Revista Española de Innovación, Calidad e Ingeniería del Software

REICIS

Volumen 6, No. 4, diciembre, 2010

Web de la editorial: www.ati.es
Web de la revista: www.ati.es/reicis
E-mail: calidadsoft@ati.es
ISSN: 1885-4486

Publicado por la Asociación de Técnicos de Informática (ATI), Via Laietana, 46, 08003 Barcelona.

Secretaría de dirección: ATI Madrid, C/Padilla 66, 3º dcha., 28006 Madrid

Revista Española de Innovación, Calidad e Ingeniería del Software (REICIS)

Contenidos

REICIS

Transición a equipos por funcionalidades en F-Secure

Juan Gutiérrez Plaza y Markku Kutvonen
F-Secure Corporation, Tammasaarenkatu 7, 00181, Helsinki, Finlandia
{juan.gutierrez, markku.kutvonen}@f-secure.com

Resumen

La capacidad de adaptación al cambio y la velocidad con la que las empresas de software responden a las necesidades de los clientes son cruciales a día de hoy. Trabajar con equipos por componentes puede crear cuellos de botella que dificultan la flexibilidad y tiempo de respuesta requeridos en este escenario. Este artículo muestra una experiencia real de cómo los equipos por funcionalidades pueden incrementar la flexibilidad, aumentar el rendimiento y la motivación en equipos de una empresa de desarrollo de software.

Palabras clave: Equipos por funcionalidades, equipos por componentes, transición

Transition to feature teams at F-Secure

Abstract

The ability of adapting to changes and quick answering the needs of the customers are important skills for the software companies nowadays. Working with component teams may create bottlenecks that make the flexibility and response time difficult which are key requirements in this scenario. This article shows a real experience on how feature teams may improve the agility, increase the performance and improve the motivation of teams in a software development company.

Key words: feature teams, component teams, transition

Gutiérrez-Plaza, J. y Kutvonen, M., "Transición a equipos por funcionalidades en F-Secure", REICIS, vol. 6, no.4, 2010, pp.29-42. Recibido: 9-5-2010; revisado: 20-10-2010; aceptado:3-12-2010

1. Introducción

La idea de equipos por funcionalidades no es nueva y se lleva aplicando durante años. En 1980, Microsoft ya utilizaba equipos multidisciplinares [1] que es la versión originaria de la cual se ha evolucionado a los equipos por funcionalidades.

Los equipos por funcionalidades se pueden definir como equipos de larga duración y multidisciplinares que completan funcionalidades para el cliente de principio a fin, una por una [2] (ver figura 1). En contraposición, los equipos por componentes enfocan sus esfuerzos en completar funcionalidades que sólo afectan a un componente determinado.

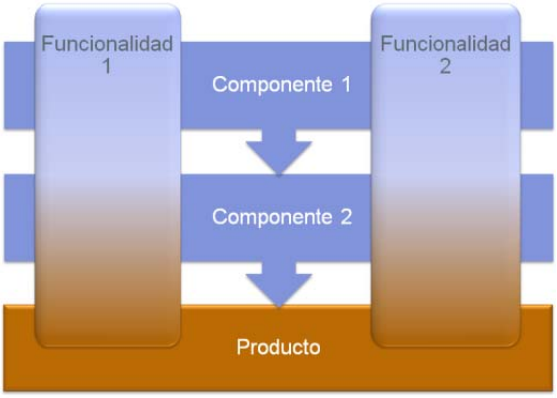

Figura 1. Comparación entre equipos por funcionalidad, por componentes y producto final

Un cambio de estas características que, en principio, puede parecer simple, puede traer muchos quebraderos de cabeza a una organización. Este tipo de cambios no sólo conllevan una modificación organizativa y estructural, sino también un cambio cultural, de valores y de modelo de trabajo dentro de la empresa.

Este artículo muestra cómo se ha realizado esta transición en una parte dentro de F-Secure para que pueda servir de ejemplo, modelo o ayuda a otras organizaciones que deseen hacer un cambio similar. Sin embargo, hay que tener muy en cuenta que el contexto es muy importante y hacer lo mismo en distintas organizaciones puede traer resultados dispares.

2. Antecedentes

F-Secure es una compañía de software finlandesa enfocada a productos de seguridad informática que cuenta con unos 900 trabajadores. El desarrollo se hace en varias oficinas situadas en distintos países.

La empresa empezó su transición a las metodologías ágiles en el 2005 de una manera escalonada, primero con algunos equipos pasando a involucrar más y más con el paso del tiempo.

El cambio a equipos por funcionalidades mostrado en este estudio se llevó a cabo en 2008 en un proyecto que involucraba alrededor de 50 personas en 5 equipos distintos. Todos los equipos estaban situados en la misma oficina.

En ese momento, todos los equipos eran equipos por componentes (aunque los integrantes del equipo eran multidisciplinares) que trabajaban con Scrum [3] a nivel de gestión y con algunas prácticas de eXtreme Programming (XP) [4] en el nivel más técnico. Estas prácticas eran dependientes de los equipos, sin haber una homogeneidad clara.

3. Problemas que llevan al cambio

Los equipos estudiados llevaban varios años trabajando con Scrum y usando ciertas prácticas de software ágiles. Durante estos años, en general, el rendimiento de los equipos mejoró considerablemente. A nivel de proyecto, los tiempos de entrega mejoraron notablemente con respecto a proyectos similares anteriores. Sin embargo, estas mejoras más o menos estables durante meses se estancaron, lo que llevó a hacer un análisis (mediante retrospectivas, cuestionarios y entrevistas informales) para encontrar problemas y continuar con el ritmo de mejoras alcanzado anteriormente.

Los problemas encontrados por el análisis fueron los siguientes:

- Muchas dependencias entre los equipos que causan retrasos.
- No se puede acortar el ciclo de entrega debido a las dependencias.
- Los equipos trabajando en varias funcionalidades simultáneamente tendían a cambiar de contexto más frecuentemente que los equipos trabajando en una sola funcionalidad. Esto causaba demoras además de un rendimiento poco óptimo.
- Varios equipos trabajando en la misma funcionalidad gastaban mucho tiempo en actividades de transferencia de control y coordinación.

- La especialidad de los equipos (especializados en componentes) causaba retrasos debido a un desequilibrio en la carga de trabajo que desencadenaba en cuellos de botella (algunos equipos estaban trabajando más allá de su límite mientras otros estaban bastante ociosos).
- La especialización de los equipos causaba, en ciertas ocasiones, incapacidad para entregar funcionalidades en orden de prioridad debido a las dependencias y a que las prioridades se tenían que establecer por equipo.
- La excesiva complejidad en la coordinación de todos los equipos solía causar retrasos en las entregas. Esto ponía mucha presión en la validación final para cumplir los plazos o retrasarlos lo mínimo posible lo que provocaba, finalmente y en la mayoría de los casos, un recorte de calidad y un incremento en la deuda de técnica (término introducido por primera en [5]) para futuras funcionalidades.
- Aunque se intentó acortar las iteraciones de 4 a 2 semanas, no se consiguió debido a que las dependencias entre los equipos traían muchos problemas.

4. Equipos por funcionalidades como solución

La transformación a equipos por funcionalidades en el proyecto se realizó formando los equipos de nuevo por completo (ver sección 4.1). Una vez formados, la organización de los equipos se hizo de la siguiente manera: de los 5 equipos, 4 fueron designados como equipos por funcionalidades para desarrollar nuevos requisitos. El equipo restante se especializó en mantenimiento y en soporte para la automatización de pruebas. Este equipo se estableció como un equipo rotante después de un tiempo. Cada cuatro meses (8 iteraciones, *sprints* en la jerga de Scrum, en nuestro caso de 2 semanas), un equipo nuevo tomaba este rol. Todos los equipos, por tanto, llegaron a ser el equipo de mantenimiento y soporte antes o después.

Esta nueva organización de los equipos dio los siguientes resultados:

- Menos dependencias que producen menos retrasos.
- Menos cambios de contexto lo que conlleva más efectividad en el trabajo.
- Los equipos producen la funcionalidad de principio a fin lo que minimiza las transferencias de control entre los equipos.

- Mayor conocimiento del producto en sí, y mayor transferencia de conocimiento de los distintos componentes, lo que da mucha más flexibilidad para los mismo recursos.
- A las funcionalidades se las puede dar prioridades libremente basadas en el ROI[1].
- Las cargas de trabajo son mucho más equilibradas, reflejándose en un mayor rendimiento y efectividad además de en menos horas extraordinarias.
- El estado de "listo para entrega" en el software se alcanza mucho más fácilmente.
- Los ciclos de entrega se acortan, pasando de un mínimo de 6 a 3 meses.

4.1. Pasos en el cambio

Una transformación de estas características lleva tiempo puesto que afecta a muchas personas. Hay que tener cuidado con los pasos dados (ver figura 2) para que el impacto sea el mínimo posible. Sin embargo, hay que aceptar desde el principio que el rendimiento se verá negativamente afectado al inicio de la transición.

Como regla general hay que mantener transparencia en todo momento y comunicar los cambios y las razones de una manera abierta además de abrir diálogo y debates para escuchar a todas las personas afectadas en la transformación.

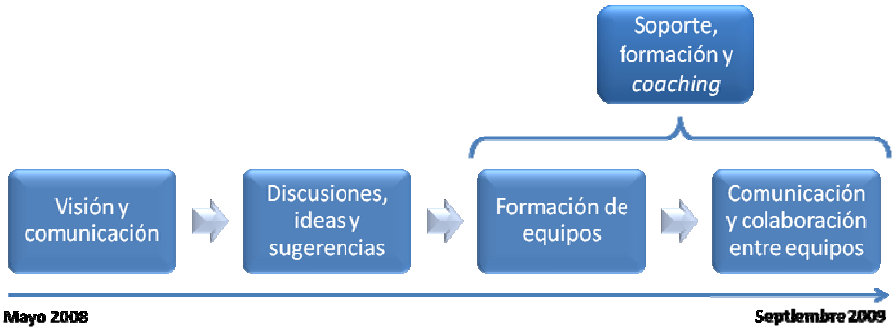

Figura 2. Pasos en la transición a equipos por funcionalidades

El primer paso fue reunir a todas las partes afectadas por el cambio y comunicar la visión, el porqué del cambio, los problemas actuales y las metas que se quieren alcanzar con la transformación. Los objetivos mostrados fueron los siguientes:

- Todos los equipos deben ser capaces de entregar elementos completos de la pila de funcionalidades.

[1] Retorno de inversión (*Return Of Investment* en inglés)

- Cualquiera debe ser capaz de participar en la implementación de nuevas funcionalidades.
- Todo el mundo debe aprender cosas nuevas.
- La prioridad de negocio y el esfuerzo definen la prioridad en la pila de funcionalidades (y no la disponibilidad de los recursos).
- Todos analizamos nuestro trabajo para que añadamos valor a nuestros clientes y a nosotros mismos (eliminar basura).
- Debemos tener una velocidad predecible, una carga de trabajo balanceada y una mínima deuda de calidad.
- Debemos aprender a mejorar, a identificar y subsanar nuestros problemas con las retrospectivas.
- Todos debemos colaborar activamente para innovar.

Los equipos se formaron 4 meses después de la comunicación de la visión. Durante todo este tiempo hubo distintos medios de discusión sobre el cambio y petición de ideas y sugerencias. Además, se hizo un análisis por parte de todas las personas involucradas para encontrar y eliminar "basura"[2] dentro de los equipos y de una manera global. Los análisis se hicieron tanto en reuniones multitudinarias como por e-mail y por la wiki interna.

La estrategia de formación fue completamente abierta dando el poder de decisión a los integrantes de los equipos. Ellos podían elegir a qué equipo unirse, en qué trabajar (ya que los equipos tenían pre-asignados determinadas funcionalidades para los primeros *sprints*) y con quién. Únicamente había que garantizar que los equipos tuvieran las siguientes características una vez formados:

- Entre 3 y 5 ingenieros de software (también con conocimientos de *testing*).
- Entre 2 y 3 ingenieros de calidad (también con conocimientos de desarrollo).
- Al menos una persona debería tener buenos conocimientos de interfaces de usuario.
- Al menos una persona debería tener buenos conocimientos de programación con Win32.
- Al menos una persona debería tener buenos conocimientos de Python.

[2] *muda* en japonés, término muy utilizado en el pensamiento *Lean* [6].

- Personas experimentadas y noveles en un rango de 40/60 aproximadamente.
- Un *Scrum Master* por equipo o una persona interesada en serlo.

Junto con el paso de formación de nuevos equipos, un nuevo equipo fue formado para dar soporte a la transición. Este equipo se encargó principalmente de dar soporte en forma de cursos y *coaching* durante todo el proceso de la transición. Cursos para Scrum Masters, cursos técnicos de Desarrollo Orientado a Pruebas (TDD) [7], de integración continua [8], etc., fueron repetidos en varias ocasiones. Los *coaches* se centraron en dar soporte práctico (*hands-on coaching*) en Scrum, integración continua y test unitarios principalmente así como en *coaching* puro para ayudar a mejorar a todos los equipos.

El siguiente paso fue poner especial énfasis en la integración continua a nivel de proyecto (y no sólo de equipo), así como en la comunicación y colaboración entre los equipos. Los jefes de equipo3 y Scrum Masters fueron especialmente entrenados para manejar situaciones personales y no técnicas. Además, se estableció un "Guardián de código" (un experto en el área) para asegurar que los cambios en componentes críticos se hacían de manera correcta siguiendo un mínimo de calidad.

En la práctica, después de unas cuantas iteraciones, se vio la necesidad de incorporar los siguientes elementos para solucionar diversos problemas surgidos durante la transición:

- Los equipos también son responsables de la entrega (y no sólo de la implementación y pruebas) de la funcionalidad y de todo el trabajo que ello conlleva.
- El equipo de mantenimiento, además de dar soporte a los clientes, es el encargado de desarrollar test automáticos a nivel de sistema y de refactorizar y mejorar el código de las partes legadas del sistema (el resto de los equipos son responsables de la automatización y refactorización del código nuevo y viejo con el que tienen que trabajar).

Muchos de los conceptos tenidos en cuenta durante la transición están basados en el pensamiento *Lean* y *Lean Software Development* [9]. A esta corriente de pensamiento

[3] Definido como la persona responsable de los recursos humanos del equipo

enfocada al desarrollo de software se le atribuyen principalmente siete pilares que están muy presentes a lo largo de toda esta transformación. Éstos son:

1. Eliminar basura. Todo aquello que no trae valor y que nos hace perder tiempo es eliminado durante la transformación.
2. Amplificar el aprendizaje. Todo el mundo debe aprender cosas nuevas para mejorar continuamente.
3. Decidir lo más tarde posible. Si no tenemos toda la información necesaria para tomar una buena decisión y todavía tenemos tiempo, la decisión se pospone hasta el "último momento responsable" para tener más información que nos habilite a tomar una decisión mejor.
4. Entregar lo antes posible. Minimizar el tiempo entre entregas para dar valor al cliente justo cuando lo necesita.
5. Dar poder de decisión al equipo. Los equipos tienen más poder de decisión ya son los que más conocimiento tienen para tomar la decisión adecuada.
6. Incorporar integridad. La transformación toca áreas muy diferentes, todas ellas deben mirar por optimizar el sistema global y no sólo local de manera que todo el sistema sea íntegro.
7. Ver el todo. Los equipos tienen visibilidad de principio a fin, desde el cliente al producto final, no sólo de un componente o de una parte específica del sistema.

5. Resultados

Durante la transformación a equipos por funcionalidades se pasó un cuestionario a todas las personas involucradas en el cambio en dos ocasiones: la primera en enero de 2009 y la segunda en junio del mismo año. Las respuestas a los cuestionarios provinieron de 25 ingenieros de software, 8 ingenieros de calidad y 7 gerentes, jefes de equipo, etc.

Los resultados de la segunda ronda se muestran en las figuras siguientes y se explican con más detalle a continuación. Primero, se presentan los resultados generales para pasar, seguidamente, a los resultados más enfocados en las siguientes áreas: flexibilidad, rendimiento y ambiente de trabajo.

En este artículo no se comentan los resultados de la primera ronda ya que son muy similares a los de la segunda. Estudiando ambos resultados a la vez se observa que en el segundo no hace más que seguir con la tendencia ya marcada en la primero ronda.

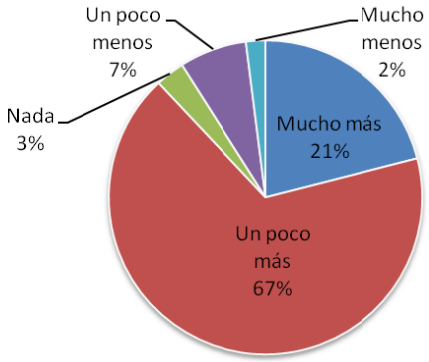

Figura 3. Resultados de la pregunta "Los equipos por funcionalidades mejoran la forma en la que trabajamos"

Como se puede apreciar en la anterior (ver figura 3), a la pregunta general "Los equipos por funcionalidades mejoran la forma en la que trabajamos", un total de 88% de los encuestados opinan que los equipos por funcionalidades mejoran la forma de trabajo.

Este resultado contrasta con el de la siguiente (ver figura 4), ya que en ella se muestra que sólo un 11% no estaría muy satisfecho con la vuelta al método anterior. El 89% restante contestó un "me da igual" o "sin comentarios".

Creemos que estos resultados, que pueden parecer contradictorios en principio, se complementan. La mejora global con equipos de funcionalidades es manifiesta como se verá a continuación al analizar la flexibilidad, el rendimiento y el ambiente de trabajo. Sin embargo, el 89% mostrado en la figura 4, no hace más que reflejar la dificultad en el cambio. Si bien es cierto que hay mejora, también lo es que el cambio es costoso y requiere esfuerzo por parte de todas las partes.

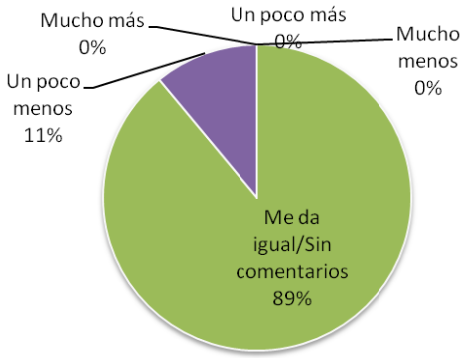

Figura. 4. Resultado de la pregunta "¿Volverías a la antigua forma de trabajo?"

A continuación pasamos a analizar un poco más los resultados por área de interés: flexibilidad, rendimiento y ambiente de trabajo.

5.1. Flexibilidad

La flexibilidad, capacidad para hacer tareas distintas más allá del área de especialización, es la zona dónde la mejoría ha sido más notoria. A las preguntas "¿estás haciendo cosas nuevas?" (ver figura 5) y "¿has aprendido cosas nuevas?" (ver figura 6), un 63% y un 81% respectivamente han respondido que más o mucho más. Solamente un 5% ha respondido "un poco menos" a la primera pregunta siendo el resto un "igual que antes" para ambas preguntas.

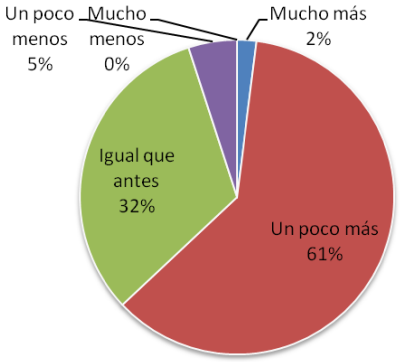

Figura 5. Resultado de la pregunta "¿Estás haciendo cosas nuevas?"

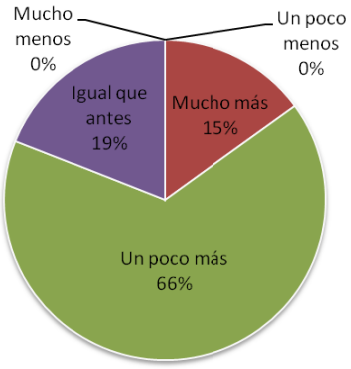

Figura 6. Resultado de la pregunta "¿Has aprendido cosas nuevas?"

Estos resultados nos muestran cómo los equipos por funcionalidades ayudan a compartir conocimiento entre los trabajadores y promueven el aprendizaje de nuevas técnicas, áreas, tecnología, etc., fomentando la flexibilidad en el entorno de trabajo y un mayor rendimiento y productividad a largo plazo, además de evitar los cuellos de botella y aliviar los problemas que ocurrirían si un experto o persona clave no estuviera disponible por un periodo de tiempo.

5.2. Rendimiento

El rendimiento se ha medido con dos preguntas que se centran en los cambios de contexto (rendimiento individual) y en el valor entregado al cliente (rendimiento global).

Como se puede apreciar en la siguiente (ver figura 7), un 41% cree que hay menos cambios de contexto y transferencias de control. Un 40% no ve diferencia y un 19% cree que más.

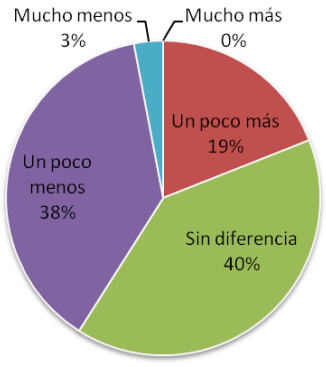

Figura. 7. Resultado de la pregunta "Cantidad de cambios de contexto y transferencias de control"

Si nos movemos al valor entregado al cliente (ver figura 8), los números son más notables que en la figura anterior. Un 64% cree que el valor entregado es mayor o mucho

mayor que anteriormente, un 20% no ve diferencia y sólo un 16% cree que el valor entregado en algo menor.

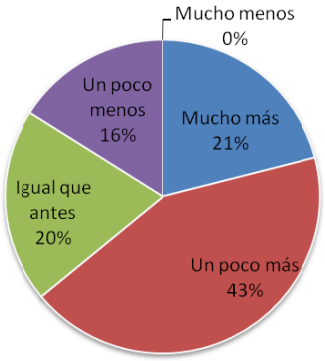

Figura 8. Resultado de la pregunta: "¿Entregamos más valor al cliente?"

5.2. Ambiente de trabajo

El ambiente de trabajo parece mejorar significativamente al trabajar con equipos por funcionalidades. Dos de las preguntas del cuestionario se enfocaban a esta área: "Me gusta el cambio a equipos por funcionalidades" y "¿Ha mejorado tu carga de trabajo?".

A la primera pregunta (ver figura 9), un 69% han respondido que algo más y un 23% que mucho más. Sólo el 2% ha respondido que algo menos quedando el 6% que o bien no tiene comentarios o bien le da igual.

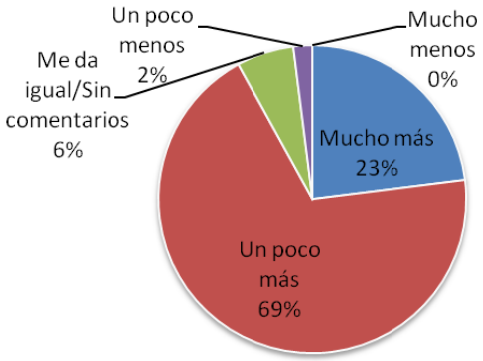

Figura 9. Resultado de la pregunta "Me gusta el cambio a equipos por funcionalidades"

Las respuestas a la segunda pregunta no son tan extremas (ver figura 10) pero también reflejan mejoría. Un total de 37% respondieron que los equipos por

funcionalidades mejoran la carga de trabajo siendo sólo un 16% lo que opinan que, por el contrario, ha empeorado un poco (el resto, 47%, no ve diferencia).

Hay que tener en cuenta que el ambiente de trabajo puede afectar indirectamente al rendimiento [10] y [11]. Esta es la razón por la que creemos que es especialmente importante tener un buen ambiente de trabajo.

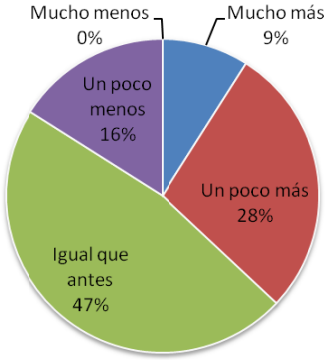

Figura 10. Resultado de la pregunta "¿Ha mejorado tu carga de trabajo?"

6. Conclusiones

Según lo mostrado en los resultados, parece claro que el cambio a equipos por funcionalidades en el caso particular de F-Secure supuso una mejora clara tanto en flexibilidad, rendimiento y ambiente de trabajo.

La transformación a equipos por funcionalidades sacó a la luz la importancia del trabajo en equipo y reveló problemas de algunos individuos hacia esta manera de trabajar no individualista. Además, sirvió para descubrir individuos con un potencial, actitudes y aptitudes realmente buenas que estaban escondidas bajo las limitaciones de los equipos por componentes. Igualmente, el hecho de involucrar a todos los individuos desde el principio de una manera transparente ayudó a descubrir personas que piensan "a lo grande" con ideas innovadoras.

Si vamos un poco más allá de las conclusiones sacadas a través de los resultados y nos movemos a un área un poco menos tangible, el cambio a equipos por funcionalidades nos ha dejado los siguientes puntos que, creemos, son cruciales para una transformación de este estilo:

- El *coaching* y soporte a los equipos resultó fundamental.

- El "guardián de código" es necesario mientras se está en fase la transferencia de conocimiento. Una vez que la transferencia sobre el componente es suficiente, el "guardián" se necesita menos y menos.
- La curva de aprendizaje de nuevas áreas de conocimiento que implicará un rendimiento mayor a largo plazo, traerá un decremento en la productividad a corto plazo.
- El trabajo de mantenimiento se tiene que llevar a cabo en un equipo separado que deberá rotar para facilitar la transferencia de conocimiento en todas las áreas.
- La integración continua, pruebas automáticas y otras prácticas ágiles bien definidas se mostraron indispensables para la transformación, ya que garantizan un enfoque en la calidad, visibilidad rápida sobre defectos y habilitan la flexibilidad y rapidez en los cambios.

Agradecimientos

Nos gustaría agradecer a todos los empleados de F-Secure que han ayudado a escribir este artículo respondiendo al cuestionario (en dos ocasiones) y respondiendo a preguntas sobre la transición y sobre equipos por funcionalidades.

Estamos especialmente agradecidos a los equipos Wincore y a CIS por su ayuda. Y en especial a Janne Järvinen, Miguel Rodríguez y José María Vallet García por sus correcciones e ideas.

Referencias

[1] Cusumano, M. y Selby, W., *Microsoft Secrets*. Touchstone, 1995.

[2] Larman, C. y Vodde, B., *Scaling Agile and Lean Development*. Addison-Wesley Professional, 2008.

[3] Schwaber, K., *Agile Software Development with Scrum*. Microsoft Press, 2004.

[4] Beck, K., *Extreme Programming Explained: Embrace Change (2nd Edition)*, Addison-Wesley Professional, 2004.

[5] Cunningham, W., "The WyCash Portfolio Management System", *OOPSLA '92 Experience Report*, http://c2.com/doc/oopsla92.html, 1 de mayo de 2010

[6] Womack, J. P.; Daniel T. J., and Daniel R., *The Machine That Changed the World: The Story of Lean Production*, Harper Perennial, 1991

[7] Blé Jurado, C., et al., *Diseño Ágil con TDD*, Lulu, 2010.

[8] Duvall, P. M., Matyas, S. y Glover, A., *Continuous Integration: Improving Software Quality and Reducing Risk*. Addison-Wesley Professional, 2007.

[9] Poppendieck M. y Poppendieck, T., *Lean Software Development: An Agile Toolkit*, Addison-Wesley Professional, 2003

[10] Lawler, E.E., Hall, D.T., y Oldham, G.R., "Organizational climate: Relationship to organizational structure, process, and performance", *Organizational Behaviour and Human Performance*, vol. 11, nº 1, pp. 139-155, 1974.

[11] Egan, T. M., Yang, B., Bartlett and Kenneth R., "The Effects of Organizational Learning Culture and Job Satisfaction on Motivation to Transfer Learning and Turnover Intention", *Human Resource Development Quarterly*, vol. 15, nº 3, pp. 279-301, 2004.