemente porque hace feliz a los clientes saber que pueden cambiarlo todo, aunque de hecho, no es un cambio salvo que no cuadre con el plan de base inicial que no tenemos en los sistemas Adaptativos. Técnicamente, lo que tenemos es un flujo continuo de ideas nuevas. Sin embargo, continuemos llamándolos cambios, por el bien de todos los clientes.

- Muy probablemente haya preguntas sobre el Manifiesto Agile en el examen. No es mala idea revisarlo varias veces y poder incluso memorizar las cuatro declaraciones.

LOS PRINCIPIOS AGILE

El Manifiesto Agile es felizmente breve. Sin embargo, los autores consideraron que sería buena idea elaborar un poco la recién nombrada idea de Agile, así que crearon estos doce principios:

Principio 1: Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.

Al fin y al cabo, esto es un negocio y necesitamos que los clientes estén felices. Eso está claro. Pues, hoy en día, se dice que la satisfacción del usuario final es la medida definitiva porque eso le generará beneficios al cliente y, tarde o temprano, satisfará al cliente de una forma sostenible. ¿Es demasiado idealista?

Entonces, ¿cómo les satisfacemos? Por el software que creamos, que tiene el potencial de generarle valor (por ejemplo, dinero). Cuando hacemos entregas de forma anticipada y continua, generaremos valor antes, y además nos da la oportunidad de adaptar la solución y crear algo que el mercado quiere realmente y por lo que pagará, en vez de crear algo que nosotros esperamos que quiera.

Principio 2: Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos Ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente.

Hagamos un poco más de marketing acerca de la palabra "cambio" que tanto les encanta a los clientes ☺

Principio 3: Entregamos software funcional frecuentemente, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible.

¿Recuerda las iteraciones de las que hablamos – esos periodos de tiempo en los que iteramos (repetimos los procesos de desarrollo) para crear un incremento del producto? Este principio dice que no deben durar más que un par de meses. En Scrum, el plazo máximo es de un mes. Hablaremos mucho sobre ello de aquí al final del libro.

¿Ve también la sugerencia de un par de semanas? Muchos se reían en aquella época – la idea de tener un incremento nuevo en tan solo un par de semanas. Sin embargo, ahora tenemos proyectos con iteraciones incluso más cortas.

Principio 4: Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.

Esto va en contra de la idea de separar a los responsables de negocios (clientes u otros) de los técnicos, lo cual sigue siendo un problema en proyectos hoy en día. A veces, se consideran el enemigo el uno al otro, lo cual no es beneficioso para un proyecto.

Además, no podemos realizar adaptaciones si los responsables del negocio no están disponibles en todo momento. Piense en el análisis continuo de las nuevas características y en las pruebas de las unidades completadas. Además, ¡siempre será más divertido, cuántas más personas celebren el haber completado otra iteración!

Principio 5: Los proyectos se desarrollan en torno a individuos motivados. Hay que darles el entorno y el apoyo que necesitan, y confiarles la ejecución del trabajo.

Pronto, hablaremos de más aspectos de los sistemas Adaptativos. Uno de ellos es que necesitamos tener a personas empoderadas a nivel de proyecto; no solo porque es bueno, sino porque el ciclo de vida Adaptativo lo requiere. Quizá