Universidad Internacional de La Rioja
Máster universitario en Diseño y Gestión de Proyectos Tecnológicos

Desarrollo de una metodología para la gestión integral de un software altamente cambiante. Caso de estudio: La Farmacia Española

Trabajo Fin de Máster

presentado por: Calvín Tienza, Paloma

Director/a: Nuñez Valdez, Edward

Ciudad: Zaragoza
Fecha: 9 de Septiembre 2014

Resumen

Muchas son las metodologías creadas y diseñadas para la gestión de desarrollos de software pero pocas para gestionar el proceso de creación, mantenimiento y soporte del mismo. Dentro del marco de las prácticas derivadas del Manifiesto Ágil el presente trabajo pretende crear una metodología con procedimientos y buenas prácticas que gestione las diferentes áreas funcionales de soporte a clientes y desarrollo de un producto de software con un alto nivel evolutivo. Tras la identificación de la necesidad de crear este tipo de metodología por la observación de sistemas similares al descrito, se procede a una investigación de la literatura y estudios realizados al respecto para posteriormente crear el marco de trabajo que cumpla los objetivos planteados.

Palabras Clave: metodologías ágiles, servicio a clientes, nivel evolutivo del software

Abstract

There are many methodologies created and designed in order to manage software development processes, but few of them have been created to manage creation, maintenance and customer support processes. This work aims to build a methodology with both procedures and best practices in order to manage different functional areas as customer support and a high changeability level software development in the framework of Agile Manifest practices. After the identification of the need of this kind of methodology because of the observation of similar systems, I do an investigation of the current literature and related researches, and subsequently to create the suit framework to aim the objectives set by this work.

Keywords: agile methodologies, customer support, software evolution level

1. Tabla de contenidos

2. Introducción

Las metodologías ágiles han demostrado una gran eficacia en la aplicación de técnicas de gestión y practicas de desarrollo de un software con requerimientos muy cambiantes. Sin embargo pocas son las aplicaciones que se encuentran de este tipo de prácticas en áreas dedicadas a otro tipo de actividades.

2.1. Metodologías ágiles: ¿Solo para Desarrollo?

Es una realidad que el fenómeno de las metodologías ágiles aplicadas al desarrollo de software ha revolucionado este campo de las tecnologías de la información en los últimos quince año. La causa de su aparición fue generada por la búsqueda de alternativas a las demasiado pesadas y burocráticas metodologías existentes hasta el momento por un grupo de expertos en los modelos de desarrollo y gestión de software (agilemanifesto.org, 2001).

Sin embargo cuando se habla de metodologías ágiles casi siempre se hace referencias solo al área de desarrollo de software. Las últimas tendencias en esta línea de pensamiento es la aplicación de metodologías o procedimientos similares en otras áreas como pueden ser el Testing o sistemas. De aquí surgen movimientos como Testing Agile o DevOps. Se empiezan a conocer incluso movimientos que abogan por aplicar la filosofía del Manifiesto Ágil en los ServiceDesk de los departamentos de IT.

2.2. Creación de un producto Software: más que código

Un negocio basado en la creación, comercialización y mantenimiento de un producto Software no solo depende del desarrollo del Software propiamente dicho.

El equipo de trabajo en este tipo de unidades de producción, bien como un negocio independiente o como un área de negocio dentro de una gran compañía que se dedique a otras actividades, está formado por diferentes áreas funcionales desde el punto de vista tecnológico. Como ejemplo, siendo las más comunes, podríamos citar: Desarrollo, Testing, Helpdesk y Asistencia Técnica en el cliente.

Las metodologías ágiles han demostrado tener un alto grado de adecuación en el desarrollo de productos Software altamente cambiantes. Pero ¿es que acaso solo un buen desarrollo asegura el éxito de un Software? Ni mucho menos. Solo cuando el engranaje entre las unidades funcionales que conforman el equipo es perfecto se consigue un producto sólido y con posibilidades de éxito.

Imaginemos un equipo multifuncional formado por las áreas mencionadas anteriormente: Desarrollo, Testing, Helpdesk y Asistencia Técnica. El equipo de desarrollo aplica metodologías ágiles por lo que ha conseguido, entre otras cosas, más productividad. El número de entregas realizadas es alto, en función de los continuos requerimientos que surgen en los clientes. Sin embargo el resto de áreas no ha evolucionado en cuanto a metodologías y procedimientos aplicados en su gestión y además no existe una interrelación entre las diferentes áreas. En este escenario estos son los principales problemas que nos podemos encontrar:

- Usuarios con un bajo grado de formación sobre el producto: muchos cambios en el producto y no da tiempo a que los clientes los conozcan
- Help Desk saturado por llamadas de consultas de usuario: insatisfacción del cliente por no ser atendido
- Agentes de primera línea y técnicos de calle con bajo índice de resolución debido a la falta de formación: alto índice de escalabilidad de los problemas al equipo desarrollo
- Detección tardía de los problemas y la causa de los mismos debido a un flujo de información débil y tardío entre Help Desk (en contacto con el cliente) y equipo de desarrollo lo que lleva a que las causas de las incidencias producidas en el cliente se resuelvan tarde
- Cuello de botella en el testeo de las entregas: entregas poco testeadas y alto índice de error en producción

Ante estas premisas se considera necesario realizar una metodología integral aplicada a cada una de las áreas antes mencionadas y que consiga crear procedimientos que mejoren tanto las tareas propias de cada una de las unidades funcionales del equipo de trabajo como el flujo de trabajo entre ellas.

Debido a que se parte de la premisa de que el Software que se produce tiene un alto grado de cambio y las metodologías ágiles son adecuadas en este caso, la metodología que se va a desarrollar estará basada en los principios de manifiesto ágil.

2.3. Caso de Estudio: la Farmacia Española

La metodología se va a diseñar basándose en el caso de estudio de un Software desarrollado y comercializado para las oficinas de farmacia españolas.

La farmacia en España está hoy en día en un nivel de informatización del casi el 100%. Es decir, depende totalmente de las nuevas tecnologías para realizar su trabajo diario. La legislación sanitaria, concretamente la parte farmacéutica, está en continua evolución, sobre todo desde el comienzo de la crisis por la necesidad de reducción del gasto farmacéutico. Se puede decir que dentro de las PYMES la farmacia es una de las que mas sometida está a cambios legislativos. La aparición de la receta electrónica en todo el territorio nacional, el copago farmacéutico, la legislación relativa a precios de medicamentos, etc, supone un nivel de dificultad muy elevado para la solución tecnológica implantada en la farmacia.

Por tanto el Software para la gestión integral de una oficina de farmacia está sometido a una continua evolución para satisfacer los innumerables cambios a los que las farmacias se tienen que adaptar.

Una unidad de negocio o entidad empresarial que se dedique a la creación, comercialización y mantenimiento del un Software para la Oficina de Farmacia es un marco perfecto para la aplicación de una metodología basada en los criterios del manifiesto ágil para la gestión integral de las diferentes unidades funcionales participantes.

4. Objetivos

A continuación se detallan cual es el objetivo general y los objetivos específicos del presente trabajo.

4.1. Objetivo General

Una de las principales características de cualquier Software, por su propia naturaleza, es el carácter cambiante del mismo. El nivel de cambios al que se ve sometido un producto Software viene determinado por las necesidades el mercado para el que ha sido diseñado así como por otro tipo de factores como puede ser el avance tecnológico, la aplicación de la legislación en el campo de acción en el que opera el producto u otras razones internas al propio proceso de producción de Software.

En los productos de Software comerciales creados con una determinada aplicación en determinados tipos de actividades y negocios, varios son los agentes implicados en la

vida del producto como puede ser un Help Desk de atención al cliente, técnicos de asistencia en casa del cliente, equipo de Testing y documentación, etc.

Varias han sido las metodologías creadas y aplicadas específicamente en el desarrollo de Software con un alto grado de cambios pero ninguna ha sido creada para desarrollar procedimientos y buenas prácticas que combinen las diferentes áreas implicadas en el desarrollo y mantenimiento del producto.

El objetivo general de este trabajo es, dado un equipo cuya finalidad es crear, comercializar y mantener un producto de Software altamente cambiante, aumentar la productividad de cada una de las áreas funcionales que lo conforman así como generar procedimientos de trabajo que aprovechen las sinergias entre ellas consiguiendo una mejora significativa del producto y del servicio realizado. Para ello se diseñará una metodología basada en los fundamentos del pensamiento Ágil.

La metodología se centrará en el caso de estudio de la creación, evolución y mantenimiento de un Software para la oficina de farmacia. En dicho contexto la metodología estandarizará procedimientos para la interacción entre las áreas de Help Desk, desarrollo y asistencia técnica.

4.2. Objetivos Específicos

Los pasos a seguir para desarrollar la metodología requerida serán:

- Realizar un estudio del arte en tendencias existentes de cara a la aplicación de los principios del manifiesto ágil en actividades diferentes al desarrollo del Software.
- Realizar un planteamiento del estado actual del caso de estudio para identificar los problemas y áreas de mejora.
- Estudiar los resultados y obtener conclusiones
- Implementar la metodología de trabajo para la coordinación de las diferentes áreas funcionales encargadas de la creación, distribución y mantenimiento del Software que solvente los problemas encontrados y refuercen las áreas de mejora.

5. Descripción de la Metodología de trabajo

La metodología que se quiere realizar como objetivo de este trabajo tiene dos factores fundamentales que la caracterizan:

- El producto que se trabaja es un Software con un alto grado de cambios y un porcentaje de evolución muy elevado
- La metodología debe cubrir procedimientos a aplicar a diferentes áreas funcionales como son desarrolladores, Testing, Help Desk y atención técnica en el cliente

Dadas estas premisas el primer paso es realizar un estudio de las corrientes metodológicas existentes actualmente.

Respecto al primer punto, cualquier estudio de procedimientos de gestión y desarrollo de proyectos de Software con un alto nivel de cambios se encuentra inequívocamente relacionado con las metodologías Ágiles. Se realizará un amplio estudio del origen de estas metodologías, la relación con el pensamiento Lean, así como descripción de algunos de los marcos de trabajo o técnicas ágiles (Scrum, Kanban, etc).

En relación al segundo punto el estudio se orienta en dos sentidos. Primeramente se realiza la búsqueda de metodologías que relacionen diferentes áreas funcionales involucradas en el campo de Sistemas de Información como pueden ser Service Desk, departamentos de operaciones, desarrolladores, etc. En segundo lugar el estudio es orientado en la búsqueda de información acerca de metodologías ágiles aplicadas en áreas que no son de desarrollo, buscando sobre todo literatura acerca de técnicas consideradas “ágiles” aplicadas a equipos de Help Desk.

Después de la búsqueda del marco teórico relacionado con los objetivos de la metodología que se va a realizar se llevará a cabo un estudio del caso de estudio en el que se va a fundamentar el trabajo. Mediante la observación y estudio de los procedimientos realizados actualmente se obtendrán los problemas existentes y las áreas de mejora del sistema en su totalidad.

Una vez concluidos cuales son los problemas se reflexionará sobre cuales pueden ser los procedimientos y buenas prácticas a realizar en los diferentes equipos de trabajo de cara a mejorar cada una de las áreas así como procesos que aprovechen las sinergias existentes entre ellos y mejoren tanto el producto de Software como el servicio prestado a los clientes.

6. Contexto y Estado del Arte

Dos son los principales temas hacia los que se ha dirigido la investigación llevada a cabo para realizar el presente trabajo: la evolución del software y las metodologías ágiles.

Inicialmente se profundizará en la naturaleza cambiante de cualquier software que se encuentra activo en el mundo real. Una de las premisas de este trabajo es que la metodología objetivo del mismo se aplicará en sistemas software con un alto nivel evolutivo.

Posteriormente el estudio se enfocará hacía las metodologías ágiles como marco bajo el que se quiere construir el glosario de buenas prácticas y procedimientos. Se revisarán el origen, las influencias y diferentes técnicas aplicadas en el llamado desarrollo ágil.

En el marco de las metodologías ágiles se realizará la búsqueda de información acerca de la aplicación de técnicas de está índole en áreas diferentes a la de desarrollo como puede ser un Help Desk o área de Testing.

Por último se investigará que estudios se están realizando de cara a la aplicación de procedimientos que favorezcan el flujo de información y la unificación de objetivos de las diferentes áreas componentes del proceso de creación y mantenimiento del software.

6.1. Evolución del Software

Uno de los principales aspectos que se ha tenido en cuenta para la elaboración del presente trabajo es que la metodología diseñada será destinada principalmente a la gestión integral de la producción y mantenimiento de un Software altamente cambiante.

La evolución del Software es una de las partes estudiadas por la Ingeniería del Software (Ian Sommerville, 2005). Una vez que un sistema es desarrollado inevitablemente debe sufrir modificaciones de lo contrario dejará de ser útil para el objeto para el que fue creado. Los motivos por los que un Software tiene que evolucionar son de diferente índole, puede ser desde requerimientos del sistema en el que reside, modificaciones para subsanar errores del propio Software, o necesidades funcionales que aparece a lo largo de la vida del sistema.

Las empresas dedican mucha parte del presupuesto destinado al Software a las labores de mantenimiento del Software ya existente. Como se hace referencia anteriormente la causa de los cambios que se han de realizar en un Software puede ser de diferente naturaleza

pero la mayoría de las veces son generados por necesidades y nuevos requerimientos del contexto o usuarios que utilizan el Software.

Uno de los enfoques del modelo de vida de Software es el modelo en espiral según podemos ver en la figura 1.

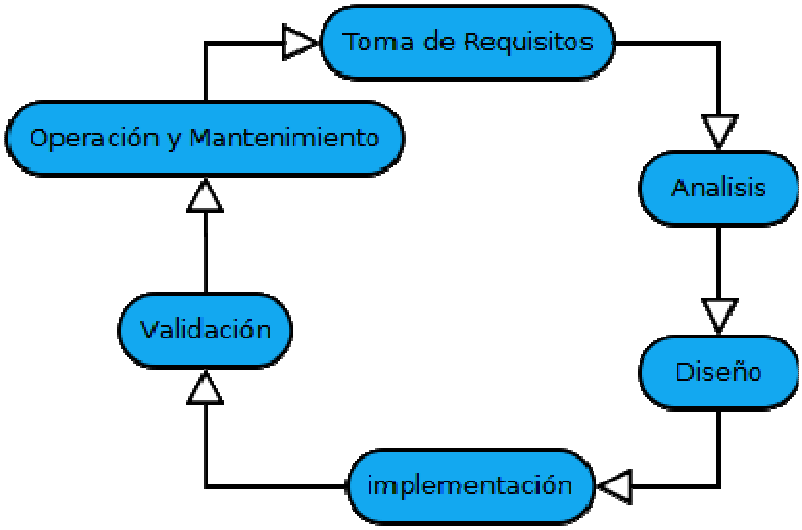

Figura 1 – Modelo de vida del desarrollo del software (Propia)

Llamaremos fase de operación a la fase cuando un Software ha sido entregado al cliente y se hacen las pruebas de aceptación por parte de los usuarios. Cuando el cliente da por finalizada la fase de aceptación ya se comienza lo que se llama el mantenimiento del producto. Según el Instituto de Ingenieros Eléctricos y Electrónicos (IEEE) se define como mantenimiento del Software “la modificación de un producto de software después de la creación para corregir fallos, mejorar el rendimiento u otros atributos o para adaptar el producto a un entorno modificado”

Existen varios tipos de mantenimientos (Sommerville, 2005):

- Mantenimiento correctivo: se produce cuando hay que localizar y eliminar un defecto detectado en el sistema. Un defecto es una característica del sistema que puede derivar en un fallo.
- Mantenimiento adaptativo: se debe a cambios en el contexto en el que se ejecuta el Software. Puede deberse a cambios del sistema operativos o cambios en el entorno de desarrollo por ejemplo.
- Mantenimiento perfectivo: mantenimiento producido por la implementación de nuevas funcionalidades que requiere el usuario.

- Mantenimiento preventivo: se trata de un mantenimiento destinado a mejorar la calidad del Software sin modificar las funcionalidades del mismo.

6.2. Las leyes de Lehman

Existe un estudio importante realizado en torno a la dinámica de la evolución de los programas que fue realizado por Lehman y Belady inicialmente en los años 70 y que posteriormente fue retomado por los mismos autores en los 80. Posteriormente, en los años 90 Lehman repitió un estudio demostrando que las mismas teorías que se estudiaron en 20 años antes se seguían cumpliendo a pesar de la evolución de la Ingeniería del Software además de otros estudios que se realizaron demostrando la importancia de la realimentación en los procesos de evolución.

De estos estudios se crearon las llamadas Leyes de Lehman o leyes de la evolución del Software. Según Lehman los sistemas de Software se dividían en tres tipos que son los que pueden especificarse totalmente que se definen como sistemas estáticos, los que no pueden ser especificados a priori y necesitan de un proceso iterativo para poder estar totalmente definidos y los sistemas que ya funcionan en el mundo real. El estudio se centró sobre todo en los sistemas del tercer tipo, sistemas altamente cambiantes y en continua evolución (Garzás, 2014).

Las 8 leyes de Lehman son las siguientes (Sommerville, 2005):

- **Cambio continuado**: un programa que se usa en un entorno real necesariamente debe cambiar o se volverá progresivamente menos útil en ese entorno.
- **Complejidad creciente:** A medida que un programa en evolución cambia, su estructura tiende a ser cada vez más compleja. Se deben dedicar recursos extras para preservar y simplificar la estructura.
- **Evolución prolongada del sistema:** La evolución de los programas en un proceso autorregulativo. Los atributos de los sistemas, tales como tamaño, tiempo entre entregas y el número de errores documentados, son aproximadamente invariantes para cada entrega del sistema.
- **Estabilidad organizacional:** Durante el tiempo de vida de un programa, su velocidad de desarrollo es aproximadamente constante e independiente de los recursos dedicados al desarrollo del sistema.

- **Conservación de la familiaridad:** Durante el tiempo de vida de un sistema, el cambio incremental en cada entrega es aproximadamente constante.
- **Crecimiento continuado:** La funcionalidad ofrecida por los sistemas tiene que crecer continuamente para mantener la satisfacción de los usuarios.
- **Decremento de la calidad:** La calidad de los sistemas comenzará a disminuir a menos que dichos sistemas se adapten a los cambios en su entorno de funcionamiento.
- **Realimentación del sistema:** Los procesos de evolución incorporan sistemas de realimentación multiagente y multibucle y estos debe ser tratados como sistemas de realimentación para lograr una mejora significativa del producto.

Una vez estudiadas las diferentes leyes de Lehman se consideran muy interesante y totalmente relevantes para el objeto del presente trabajo la ley de la Conservación de la familiaridad y la Retroalimentación del sistema.

Según la ley de la conservación de la familiaridad cuanto mayor sea la cantidad de mejoras que se incorpore en una versión del Software más probable será que los implicados en la utilización de dicho Software desconozcan todas las modificaciones realizadas y se perderá el conocimiento sobre el sistema. Por tanto, para mantener la familiaridad, las entregas deben ser invariantes en cuanto a tamaño no haciendo entregas demasiado grandes en funcionalidad. Esto no siempre se consigue provocando la consecuencia negativa de desconocimiento por parte del usuario y otros agentes implicados. Según veremos en el caso de estudio que se presentará mas adelante este es uno de los problemas detectados y que ha de ser subsanado con la metodología a diseñar.

Respecto a la ley de Realimentación del Sistema lo que se deja de manifiesto es que el proceso de evolución del sistema que traba en un entorno real es un proceso continuo de realimentación donde están recibiendo constantemente entradas de diferentes agentes implicados como puede ser grado de satisfacción del usuario, sugerencias de nuevas funcionalidades a realizar, número de defectos detectados en la presente versión del producto, etc. Lo que dice esta octava ley de Lehman es que toda esta información obtenida debe plantear nuevas acciones a realizar e incorporar en la siguiente iteración o evolución del Software. Esta ley también está directamente relacionada con la metodología a realizar ya que se quieren poner en práctica procedimientos que recojan las entradas que lleguen de

los diferentes agentes que intervienen en el proceso productivo del Software como son agentes de Help Desk, Testing y de asistencia técnica.

6.3. Metodologías Ágiles

Las últimas dos décadas han sido de gran actividad en cuanto a movimiento en las tendencias de desarrollo de Software se refiere. Ha mediados de los años 90 comenzaron a hacer su aparición métodos de desarrollo y de gestión de Software que querían romper con las metodologías llamadas “pesadas” que obligaban a invertir muchos esfuerzos en procesos burocráticos de documentación y que fracasaban habitualmente en las entregas de productos que diferían mucho de lo que el cliente o destinatario del producto necesitaba.

En 2001 un grupo de diecisiete expertos en el desarrollo del Software que se habían mostrado críticos de los modelos existentes en el momento se reúnen en Snowbird con la finalidad de reflexionar y estudiar procedimientos y técnicas para trabajar en el desarrollo de Software. Estos procedimientos fueron acuñados con el nombre de métodos ágiles en contraprestación de los métodos tradicionales que se consideraban pesados y lentos (Wikipedia, 2014).

De esta reunión surgió lo que se ha llamado el manifiesto ágil en el que se resumen doce principios que se han de mantener inherentes a cualquier práctica que se quiera enmarcar en el contexto de estas metodologías ágiles.

Primeramente se definieron los Valores del Manifiesto Ágil (agilemanifesto.org, 2001):

- Individuos e interacciones sobre procesos y herramientas
- Software funcionando sobre documentación exhaustiva
- Colaboración con el cliente sobre negociación contractual
- Respuesta antes el cambio sobre seguir un plan

En base a estos valores se crean los doce principios del manifiesto Ágil (agilemanifsto.org, 2001):

- Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.
- Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos Ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente.

- Entregamos software funcional frecuentemente, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible.
- Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.
- 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.
- El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara.
- El software funcionando es la medida principal de progreso.
- Los procesos Ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios debemos ser capaces de mantener un ritmo constante de forma indefinida.
- La atención continua a la excelencia técnica y al buen diseño mejora la Agilidad.
- La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.
- Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados.
- A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para a continuación ajustar y perfeccionar su comportamiento en consecuencia.

6.4. Lean y Desarrollo Ágil

Lean es un sistema de producción que se adoptó en Toyota en los años cuarenta, de la mano de Taiichi Ohno, un directivo de la empresa, después de observar antes y después de la segunda guerra mundial la forma de trabajar en Estados Unidos (Garzás, 2014)

Dentro del pensamiento Lean aparece el término “muda” que se define como un despilfarro, es decir, algo que absorbe recursos pero no crea valor (Martin, 2008). El pensamiento Lean lo que persigue generar procedimientos que generen valor con menos recursos (humanos, técnicos, tiempo, etc) pero que a su vez se acerque todo lo posible a lo que los clientes demandan. Otra base del pensamiento Lean es realizar un continuo control de la calidad buscando lo que se viene a llamar principio de cero defectos.

El pensamiento Lean se construye sobre siete principios Unidos (Garzas, 2012):

- Eliminar desperdicios
- Amplificar el aprendizaje
- Decidir lo mas tarde posible

- Entregar lo mas rápido posible
- Capacitar y potenciar el equipo
- Construir con integridad
- Ver el todo

Muchos son los que han relacionado el pensamiento Lean con la ingeniería del software y mas concretamente con el desarrollo ágil. Debe ser mencionado el estudio realizado por Mary y Tom Poppendieck en su libro Lean Software Development. AnAgile Toolkit (Poppendieck, 2003). En este libro los autores desarrollan una metodología ágil basándose en el pensamiento Lean de Toyota. Hoy en día la industria del software se asimila a una fábrica que ha de producir rápidamente y sin defectos, como si se tratara de fábricas humanas (C. Martin, 2008) y de ahí que metodologías de gestión como SCRUM estén inspiradas en las cadenas de producción automovilística.

En los estudios realizados por los hermanos Poppendieck se redefinen los principios de Lean aplicándolos al desarrollo del software (Poppendieck, 2003):

- Eliminar el desperdicio
- Calidad integrada, es decir, desde el primer momento se desarrolla con calidad y se han de llevar a cabo acciones correctivas lo antes posible
- Crear conocimiento: se ha de evitar el adquirir todo el conocimiento necesario para la creación de un software desde el primer momento. Será un proceso evolutivo donde el conocimiento se irá ampliando a medida que vaya creciendo el producto desarrollado.
- Aplazar las decisiones
- Entregar tan rápido como sea posible
- Respetar a las personas
- Optimizar el conjunto

Cabe destacar que mientras las metodologías ágiles están orientadas al desarrollo del software el pensamiento Lean puede ser aplicado a todas las áreas desde el desarrollo a la empresa en la que se produce.

Otro aspecto a tener en cuenta en el que inciden tanto el pensamiento Lean como las metodologías ágiles es la importancia de las personas en el proceso productivo.

Se han de crear equipos de trabajo que sean colaborativos y tiendan a alcanzar objetivos comunes. El realizar un software conlleva que continuamente se tengan que tomar decisiones que seguramente no solo afectan a quien las está tomando sino al resto de personas del equipo implicadas en el desarrollo (Womack, Jones, 2005).

6.5. "Mudas" en el desarrollo del software

Respecto a las mudas podríamos decir que en el ámbito del desarrollo del software se contemplan las siguientes (Poppendieck, 2009):

- Trabajo sin terminar: un producto sin entregar, es decir, sin llegar a ponerse en producción es una fuente de desperdicio total.
- Reaprendizaje; se trata de acciones que provocan que se tenga que rehacer trabajo. Por ejemplo una documentación excesivamente débil del código hace que se pierda tiempo en la modificación posterior del mismo.
- Desarrollo extra que no estaba contemplado en las especificaciones
- Cambios de asignación de las tareas

6.6. Kaizen: mejora continua

El concepto de Kaizen o mejora continua resulta muy interesante de cara a la realización de la metodología objeto de este trabajo ya que se fundamenta en que la mejora continua del producto se produce a través de las personas y los procesos, creando acciones de mejora por parte de los miembros del equipo en el proceso de producción.

La producción Lean se basa en la aplicación de Kaizen, es decir, en la mejora continua. Kaizen implica la realización de pequeñas mejoras que vayan incrementando poco a poco el valor del producto consiguiendo a lo largo del tiempo mejoras muy sustanciales.

Kaizen o la mejora continua ha sido también aplicada al desarrollo del software. (Huda, Presto, 1992). En el artículo de Preston y Huda se determina que la capacidad de mejora se realiza a través de los procesos y las personas. Se incide en la importancia del factor motivacional para que las personas realicen correctamente su trabajo y estas deben estar capacitadas e involucradas totalmente en las organizaciones.

Los eventos Kaizen son eventos que se realizan en una organización para solucionar un problema detectado. Al evento acuden personas de diferentes áreas que están relacionadas con el tema a tratar. Durante el tiempo que dura el evento el grupo debe estudiar el problema, eliminar los puntos del proceso a estudiar donde se generan desperdicios y replantear el proceso para mejorarlo y solucionar el problema.

6.7. SCRUM

Según la guía de SCRUM editada por scrum.org SCRUM es un marco de trabajo para de desarrollo y mantenimiento productos complejos (traducción realizada en 2013).

Más específicamente podemos decir que SCRUM consiste en una serie de prácticas y roles destinados a gestionar principalmente desarrollos de software sometidos a requisitos inestables y que se necesita una rápida reacción ante los imprevistos que se van produciendo a lo largo del desarrollo y mantenimiento.

SCRUM tiene su origen en el estudio que realizaron Ikurijo Nonaka e Hirotaka Takeuchi de los procesos productivos de varias empresas de manufactura tecnológica como son Fuji-Xeros, Hewlett-Packard y 3M entre otras (Wikipedia, 2010). El término SCRUM proviene de la forma en la que se agrupan los jugadores de rugby en una determinada jugada llamada melé y cuyo término anglosajón es Scrum en la cual los jugadores se agrupan entre sí.

Mas adelante, en 1995 Ken Schwaber y Jeff Sutherkand realizaron la primea versión del proceso de desarrollo de SCRUM en la OOPSLA 95. La guía de SCRUM editada por ambos autores y revisada por última vez en 2011 define lo que es SCRUM y recopila los roles, eventos y artefactos de lo componen (Wikipedia, 2010).

SCRUM se basa en el control de procesos basado en la experiencia. Propone un enfoque iterativo e incremental de manera que el desarrollo no se produce de una vez sino que se van realizando pequeñas iteraciones. En cada una de las iteraciones el producto va evolucionando y se van añadiendo nuevos requisitos. Se han de priorizar los requisitos que más valor dan al producto de cara al cliente.

En la imagen podemos ver un diagrama en el que se resumen los grandes pilares de SCRUM:

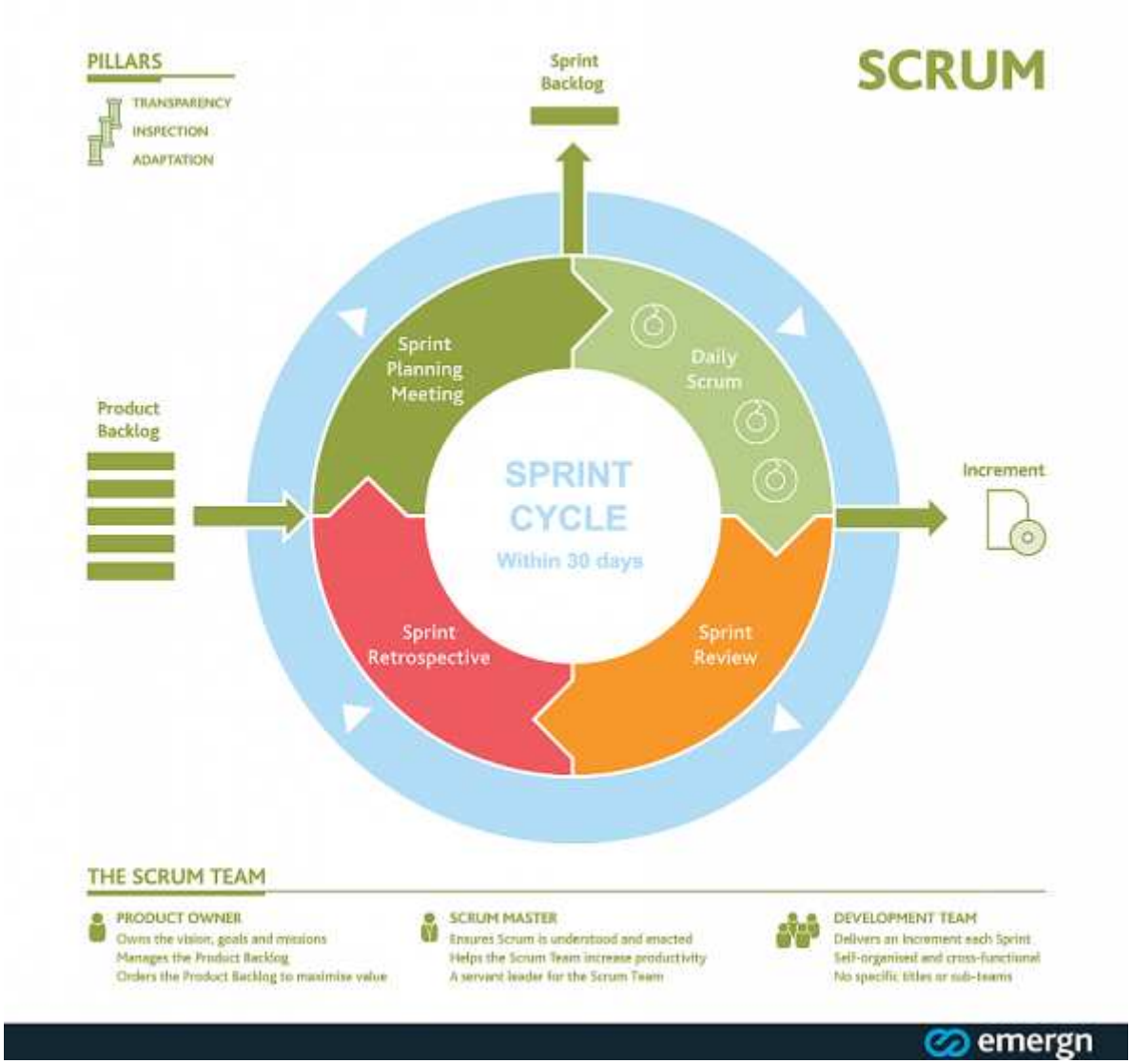

Figura 2: Diagrama de SCRUM (https://www.valueflowquality.com/)

6.7.1. Glosario de SCRUM

Estos son algunos de los conceptos que nos encontramos en el marco de SCRUM:

- **SCRUM team**: está formado por el Dueño de Producto, el Equipo de Desarrollo y el SCRUM Master. Son equipos autoorganizados y multidisciplinares de manera que no necesitan a nadie fuera del equipo para la entrega del producto en cada iteración al cliente con los requisitos planificados (Scrum.org, 2013)

- **Dueño del Producto**: el quien controla la pila de producto y el que prioriza y decide que es lo que se debe de hacer. Debe ser una única persona y toda la organización debe respetarle en sus decisiones en cuanto a las historias que se van a realizar en cada SPRINT (Scrum.org, 2013)

- **SCRUM Master:** se encarga de asegurar que se aplique SCRUM correctamente, que se siguen las indicaciones teóricas. Debe estar al servicio de los miembros del equipo como facilitador para cualquier tema que pueda surgir. De igual manera ayudará a las personas ajenas al equipo a realizar las interacciones que fueran necesarias con el equipo (Scrum.org, 2013).

- **Pila de producto:** se trata de una lista de todo lo que debe ser desarrollado. Son los requisitos que se deben implementar. No es una lista estática. A medida que se van realizando las iteraciones la lista va cambiando y evolucionando (Scrum.org, 2013).

- **Pila de Sprint:** se trata de las tareas que se van a realizar en el Sprint que han sido de la pila de producto. Es seleccionada por el equipo de desarrollo y durante la ejecución del Sprint el equipo de desarrollo va generando nuevas entradas en la pila de Sprint cuando se genera trabajo nuevo o bien se eliminan entradas que ya no son necesarias y se realizan estimaciones del trabajo pendiente (Scrum.org, 2013).

- **Definición de Terminado:** el equipo de SCRUM decidirá que es lo que se entiende por un trabajo terminado. El término terminado o “done” variará pudiendo ser una tarea que ha sido desarrollada pero no probada, una tarea que ya ha pasado la fase de pruebas o una tarea que ya ha sido terminada, probada e instalada, etc.

- **Reunión diaria o Daily Scrum:** se trata de un evento en que el diariamente el equipo de desarrollo y el Scrum Master se reúnen durante unos quince minutos para tratar cual es el curso del Sprint. Cada miembro del equipo debe informar de que hizo el día anterior, que va a realizar en las próximas 24 horas y si hay algún problema que haya que tratar. Debe ser una reunión dirigida por el equipo y el Scrum Master se debe responsabilizar de que no se extienda en el tiempo y que se realiza a diario (Scrum.org, 2013).

- **Reunión de revisión de Sprint (Sprint Review)**: es una reunión que se realiza al finalizar el Sprint entre el equipo de desarrollo, el Scrum Master y e Dueño del producto. Se trata como ha ido el Sprint, cual es el incremento realizado, se revisa la pila de producto y se prepara la información que servirá de entrada para la reunión de planificación de Sprint (Scrum.org, 2013).

- **Reunión de Retrospectiva (Sprint Retrospective):** se trata de una reunión donde el equipo se estudia a si mismo y se determinan acciones de mejora de cara a aplicar en futuros Sprints.

6.8. Agilidad no solo para desarrollar software

El éxito que han tenido en las últimas décadas en el mundo del desarrollo software las metodologías ágiles ha llevado a expertos en otras materias de IT e incluso de otras áreas a acercarse al mundo ágil y abrir nueva posibilidades en cuanto a metodologías de trabajo y gestión de procesos. Se ha de tener en cuenta que las propias metodologías ágiles tienen su origen en actividades tan diferentes de los sistemas de información como es la industria automovilística.

6.8.1. Testing Ágil

Testing es la primera actividad diferente del desarrollo de software a la que también se le han aplicado técnicas enmarcadas dentro de la filosofía de trabajo que podríamos llamar Ágil.

El desarrollo ágil incorpora el Testing dentro del mismo proceso de desarrollo. Las pruebas deben de ser desarrolladas dentro del mismo equipo de desarrollo y a lo largo de las diferentes iteraciones realizadas. Puede ser realizado por los mismos desarrolladores o añadiendo testers especializados

Un tester ágil se encuentra cómodo con el cambio, colabora con programadores y gente del negocio y entiende perfectamente el concepto de que las propias pruebas se pueden utilizar como documentación de los requerimientos y para dirigir el desarrollo (Crispin, Gregory 2009).

La calidad y las pruebas están intrínsecas en un equipo ágil, por lo que los propios desarrolladores realizarán pruebas con técnicas como TDD. Por lo tanto el tester ágil debe ir más allá. Entre las técnicas a realizar por un tester ágil pueden ser la automatización de pruebas, participación en el diseño junto con los desarrolladores, interlocutor con el cliente para asegurarse del cumplimiento de los requisitos, etc.

6.8.2. Agilidad en Service Desk

Entendemos con Service Desk el servicio de soporte a usuarios que se realizan desde departamentos de IT u otro tipo de entidades o negocios en los que el producto o servicio ofrecido es un producto tecnológico. Los usuarios solicitan ayuda a un Service Desk o Help Desk debido a problemas, consultas varias, instalación de nuevos dispositivos, etc.

Existen varios estudios que contemplan la posibilidad de aplicar metodologías ágiles en un servicio de atención al cliente o Service Desk.

Noel Bruton, un experto en la gestión de Service Desk de departamentos de IT comenta como las técnicas utilizadas en servicios del tipo Service Desk para soporte a usuarios se han quedado estancados desde que en 2002 apareció ITIL. Mientras que otras metodologías como las ágiles han revolucionado el desarrollo de software en los últimos años, en lo que es servicios no se ha avanzado. (Bruton, 2013).

Existen ya tendencias que plantean la aplicación de Kanban a equipos de Help Desk (Weinstock, Herman 2009). Se plantea la posibilidad de aplicar Kanban para realizar todas aquellas tareas que puedan ser planificadas. El concepto de Work In Process es muy importante de cara a evitar cuellos de botella y asegurarse de que el trabajo se termina antes de comenzar otras tareas.

7. Implementación

En este apartado se llevará a cabo el diseño de la metodología objeto del presente trabajo.

Primeramente se presentarán los requisitos que ha de cubrir la metodología. Para la identificación de estos requisitos se realizar un estudio detallado de los diferentes componentes que conforman el marco de trabajo en el que la metodología se va a aplicar y dichos componentes se identificarán en el caso de estudio seleccionado.

Se realiza la descripción detallada del caso de estudio en cuanto a roles implicados, procesos, procedimientos, etc. A partir de la presentación del marco actual de trabajo del caso de estudio se deducirán los requisitos que ha de cumplir la nueva metodología siendo estos directamente deducidos de los problemas actuales y las áreas de mejora que se identifiquen en el estudio del caso.

Una vez identificados los requisitos se procederá a revisar toda la literatura relacionada de cara a obtener un fundamento teórico en el cual poder enmarcar la metodología generada.

7.1. Caso de estudio: Software para una oficina de farmacia

7.1.1. Presentación del caso

El caso de estudio elegido es el proceso integral de creación, distribución y mantenimiento de un software destinado a la oficina de farmacia.

El caso se enmarca en un área de negocio perteneciente a una compañía cuya principal actividad no es la producción de software sino la distribución de productos farmacéuticos.

Como servicio adicional a las oficinas de farmacia dicha compañía ofrece la informatización integral de la farmacia. Este servicio no cubre solamente la instalación de un software sino que es una solución tecnológica integral que consiste en etiquetas electrónicas para los productos, robotización, cámaras de seguridad, etc. El software objetivo de este estudio interacciona con todos estos elementos tecnológicos y además cubre los múltiples requisitos funcionales que requiere este tipo de negocio.

7.1.2. Nivel de evolución del software de farmacia

La farmacia se ve sometida a un proceso de adaptación continuo a nuevas necesidades originadas por varias causas. Es por esto que el software de la farmacia tiene un nivel de evolución superior al que ya de por si tiene cualquier sistema que actúe en un entorno real.

Estas son las causas principales del carácter cambiante de este tipo de software:

7.1.2.1. Impacto de la legislación

El sistema de seguridad social en España contempla, entre otras áreas, el seguimiento de la prescripción y dispensación de medicamentos como medidas para controlar el gasto farmacéutico a cargo del estado. La farmacia es un agente fundamental en esta cadena de procesos formada por la fabricación del medicamentos y productos farmacéuticos por parte de los laboratorios, la distribución de los mismos a las farmacias por las distribuidoras farmacéuticas, la prescripción por los facultativos en los centros de salud, hospitales, etc, y la dispensación a los pacientes en las oficinas de farmacia.

El estado español teniendo como objetivos principales la reducción del gasto farmacéutico, el control del consumo de determinados fármacos o garantizar la cobertura sanitaria a todos los españoles ha llevado y lleva a cabo continuamente medidas que imponen mediante leyes y órdenes ministeriales a los agentes de la cadena sanitaria. Dos de estas medidas que más han impactado en la farmacia española en la última década son la Receta Electrónica y el Copago Farmacéutico.

Receta Electrónica

En el concepto Receta Electrónica se engloba la informatización total de todos los procesos que intervienen en la prescripción de un medicamento o producto farmacéutico por el facultativo y la posterior dispensación del mismo en la oficina de farmacia. Dicha informatización se lleva a cabo mediante flujos de información entre los diferentes agentes. La comunicación entre las farmacias, los centros de prescripción (centros de salud y hospitales), las consejerías de sanidad y los colegios de farmacéuticos se realizan mediante

la conexión de varias redes de comunicación seguras que aseguran la confidencialidad de los datos intercambiados.

El escenario de asignación de competencias a las diferentes comunidades autónomas en el estado para sanidad, educación, justicia, etc, hace que la receta electrónica en España haya sido implementada como mínimo de 17 formas diferentes, una por cada comunidad autónoma, y en algunos casos extremos, como el de la Comunidad Valenciana uno por provincia, teniendo modelos diferentes Valencia, Castellón y Alicante.

El software de la oficina de farmacia ha de ser capaz de interaccionar con los sistemas de receta electrónica para llevar a cabo el proceso de la prescripción y otros servicios asociados como la consulta de dispensaciones realizadas a un cliente, consulta de tratamientos y dispensaciones futuras, facturación de las recetas dispensada, etc.

Un software de oficina de farmacia que tenga presencia en todo el territorio nacional tiene que desarrollar al menos 17 sistemas diferentes de dispensación electrónica. Esto incrementa considerablemente la complejidad del mismo.

Hay que tener en cuenta que los sistemas de Receta Electrónica están en continua evolución. El proceso habitual es que una comunidad comience la dispensación electrónica con unos servicios básicos que cubran los procesos básicos de la dispensación. Cuando el nivel de cobertura del sistema en la comunidad es total se realizan periódicamente una serie de evolutivos que van mejorando e incrementado la funcionalidad del sistema.

En conclusión, la Receta Electrónica en España es un sistema en continua evolución, en cada uno de los 17 modelos diferentes, lo que obliga al software de la farmacia a estar en constante adaptación y ampliación de la funcionalidad.

Copago Farmacéutico

Otro de los impactos que más ha influido últimamente en el mundo de la farmacia ha sido la puesta en marcha del copago farmacéutico el 1 de Julio de 2012. Mediante el copago farmacéutico la aportación que el paciente debe realizar en la farmacia de los medicamentos depende de su renta, con la particularidad de que los individuos que reciben una pensión del estado deben pagar un importe máximo al mes pasando a no pagar nada cuando alcanzan dicho importe. Este cambio, sobre todo la gestión del tope mensual, significó un cambio sustancial en la dispensación de los medicamentos. Los organismos sanitarios tenían que realizar un control de la dispensación mensual realizada a los pacientes para controlar el importe que éstos debían aportar. Ese control solo se puede hacer a través de la farmacia y

por supuesto de forma informatizada. Comenzaron a surgir sistemas que en paralelo a la receta electrónica hacen que las recetas convencionales, es decir, en papel, entren en el circuito y queden registradas en el sistema para así tener una gestión total de todas las recetas dispensadas.

Un ejemplo de esos sistemas es el llamado SISCATA de la Comunidad de Madrid. Las recetas de papel en esta comunidad tienen que ser registradas en SISCATA. En una primera petición a este servicio la farmacia, informando de los datos del paciente y el identificativo de la receta, obtiene información acerca del nivel de cobertura del paciente así como si ha alcanzado o no el límite mensual en caso de ser un pensionista. De esta manera la farmacia sabe que importe debe aportar el paciente por los medicamentos. Posteriormente la farmacia deberá informar a SISCATA de los medicamentos que retira el paciente y del importe pagado, actualizando de esta manera su gasto mensual y quedando registrados los datos para la posterior facturación a la farmacia.

Sistemas paralelos a SISCATA han sido implementados en Comunidad Valenciana, Cataluña, Galicia, País Vasco y otras comunidades, cada uno de ellos con sus propias arquitecturas y modelos de datos.

7.1.2.2. Necesidades del mercado

Otra causa de evolución del software de la farmacia la encontramos en las necesidades originadas por el propio mercado. Factores como la crisis económica y la reducción del precio de los medicamentos, el incremento del ratio de farmacias por habitante en algunas comunidades autónomas como Navarra, la creación de las llamadas parafarmacias, etc han provocado que la farmacia haya tenido que cambiar el modelo de negocio en algunas zonas teniendo que aplicar técnicas de merchandising y marketing para mantener la cuota de mercado.

Por las razones expuestas anteriormente los proveedores de software para la farmacia han tenido que realizar desarrollos para técnicas de merchandising como tarjetas de fidelización con sistemas de puntos y bonificaciones. En el mercado han surgido muchos y variados de estos sistemas de fidelización. Son sistemas que suelen agrupar a varias farmacias formando una red de fidelización y haciendo que los clientes que compran en farmacias del grupo obtengan beneficios en forma de puntos canjeables por dinero, regalos, compras de productos, etc. Estos sistemas suelen residir en un hosting proporcionado por empresas que comercializan el servicio a las farmacias. El software de la farmacia debe ser capaz de interactuar con cada uno de esos sistemas para informar de las ventas realizadas a clientes mediante las tarjetas de fidelización.

También se ha incrementado considerablemente la robotización de las farmacias, ya que mediante la utilización de estos sistemas el personal de la farmacia está más en contacto con el cliente y se fomenta la venta cruzada. El software de gestión de la farmacia se comunica con el robot para obtener los productos dispensados, para cargar el robot y otros procesos relacionados.

Las etiquetas electrónicas informando del precio y otros datos de los productos también han tenido un papel importante en el desarrollo del software de farmacia en los últimos años. La aparición de la necesidad de este tipo de tarjetas electrónicas se incrementó con la obligatoriedad que tienen las farmacias, como otros negocios de venta al por menor, de informar del precio por unidad de medida de los productos de parafarmacia. Mediante la utilización de tarjetas electrónicas la farmacia no tiene que mantener manualmente esta información en las exposiciones de los productos.

7.1.2.3. Nuevas tecnologías: eCommerce

El comercio por Internet está cada vez más extendido. La farmacia no se ha manteniendo al margen en la venta de parafarmacia. Incluso parece que está cerca la posibilidad de dispensar ciertos medicamentos mediante eCommerce.

Los programas de gestión de farmacia son la fuente que alimentan las bases de datos de estas aplicaciones de eCommerce informando de los precios de los productos y existencias, por ejemplo. La comunicación es bidireccional de manera que el software de la farmacia obtiene la información de ventas realizadas en la aplicación de eCommerce.

7.1.1. Organización de la unidad de negocio

La creación, despliegue y mantenimiento del software de farmacia objeto de este caso de estudio es llevado a cabo por una unidad de negocio perteneciente a una compañía internacional dedicada principalmente a la distribución farmacéutica.

Esta unidad está formada por tres departamentos:

- Equipo de desarrollo
- Help Desk
- Asistencia Técnica en el cliente

El servicio que se realiza a la farmacia abarca todos estos puntos:

- Creación de un software para la gestión integral de la farmacia

- Mantenimiento evolutivo del software en cuanto a desarrollo
- Instalación y mantenimiento correctivo del sotfware
- Venta de hardware: PC, servidores, periféricos, redes etc
- Instalación y mantenimiento del hardware

A continuación se detallan los roles, actividades y procedimientos que actualmente se realizan en cada departamento.

7.1.1.1. Equipo de desarrollo

El equipo de desarrollo es un equipo SCRUM habiendo adoptado esta metodología de trabajo hace varios años por lo que su nivel de madurez en cuanto a la aplicación de este marco de trabajo es avanzado.

Roles

- Scrum Master: este rol es asumido por el jefe de equipo. Además de llevar a cabo las actividades propias del Scrum Master forma parte del equipo de desarrollo llevando a cabo tareas de diseño, análisis, desarrollo y Testing, de la misma manera que el resto del equipo.
- Dueño de producto: esta figura la representa el responsable del producto en el negocio. Es el que tiene la visión global de los requerimientos y el que tiene la potestad de establecer las prioridades.
- Programadores: cada uno de los miembros del equipo de desarrollo. Realizan tareas de análisis, diseño, desarrollo y Testing. Se trata de un equipo multidisciplinar donde todos pueden asumir tareas de diferente índole.

Funciones

Las funciones del equipo de desarrollo son las siguientes:

- Analizar, diseñar y programar los requerimientos de la aplicación
- Realizar labores de Testing
- Realizar documentación técnica y de usuario de los desarrollos realizados
- Solucionar incidencias escaladas desde Help Desk y asistencia técnica

Procedimientos

Las actividades y procedimientos que lleva a cabo el departamento de desarrollo son las siguientes:

- Reuniones diarias: dentro del marco de Scrum se realizan reuniones diarias donde se revisa el estado del Sprint y cada miembro expone el trabajo realizado el día anterior y las tareas a las que se va a dedicar durante el día en curso. Si hay algún tema que comentar que pueda afectar al desarrollo del Sprint se hace en esta reunión.

- Reunión de retrospectiva: al final de cada Sprint se realiza una reunión de retrospectiva a la que acude el equipo de desarrollo, el Scrum Master y el dueño del producto. El Scrum Master presenta los resultados del Sprint, el grado de consecución, la velocidad alcanzada por el equipo. Se evalúan los resultados y se plantean entre todos posibles soluciones para problemas que hayan aparecido. Se plantean nuevos procedimientos de trabajo si se encuentra la necesidad de cambiar alguna de las prácticas realizadas en pos de mejorar y aumentar la productividad del equipo.

- Reunión de planificación de Sprint: el equipo de desarrollo, el Scrum Master y el dueño del producto deciden que tareas del la pila de producto se planifican en el siguiente Sprint y formarán parte de la pila de Sprint.

- Documentación del trabajo realizado: el programador realiza dicha documentación tanto en el propio código como en la herramienta de reporte de trabajo. La documentación en el código se lleva a cabo mediante unas normas establecidas en cuanto a nomenclatura, información de fecha y autoría de las modificaciones. En la herramienta de reporte se documenta totalmente el trabajo tanto desde el punto de vista técnico como a nivel de usuario.

Herramientas

Además de las herramientas de desarrollo propiamente dichas el equipo de desarrollo utiliza:

- JIRA: herramienta de reporte de tareas. Se utiliza concretamente JIRA Agile, un módulo de esta aplicación orientada a la gestión de equipos SCRUM o Kanban. También se utilizan diferentes pluggin de JIRA como Tempo para la planificación y gestión del tiempo de trabajo.

- Alfresco: herramienta de gestión documental en la que gestiona toda la documentación generada por el equipo, desde documentación técnica, documentación de usuario, documentación de procedimientos internos, etc.
- CVS: repositorio de versiones que está integrado en JIRA

7.1.1.2. Equipo de Help Desk

El equipo de Help Desk da soporte a los clientes. La entrada de estas peticiones de soporte llegan en aproximadamente un 90% vía telefónica y el otro 10% por correo electrónico.

Las necesidades de los clientes son motivadas por incidencias o bug producidos en el software pero predomina que el motivo de la llamada sea por consultas o falta de conocimiento de cómo realizar una operatoria con la aplicación.

Los agentes deben ser capaces de gestionar cualquier tipo de incidencia pero la realidad es que muchas veces no pueden realizarlo con éxito y deben escalar el problema al equipo de desarrollo. No existe una primera línea de atención y una segunda línea. Cuando la primera línea no es capaz de solucionar el problema se escala directamente a programación.

Roles

- Coordinador del Help Desk: esta figura realiza la priorización de las peticiones que llegan de los clientes mediante correo electrónico. Da soporte al resto de agentes ayudándoles cuando tienen dificultades.
- Agentes Help Desk: llevan a cabo las atenciones a los clientes

Funciones

- Atender las llamadas de los clientes que entran a través de una centralita habilitada para este efecto
- Atender las peticiones de los clientes que llegan por correo electrónico
- Reportar las atenciones realizadas en la herramienta destinada a este efecto
- Categorización de las peticiones de atención de los clientes que llegan por correo electrónico
- Interlocutores de los problemas del software. Comunicarlo para que el departamento de desarrollo lo solucione.

- Interlocutores de problemas de hardware. Comunicación al departamento de asistencia técnica para que lo solucione.

Procedimientos

- Reunión semanal donde se ponen en común temas que el equipo considera de interés. En estas reuniones solo intervienen miembros del equipo de Help Desk.
- Reporte de las atenciones recibidas en la herramienta de reporte. Información del motivo de la llamada y acción realizada. En caso que la atención haya derivado en un escalado al departamento de desarrollo no se realiza un seguimiento total de la incidencia.

Herramientas

- Software de centralita: los agentes están conectados a una centralita mediante un software que gestiona las llamadas mediante una cola
- Herramienta de reporte: cada una de las atenciones realizadas de forma telefónica quedan registradas en una herramienta de reporte que forma parte de un paquete mayor donde se gestionan presupuestos a clientes, facturación, inventario, atención técnica en farmacia, etc. Esta herramienta fue adquirida hace tempo por la compañía habiéndose quedado obsoleta en algunos aspectos.
- Cliente de Exchange: para gestionar las incidencias que son reportadas por los clientes mediante correo electrónico se utiliza Outllook y un buzón compartido de correo. Las incidencias entrantes son categorizadas por el coordinador y los agentes van revisando la cola de incidencias para gestionarlas. Es un sistema muy rudimentario ya que la gestión de buzones de correo no están pensados para este fin.
- JIRA: los agentes de Help Desk tienen acceso a la herramienta de reporte del equipo de desarrollo pero solo en modo de consulta

7.1.1.3. Equipo de asistencia técnica

Este departamento está formado por técnicos informáticos que asisten a las farmacias in situ. Las labores que realizan son principalmente de asistencia técnica en cuanto a montaje de hardware, instalación de redes, periféricos, etc. También deben realizar labores de mantenimiento del software, tanto a nivel de ofimática como en relación al software de farmacia.

Roles

- Coordinador del servicio de asistencia técnica: realiza labores de coordinación organizando las visitas de los técnicos, planning, etc. Gestiona también el inventario del departamento en cuanto al hardware necesario. Comunicación con clientes para temas comerciales, solución de problemas, etc.

- Técnicos informáticos: realizan visitas a las farmacias o en ocasiones realizan labores de mantenimiento en remoto. Dan asistencia de software y de hardware. Normalmente no están mucho en las oficinas sino visitando farmacias. Realizan Demos de la aplicación a potenciales clientes. Cuando visitan a las farmacias tienen que realizar también labores de formación al usuario.

- Técnico de taller: se responsabiliza de la reparación del hardware o el envío a los servicios especializados si es el caso para la reparación. Mantenimiento del inventario de hardware, envío de material a las farmacias y a los técnicos.

Funciones

- Instalación del software y del hardware necesario en la farmacia

- Formación de los clientes cuando se realiza la instalación de una nueva farmacia

- Solución de problemas tanto de software como de hardware en la farmacia o en el taller o enviando hardware estropeado a las marcas correspondientes

- Interlocutores de los problemas de los clientes en relación con la aplicación

Procedimientos

- Revisión del planning de trabajo gestionado por el coordinador

- Registro de entrada y salida de elementos de hardware en el almacén del departamento

- Registro de envío de material de hardware para reparar en taller

- Reporte de incidencias al departamento de desarrollo

Herramientas

- Herramienta de reporte que es la misma utilizada por Help Desk y en la que se hace también control del inventario, presupuestos, facturación etc

- Correo electrónico: el intercambio de información con el resto del departamento es mediante correo electrónico
- JIRA: los técnicos tienen acceso a la herramienta de reporte del equipo de desarrollo pero solo en modo de consulta

7.1.1.4. Análisis de la situación actual

La observación de la organización actual de las diferentes áreas y los procedimientos utilizados así como el estudio de determinadas métricas lleva a la identificación de una serie de deficiencias y áreas de mejora que se detallan a continuación.

Nivel de servicio de Help Desk

Se obtiene información del nivel de servicio del Help Desk obteniendo los resultados mostrados en la siguiente gráfica:

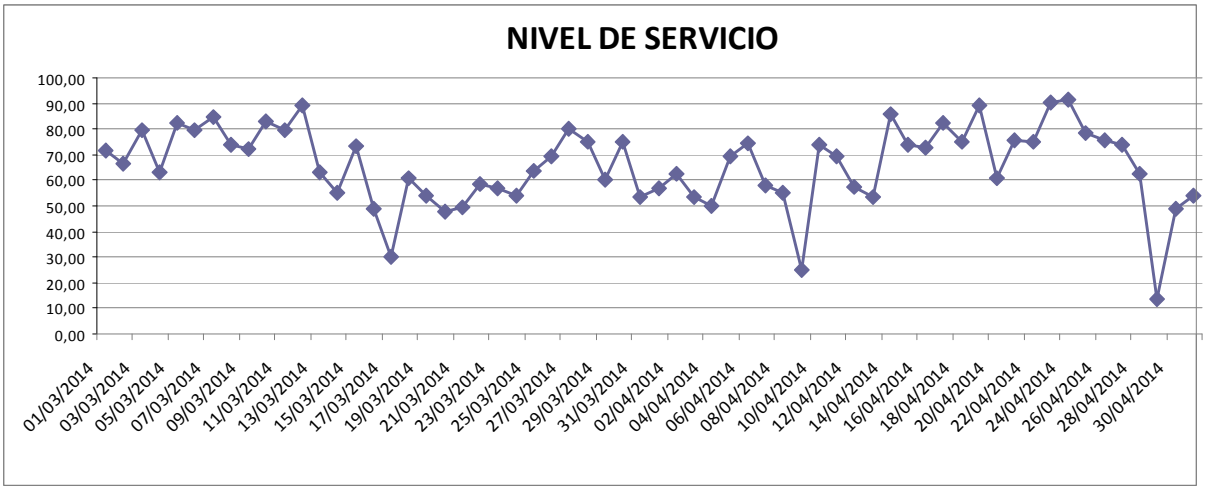

Figura 3 – Gráfica de nivel de servicio situación inicial (Propia)

El nivel de servicio tiene una media del 63%. El valor que cabría esperar por la compañía será de un 75%. Se reciben demasiadas llamadas para lo que el servicio de Help Desk es capaz de atender.

Se identifica en el gráfico que los primeros días del mes el nivel de servicio sufre una caída importante. Se identifica esta situación con el periodo de facturación de las farmacias. El primer día del mes las farmacias presentan a los colegios para que sea tramitado a los organismos de sanidad las recetas dispensadas para comenzar el proceso de facturación de

las mismas y recibir el importe que queda pendiente de pago después de recibir la aportación de los pacientes al retirar los medicamentos.

El software de farmacia tiene incorporada la funcionalidad para realizar este proceso de facturación. Los datos muestran que las farmacias tienen dificultades para realizar dicho proceso y acceden al servicio de soporte para, en la mayoría, realizar consultas acerca del proceso a realizar.

Como conclusión se deduce que el nivel de formación del cliente es bajo. Si se incrementara el conocimiento por parte del cliente se reducirían las llamadas y el nivel de servicio aumentaría.

Clasificación de las llamadas

A partir de la categorización del motivo de la llamada de los clientes y de las peticiones de soporte realizadas mediante correo electrónico se realiza un estudio para estudiar cual es el origen principal de las necesidades de los clientes. Se obtiene la siguiente gráfica:

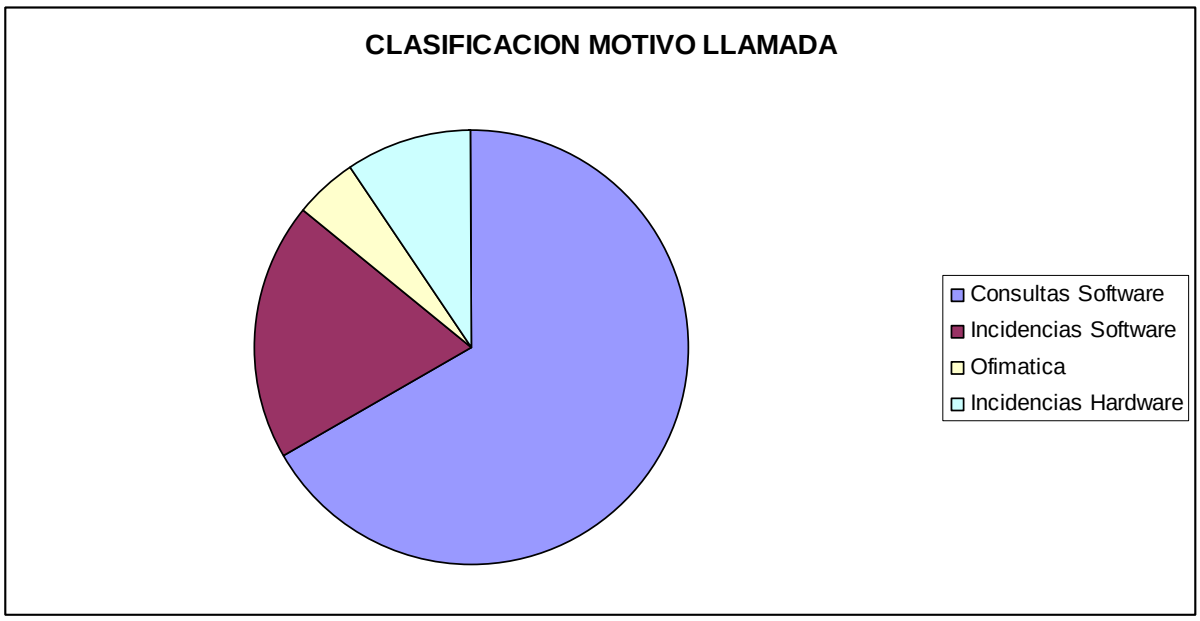

Figura 4 – Gráfica clasificación motivo de llamada situación inicial (Propia)

El principal motivo de llamada de los clientes es dudas que tienen acerca del funcionamiento del software. La falta de formación de los usuarios es de nuevo un problema a tratar.

Nivel de Bug frente a desarrollos realizados

Con el objetivo de revisar la eficacia del equipo de desarrollo se realiza un estudio de la proporción entre el desarrollo realizado para mejoras y nuevas funcionalidades de la aplicación y el esfuerzo dedicado a la solución de BUG generados por el propio software.

Se realiza un estudio con los datos recopilados del año 2013:

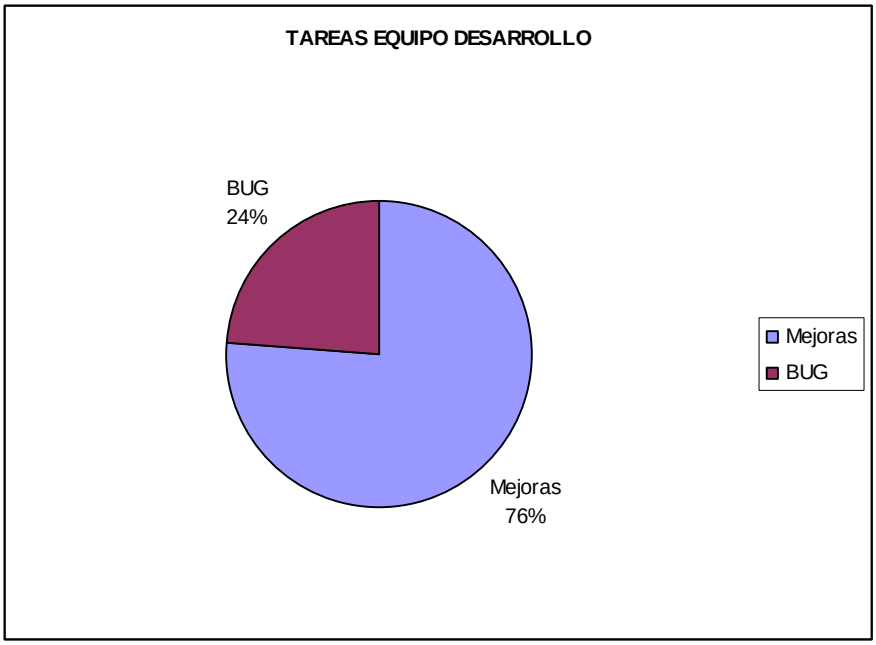

Figura 5 – Gráfica clasificación de tipos de tareas situación inicial (Propia)

El nivel de BUG es de un 24% lo cual se considera un porcentaje excesivo.

Tiempo de resolución de una incidencia

Otra métrica a tener en cuenta es el tiempo medio de resolución de los BUG aparecidos en el software por parte del equipo de desarrollo

Para la obtención de este dato se realiza una encuesta de satisfacción a los clientes. Una de las preguntas realizadas en esta encuesta es valorar de 1 a 10 la velocidad de resolución de errores de la aplicación, siendo 1 el valor más bajo y 10 el valor más alto en cuanto al grado de satisfacción.

Esta es la respuesta recibida por parte de los clientes:

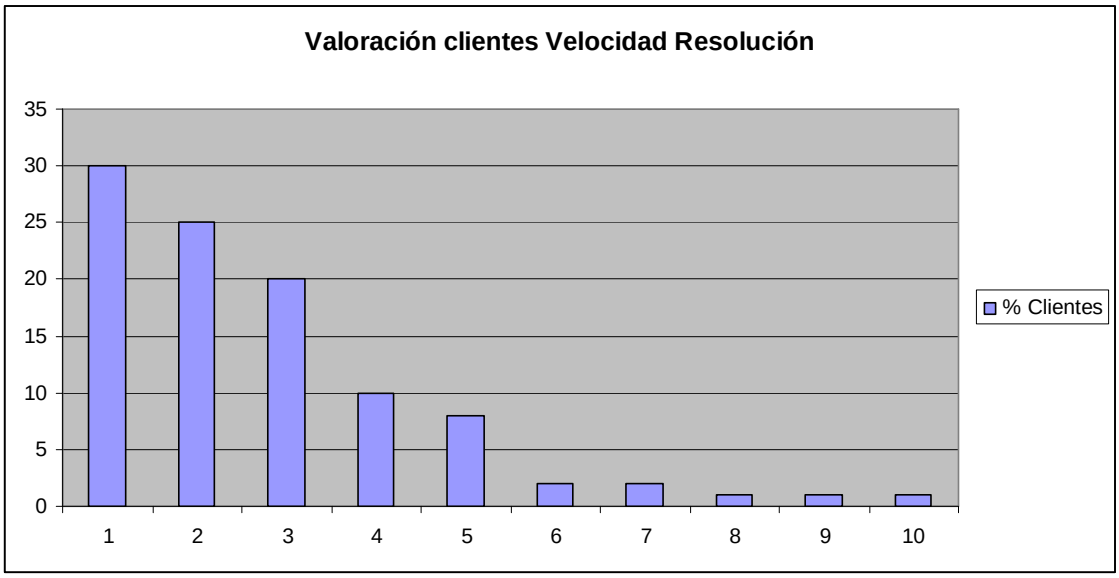

Figura 6 – Gráfica la valoración clientes velocidad resolución soporte situación inicial (Propia)

Los BUG del software se detectan de forma tardía. Este hecho no solo tiene como consecuencia la insatisfacción del cliente sino que el hecho de no detectar pronto la causa de los problemas provoca que la afluencia de llamadas por este motivo sea alto, que el software tenga errores internos no subsanados de manera que es mas inconsistente, etc.

Nivel de incidencias escaladas por Help Desk y asistencia técnica al equipo de desarrollo

Se realiza un estudio del tiempo empleado por los componentes del equipo de desarrollo en atender incidencia escaladas tanto por Help Desk como por los técnicos calle. Según los datos obtenidos alrededor de un 14% del tiempo de trabajo del equipo de desarrollo se invierte en solucionar este tipo de incidencias. El nivel de formación de los agentes de Help Desk así como de asistencia técnica es bajo, por lo que se requiere demasiado a menudo la intervención del equipo de desarrollo sin que realmente sea necesario.

Nivel de satisfacción del cliente con Help Desk y Asistencia Técnica

En la encuesta realizada a los clientes se obtiene su valoración de su nivel de satisfacción con los que son la primera línea de contacto, es decir, Help Desk y la asistencia técnica en cuanto a la capacidad de solución aportada, es decir, si les solucionan o no los problemas o consultas por los que solicitan el soporte.

Los datos obtenidos son los siguientes:

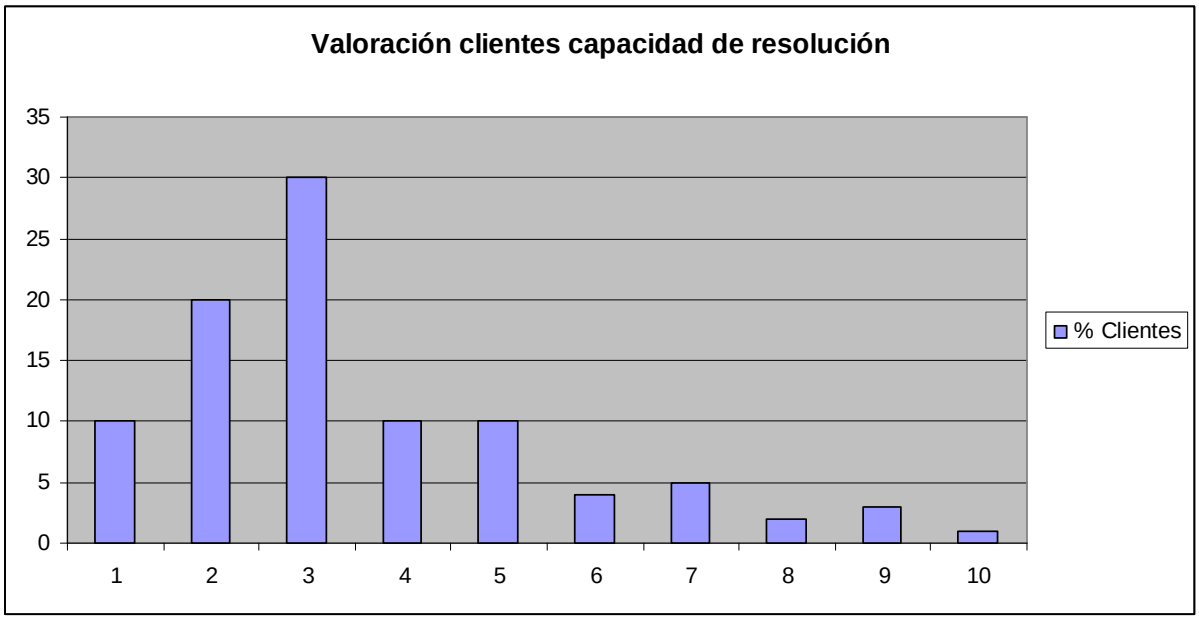

Figura 6 – Gráfica la valoración clientes capacidad resolución soporte situación inicial (Propia)

Los clientes tienen la percepción de que el departamento de Help Desk no les soluciona los problemas.

7.1.1.5. Conclusiones: objetivos de la metodología

Después de hacer un estudio exhaustivo de la metodología de trabajo aplicada actualmente por las diferentes áreas integrantes de la unidad de negocio se han detectado una serie de deficiencias que han de ser subsanadas. Estas serían los problemas detectados:

- Bajo nivel de servicio por parte de Help Desk: se producen muchas llamadas
- Nivel de formación de los clientes baja
- Nivel de formación de los agentes de Help Desk y asistencia técnica baja
- Nivel de resolución de problemas de los agentes de Help Desk y asistencia técnica baja
- Alto nivel de BUG en producción del software
- Detección y resolución tardía de los BUG del software

En base a estos problemas se detallan cuales serán los requisitos que ha de cumplir el nuevo marco de trabajo:

- Formación interna: se realizarán procedimientos y herramientas que favorezcan el incremento del conocimiento funcional y técnico de los agentes de Help Desk y asistencia técnica
- Formación de los clientes: se utilizarán procedimientos y herramientas que incrementen el nivel de formación de los clientes
- Reducción de BUG: se llevarán a cabo técnicas que reduzcan el nivel de BUG y que favorezcan la detección antes de la puesta en producción de las nuevas releases del software
- Detección y resolución temprana de los BUG: el marco de trabajo contemplará prácticas para la detección de los BUG lo antes posible y la resolución en el cliente lo antes posible
- Incrementar el grado de satisfacción de los clientes en cuanto al servicio recibido cuando solicita soporte

7.2. Descripción de la Metodología

7.2.1. Introducción

La metodología que a continuación se detalla tiene un marco de aplicación en equipos multifuncionales cuyo objetivo común es la creación y desarrollo evolutivo de un software y el mantenimiento y soporte a los clientes que adquieren dicho software.

La siguiente metodología intentar crear un marco de trabajo en el que las áreas de desarrollo y las de atención al cliente puedan aprovechar sinergias y crear procedimientos para optimizar el esfuerzo de cada área individual en pos de la mejora del producto y del servicio ofrecido al cliente.

La metodología se enmarca dentro de las metodologías ágiles. El software objetivo de este marco de trabajo tiene un alto nivel evolutivo. La aplicación de Scrum en el área de desarrollo de este tipo de software tiene resultados muy positivos. Muchas de las prácticas de esta metodología pertenecen al marco de trabajo de Scrum.

7.2.2. Conceptos

Estos con algunos de los conceptos que se aparecen en esta metodología:

Scrum

Marco de trabajo para el control de procesos de desarrollo basado los valores generados por el manifiesto ágil.

Sprint

Cada una de las iteraciones que se realizan el desarrollo y mantenimiento de un producto software aplicando la metodología Scrum

Velocidad

La velocidad del equipo de Scrum es el porcentaje del tiempo que el equipo puede dedicar a lo planificado en la pila de Sprint. Si por ejemplo la velocidad es del 70% quiere decir que el 30% del tiempo de trabajo de los miembros del equipo no se dedica a tareas propias del Sprint.

Historia de usuario

Se trata de un requerimiento que debe ser implementado en un producto software. Se describe en lenguaje natural expresando, en la medida de lo posible, escueta y claramente lo que el cliente quiere. Las historias de usuario se podrán dividir en una o varias tareas.

Pila de Producto

Todas y cada unas de las historias de usuario que el dueño del producto decide que deben ser implementadas para conseguir construir el producto total.

Pila de Sprint

Las historias de usuario que el dueño de producto y el equipo Scrum han decidido incluir en el Sprint en curso.

Kanban

Se trata de otra de las prácticas englobadas dentro de las llamadas metodologías ágiles para la gestión de procesos que persigue la no sobrecarga de trabajo de los miembros limitando el trabajo en curso.

SLA

Acuerdo de nivel de servicio (Service Level Agreement). Se trata del acuerdo al cual una entidad que da un servicio lleva con el cliente benefactor de dicho servicio en cuanto a calidad del mismo. En el caso de un servicio de soporte a clientes un SLA podría basarse en el tiempo de respuesta y el tiempo de resolución de las incidencias.

7.2.3. Equipos y Roles

Se definen los siguientes equipos de trabajo y roles.

7.2.3.1. Equipo de desarrollo

El equipo de desarrollo tendrá como objetivo implementar cada uno de los requisitos funcionales y técnicos que se vayan planteando. Dichos requisitos formarán parte de lo que llamamos pila de Scrum. Los roles en el equipo de desarrollo serán los siguientes:

Desarrolladores

Analistas programadores que forman el equipo de desarrollo. Trabajan en equipo para lograr conseguir los objetivos marcados en la planificación del Sprint. Son equipos autoorganizados de manera no necesitan la figura de un gestor que les organice el trabajo. La calidad es una de las premisas principales en los equipos Scrum. Existe sentimiento de equipo y hace que el equipo en si, y no las personas, sean las responsables del trabajo realizado.

El equipo, además de llevar a cabo la consecución del Sprint, priorizará la resolución de BUG detectados tanto por el equipo de Testing como los detectados en producción y comunicados a través del flujo de información interna procedente de los equipos de atención al cliente.

Se deben aplicar en la medida de lo posible técnicas para la disminución de errores y código mas limpio como pueden ser la programación en pareja, refactorización o desarrollo dirigido por test (TDD).

Scrum Master (SM)

Figura líder del equipo de desarrollo que está a disposición del resto eliminando los problemas que puedan aparecer mientras se lleva a cabo el Sprint. El SM es entre otras cosas el interlocutor principal con el resto de componentes y áreas del negocio. Recae sobre él la responsabilidad de informar y ser el nexo de unión con otras áreas como por ejemplo la de atención a clientes. A su vez el SM tiene que comunicar al equipo de desarrollo lo que acontece fuera del área de desarrollo, sobre todo en lo que concierne a los clientes.

El equipo de desarrollo no debe perder de vista que lo que se hace es siempre por y para el cliente. El SM debe asegurarse de que esta premisa está siempre presente en el equipo y es el interlocutor de las necesidades del cliente.

Dueño del producto

El dueño del producto es la persona que decide cuales son los elementos de la pila de producto, el que los prioriza y el que se responsabiliza de que los elementos seleccionados para la pila de producto sean de importancia relevante para el cliente.

Según el marco de trabajo de Scrum el dueño del producto debe ser una persona única que como mucho puede representar los intereses de un comité aunque en última instancia sea él que interfiera con el equipo Scrum y el SM.

Dada que esta metodología quiere aunar esfuerzos entre las diferentes áreas funcionales el dueño del producto deberá representar los intereses de los clientes, y por lo tanto los responsables de las áreas que están en contacto directo con los clientes deben ser el comité al que represente el dueño del producto. Por lo tanto se creará un comité formado por los coordinadores de los departamentos de soporte a los clientes.

Comité de atención al cliente

El comité de atención al cliente estará formado por representantes de las áreas funcionales que están directamente en contacto con el cliente. En el caso de estudio del presente trabajo el comité de atención al cliente estará formado por el coordinador del departamento de Help Desk y el coordinador del departamento de asistencia técnica. Ambos componentes deberán tener información importante y relevante de las necesidades y problemas de los clientes de manera que sean los interlocutores de los mismos a la hora de realizar la pila de Sprint, establecer prioridades, etc.

7.2.3.2. Equipo de Soporte al cliente

El equipo de soporte al cliente es el que da atención de primera línea al cliente, bien vía telefónica o bien mediante otras vías (sistema de ticketing mediante correo electrónico, Chat, consultas en web de producto, etc). También puede haber una parte del equipo de soporte que realicen atenciones en casa del cliente. En el caso de estudio de la oficina de farmacia los equipos de soporte serán el equipo de Help Desk y el equipo de asistencia técnica.

La función principal de este equipo es atender rápidamente al cliente respecto a cualquier consulta, problema o necesidad que pueda tener.

Cualquier tipo de requerimiento tiene que ser atendido y resuelto, aunque sea necesario escalarlo al departamento de desarrollo o realizar una visita insitu en el cliente.

Se distinguirán los siguientes roles:

Coordinador de Soporte

Responsable del área de Soporte a clientes. Tendrá el papel de facilitador para el resto de los componentes del equipo. Se asegurará de que todos los miembros del equipo tengan las herramientas de trabajo adecuadas. Realizará labores de seguimiento para localizar los problemas más importantes ocurridos en los clientes de cara a comunicarlo a otras áreas. Se asegurará del cumplimiento del nivel de servicio comprometido asegurando el cumplimiento de las prácticas cuya finalidad es cumplir el nivel de servicio (SLA) comprometido con los clientes.

Agentes de Soporte

Son técnicos informáticos especializados en el software al cual dan soporte. Sus conocimientos son tanto técnicos como funcionales. Deben ser capaces de solucionar cualquier necesidad que tenga el cliente. Atienden las peticiones de los clientes tanto vía telefónica como por cualquier otro medio que puedan llegar. Deben optimizar su trabajo para alcanzar el nivel de servicio acordado.

Tester

Se trata de una persona cuyo cometido es realizar pruebas de aceptación de usuario. El tester debe conocer ampliamente el requerimiento que va a probar. Para ello se contará con una herramienta donde se registre información del requerimiento realizado por el departamento de desarrollo así como la descripción completa de lo que se pretendía por parte del cliente que ha generado la petición o el BUG, según sea el caso.

El rol de Tester lo pueden llevar a cabo miembros de cualquiera de los equipos aunque será preferible que lo lleven a cabo miembros del departamento de Soporte a clientes. Con esto ser mejora el nivel de conocimiento de los agentes de soporte y además se incrementa la calidad del producto antes de salir a producción.

7.2.4. Procedimientos y buenas prácticas

Estos son los procedimientos que se aplicarán en las diferentes áreas de la unidad de negocio

7.2.4.1. Kanban para el equipo de soporte

El equipo de soporte realizará tareas de dos tipologías. Algunas que tienen que ser resueltas en el momento ya que el cliente tiene una necesidad que no puede aplazarse y otras que pueden realizarse a posteriori. Para realizar las tareas que se pueden planificar se aplicará un sistema Kanban.

Todas las tareas, sea cual sea el origen de las mismas (vía telefónica, correo electrónico, comunicación por otro miembro del equipo, etc) pondrán en un tablero Kanban informando, según el SLA pactado, cuando debe ser su resolución para cumplir los tiempos de respuesta y resolución pactados.

El rol encargado de la categorización de las tareas según el SLA y la prioridad de las mismas es el coordinador del equipo de soporte.

Los agentes irán realizando las tareas teniendo en cuenta los principios del Kanban. No se debe superar el WIP, es decir, la cantidad de trabajo en curso. El equipo establecerá a priori cual debe ser el WIP a tener en cuenta.

Los agentes tienen que se conocedores que la primera prioridad es atender las llamadas. Si el número de componentes del equipo lo permite se puede hacer que alguno de los miembros del equipo, durante franjas horarias o en días completos, se retiren de la atención telefónica y realice labores de resolución de tareas planificadas. Si se llevara a cabo esta práctica debería ser rotatoria de manera que todos los miembros del equipo realicen las mismas funciones y no se produzcan encasillamientos.

7.2.4.2. Seguimiento de las incidencias

El interlocutor con el cliente tiene que ser siempre el personal de soporte. Nunca un miembro del equipo de desarrollo debe ser el que hable directamente con el cliente. Para ello los agentes de soporte, en caso que se vean en la necesidad de escalar un problema, deben hacer el seguimiento de la resolución de dicho problema. Cuando el tema esté resuelto es el agente de soporte el que comunicará la solución al cliente. Es el agente de soporte el que debe estar al tanto de que se cumpla el SLA. Si detecta retraso en la resolución del problema debe agilizarlo o bien poniéndose en contacto directamente con el equipo de desarrollo o bien a través del coordinador de soporte.

El agente de soporte debe incrementar su conocimiento para que en futuras ocasiones no sea necesario escalar el problema. Se utilizarán herramientas colaborativas para el almacenamiento de este tipo de información y poder consultarse por parte de cualquier miembro del sistema.

7.2.4.3. Testing de miembros de soporte

Los miembros del departamento de soporte realizarán labores de Testing. Esto tendrá dos beneficios a tener en cuenta. Por un lado el incremento de la calidad de los desarrollos realizados al ser mas testeados. Por otro se incrementará el nivel de conocimiento del equipo de soporte.

7.2.4.4. “Learning Time”

Cuando un agente de soporte atiende una solicitud de un cliente la forma en la que es atendido debe ser lo mas didáctica posible. Es decir, en caso de tratarse una operatoria que el cliente desconoce se debe trata de explicar para que en futuras ocasiones pueda resolverlo por si solo.

7.2.4.5. Formaciones Online

Hoy en día el eLearning hace mucho más accesible y fácil la labor de formación. Con el fin de incrementar el nivel de formación tanto de los clientes como de todos los miembros del equipo se realizarán formaciones Online de los temas que se consideren necesarios.

7.2.4.6. Creación de una base de datos documental

Se contará con una herramienta colaborativa que permita que todos los miembros de los equipos compartan información. Se crearán entradas tipo wiky con descripción de problemas conocidos y solución de los mismos, artículos de cómo hacer operatorias frecuentes, documentación de usuario, etc.

La herramienta tiene que ser accesible de una forma fácil para que aun los miembros de soporte que visitan al cliente tengan acceso a la documentación.

Cada miembro del sistema, cuando considere que se debe documentar un problema detectado, una operatoria común, etc, debe ser responsable de crear la entrada en esta base de datos documental. Es importante que incluso los miembros del departamento de desarrollo alimenten esta información.

Se puede crear un sistema similar al que tienen algunos foros de expertos en los que se obtienen puntos por las respuestas realizadas. Este tipo de motivación es muy eficaz para incentivar la participación.

7.2.4.7. Documentación al acceso de los clientes

Se creará un portal web para los clientes donde estos puedan acceder a consultar información. En este portal se dispondrá de información tipo FAQ, manuales de usuario etc. Debe ser de fácil acceso para facilitar que los clientes lo consulten.

7.2.4.8. Información y priorización de incidencias

El equipo de soporte es el que mas inicialmente es conocedor de una incidencia o BUG ocurrido en el cliente. Se deben establecer mecanismos que prioricen estas incidencias y se resuelvan lo antes posible. Cuando un agente de soporte detecte un error del software

inicialmente realizará un Workaroud si es que existe o es conocedor de él para que el cliente pueda continuar con la actividad lo mas normalmente que pueda.

Seguidamente comunicará al coordinador de soporte la incidencia detectada. Este le asignará una prioridad en función de unos criterios que se establecerán a priori y será comunicado al Scrum Master.

El Scrum Master junto con el equipo de desarrollo, según la prioridad establecida para la incidencia, incorporarán la tarea a la pila de Sprint si lo ven adecuado. En caso contrario se dejará pendiente en la pila de producto, teniéndolo en cuenta para la próxima reunión de planificación de Sprint.

Es responsabilidad del coordinador de soporte el hacer seguimiento de este tipo de incidencias, tanto para la resolución inmediata como para asegurarse de que se tiene en cuenta en la siguiente iteración.

Cuando la incidencia es resuelta el cliente o clientes en los que se detectó deben ser informados de la solución de la misma. Esta información se llevará a cabo por un agente de soporte de manera que se fomente la credibilidad y confianza del cliente en el equipo de soporte.

7.2.4.9. Clientes Beta

Se seleccionará un grupo de clientes como clientes Beta. Las nuevas releases de la aplicación serán desplegadas inicialmente en los clientes Beta. Estos clientes tendrán que informar de cualquier incidencia detectada en la aplicación. Serán los agentes de soporte los interlocutores con estos clientes.

7.2.5. Eventos de la metodología

En este apartado se describen cada uno de los eventos que tienen lugar en la aplicación de esta metodología, en todas las áreas implicadas.

7.2.5.1. Reunión diaria de equipo

Diariamente se llevará a cabo una reunión interna de cada uno de los equipos. De la misma manera que el equipo de desarrollo realiza el llamado Daily Scrum o Scrum diario junto con el Scrum Master donde se ponen en común temas como avance del Sprint, trabajo realizado por cada uno de los miembros, trabajo planificado para el día etc, en la reunión diaria del resto de equipos se evaluará y valorará el trabajo realizado. Se revisarán las tareas pendientes en la pizarra de Kanban, se revisarán las prioridades de las tareas en función del SLA pactado con los clientes. Se realizará una puesta en común de las incidencias

procedentes de los clientes que se han reportado al equipo de desarrollo. El coordinador de soporte deberá posteriormente ponerse en contacto con el Scrum Master para conocer el estado de esas incidencias.

7.2.5.2. Reunión semanal de coordinadores

Semanalmente se realizará una reunión entre los coordinadores de soporte y el Scrum Master. Se tratarán temas como el estado de las incidencias que han sido reportadas desde los equipos de soporte a desarrollo, las que han sido incluidas en la pila de Sprint, las que no, en que situación están, etc. El Scrum Master informará del estado de evolución del Sprint. Se tratará también el estado de las tareas de Testing que realizan los agentes de soporte. Se informará debidamente al jefe de producto de lo tratado en la reunión haciendo un acta de la misma y compartiéndola para todos los miembros asistentes.

7.2.5.3. Reunión de retrospectiva

En la reunión de retrospectiva se reúnen los miembros del equipo de desarrollo, el Scrum Master y el dueño del producto. Acudirán también los coordinadores del equipo de soporte ya que ellos son la información más cercana procedente del cliente y la voz de los equipos de soporte. Se valorará como ha ido el Sprint. Se estudiarán los resultados obtenidos, nivel de cumplimiento de la pila de Sprint y nivel de imprevistos acontecidos. Se obtendrá la velocidad real del equipo durante el Sprint.

Se plantearán las historias de usuario realizadas en el Sprint. Los desarrolladores y los miembros de soporte revisarán tanto los nuevos requerimientos realizados como los que han ido surgiendo a lo largo del Sprint, sobre todo de cara a BUG solucionados. Es importante que los equipos de soporte sean conocedores cuanto antes de los BUG que ya han sido arreglados y en que releases del producto se pondrán a disposición de los clientes. Será función del equipo de soporte de informar a los clientes que hayan reportado en su momento las incidencias que ya están solucionadas.

El equipo reflexionará sobre su situación, técnicas empleadas, el grado de satisfacción de cada uno de los miembros. Se pondrán sobre la mesa temas no tanto técnicos sino de funcionamiento interno. Cualquier problema o incidencia que pueda haber se deberá plantear de cara a solucionarlo y poner las medidas.

Se revisan los objetivos acordados en la anterior retrospectiva. Se calcula el nivel de cumplimiento. En caso negativo se estudian las causas que lo puede haber provocado para tomar medidas al respecto.

Se expondrán aspectos que se deban tratar en relación con el flujo de trabajo entre los equipos de desarrollo y de soporte. Los coordinadores de soporte, como representantes del resto del equipo, plantearán lo que previamente se haya consensuado con su equipo.

Como resultado de la retrospectiva se decidirán una serie de objetivos a cumplir por el equipo de desarrollo y por los equipos de soporte. Dichos objetivos serán revisados en la próxima reunión de retrospectiva. Todo ello con el objetivo de conseguir una mejora continua tanto en la calidad del servicio y del producto como en los procedimientos empleados.

7.2.5.4. Reunión de planificación de Sprint

En esta reunión se define la nueva pila de Sprint para la siguiente iteración.

Se revisará la pila de producto pendiente. Los coordinadores de soporte aportarán información acerca de que requisitos deberían ser incorporados a la pila de producto y cuales de ellos, basándose en la experiencia del trato directo de sus equipos con los clientes, deberían ser incorporados en la próxima pila de Sprint.

En última instancia será el dueño de producto, los desarrolladores y el Scrum Master los que decidirán que historias de usuario se incorporan a la nueva pila de Sprint.

Se decide la duración el nuevo Sprint y se determinan cuando se realizarán las reuniones de Retrospectiva y de Planificación de Sprint siguientes.

7.2.5.5. Reunión de BUG

Se trata de una reunión donde el equipo de desarrollo y el Scrum Master revisan los BUG más importantes producidos en el Sprint anterior. Se evalúan las causas y se medita acerca de que herramientas o técnicas se pueden poner en práctica para solucionarlos como puede ser la programación en parejas, pruebas unitarias, etc.

Esta reunión se realizará en el un día que se dejará entre el último día de Sprint y el día utilizado para la reunión de retrospectiva y la de planificación de Sprint. Este día se deja “libre” para que el equipo realice labores de autoformación, organización de las herramientas utilizadas, revisión de documentación, etc.

7.2.5.6. Formaciones Online clientes

Con la finalidad de incrementar el grado de formación del producto tanto de clientes como del personal de soporte se organizarán formaciones periódicas mediante una herramienta de eLearning. A medida que se vayan realizando entregas del software con paquetes funcionales completados se organizarán este tipo de formaciones.

Serán impartidas por miembros del equipo de soporte. Al formar parte del equipo de Testing tendrán el suficiente conocimiento del producto como para hacer las formaciones al resto de compañeros así como a los clientes.

En las formaciones, después de hacer la presentación, los asistentes podrán intervenir dejando abierto un espacio de ruegos y preguntas. Además de ampliar el grado de conocimiento de los usuarios también se fortalecen las relaciones entre el cliente y el equipo de soporte.

De la misma manera se harán formaciones más técnicas orientadas al personal de soporte. En este caso las formaciones serán impartidas por miembros del equipo de desarrollo donde se informará no solo de la funcionalidad de los nuevos desarrollos sino de la parte técnica e en cuanto a bases de datos, arquitectura, etc.

Tanto las formaciones a clientes como a miembros del equipo serán grabadas y puestas a disposición de ambos en las plataformas habilitadas para tal efecto.

7.2.6. Artefactos de la metodología

Se definen los siguientes artefactos a utilizar:

7.2.6.1. Pila de Producto

En la pila de producto se guardan todas las historias de usuario que se tienen que implementar en el producto. Se trata de un producto altamente cambiante. Esta pila de producto es dinámica, es decir, nunca estará vacía ya que siempre aparecerán nuevos requerimientos.

Se recomienda utilizar una herramienta para gestionar esta pila de producto.

Es responsabilidad del Scrum Master mantener la pila de producto.

7.2.6.2. Pila de Sprint

Se trata de las historias de usuario seleccionadas para realizar en el Sprint actual. Se deben de mostrar en una pizarra Scrum. La pizarra puede ser real o virtual. Tanto si es de una manera como de otra se deben utilizar tarjetas para representar cada una de las tareas que conforman las historias de usuario seleccionadas para el Sprint.

7.2.6.3. Pizarra de Sprint

El equipo decidirá que columnas crean en la pizarra de Scrum. Lo más habitual es crear tres columnas: “Pendiente”, “En curso”, “Terminado”. También será decisión del equipo darle significado al concepto “Terminado”. Por ejemplo se puede añadir una cuarta columna que

sea “Probado” de manera que cuando una tarea está en “Terminado” quiere decir que “alguien” debe realizar las pruebas de aceptación de usuario. Si las pruebas son satisfactorias pasará al estado de “Probado” y de lo contrario volverá a “Pendiente” para que se resuelvan los problemas encontrados.

Existen muchas herramientas para utilizar pizarras virtuales muy adecuadas en el caso de que el equipo no esté físicamente ubicado en el mismo sitio.

7.2.6.4. Pizarra Kanban

Se trata de una pizarra de Kanban que la usarán los miembros de Help Desk para las tareas que puedan ser planificadas, es decir, todo aquello que no tenga que ser resuelto de inmediato.

Las columnas también serán elegidas por el equipo aunque también es habitual que sean “Pendiente”, “En Progreso” y “Terminado”. La particularidad principal es que en cada columna se marcará lo que se llama WIP, es decir Work In Progress. Si se decide que el WIP de la columna “En Progreso” es de 10 por ejemplo, quiere decir que no puede haber más de 10 tareas abiertas a la vez. Son tareas totales de todo el equipo. De esta manera se aumenta la productividad no permitiendo coger mucho trabajo sin haber terminado antes otras cosas empezadas. De la misma manera que la pizarra Scrum podrá ser real con tarjetas o virtual mediante una herramienta software.

7.2.6.5. Gráfica Burn&Down

Se trata de una gráfica donde se va evaluando día a día el trabajo realizado por el equipo y se compara con lo que sería el ideal del trabajo realizado:

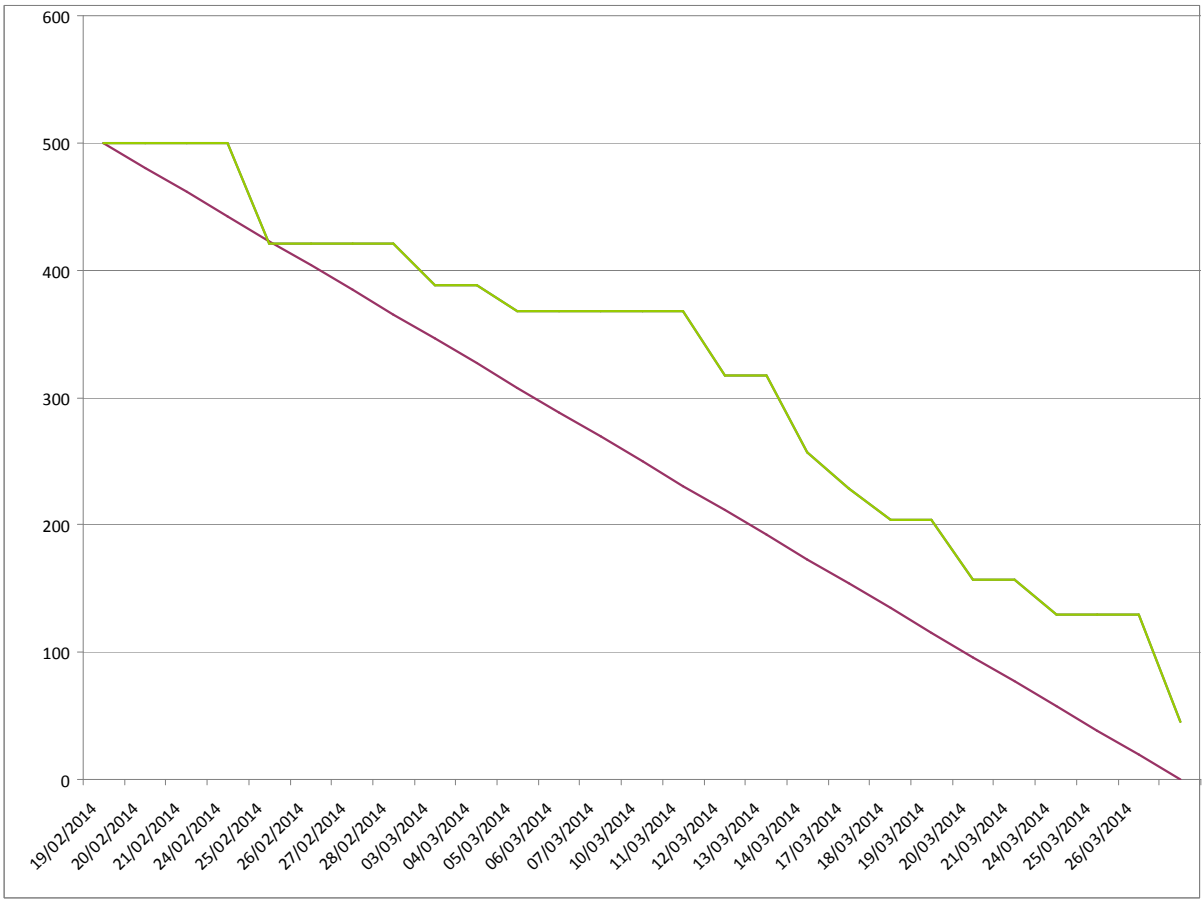

Figura 7 – Gráfica de Burn&Down (Propia)

En el eje vertical se representan los puntos de historia a realizar en el Sprint y en el eje horizontal se representa el tiempo. La gráfica muestra los puntos de historia que quedan de realizar en cada día. La línea roja es el ideal y la línea verde lo que realmente se ha trabajado.

El Scrum Master va actualizando la gráfica del Burn&Down cada día después de la reunión diaria y es visible por todo el equipo.

7.2.6.6. Herramienta colaborativa

Se elegirá un medio en el que el equipo pueda compartir información: documentación, procedimientos de trabajo, manuales de usuario, datos de clientes, etc.

Será accesible por todos los miembros el equipo.

7.3. Glosario de procesos de la metodología

A continuación se evalúan los diferentes procesos que forman la metodología:

7.3.1. Realizar una acción de soporte

Tanto un agente de Help Desk como de asistencia técnica realizan una atención de soporte a un cliente. Estos serán los pasos a realizar:

- Recopilar información del problema procedente del cliente
- Buscar información en la herramienta de base de datos documental en busca de temas relacionados
- Si es posible solucionar el problema. Si no hay documentación al respecto se hace en la herramienta colaborativa de compartición de conocimiento. Se le hace una explicación extensa al cliente de manera que éste pueda adquirir conocimiento y no necesitar nuevamente de la ayuda del soporte técnico
- Si requiere una labor de mantenimiento y no se sabe como intervenir se escala a segunda línea
- Si se trata de un BUG del producto se reporta la incidencia en la herramienta para tal efecto. Se informa al cliente de que está pendiente de resolución por parte de desarrollo. Se comunica al coordinador de soporte mediante correo electrónico o en la herramienta diseñada para ello.

7.3.2. Seguimiento de incidencias escaladas

Los agentes de Help Desk como de asistencia técnica tiene que hacer seguimiento de las incidencias que han escalado a segunda línea.

- Diariamente los agentes de soporte revisarán sus incidencias escaladas para comprobar su estado
- Las incidencias ya resueltas serán comunicadas a los clientes con un informe de la intervención realizada
- En caso que se pueda incumplir el SLA en alguna de las incidencias escaladas se informará al coordinador de soporte

7.3.3. Solución de tareas escaladas

El departamento de desarrollo, dentro de las labores imprevistas del Sprint, tendrá que solucionar tareas escaladas desde los equipos de soporte.

- El Scrum Master incorpora como imprevistos las tareas escaladas por los equipos de soporte
- Lo hará mediante tarjetas en una pizarra física o bien mediante una herramienta utilizada para esta gestión.
- El desarrollador evaluará el problema. Si es factible le explicará al miembro de soporte que lo ha reportado cual es la solución y que tiene que hacer.
- En caso que no sea posible dar instrucciones de las labores a realizar lo hará directamente el programador
- Informará en la tarea en que ha conseguido el trabajo realizado ya que es una fuente de conocimientos para los agentes de soporte en el futuro
- Cuando se termine la tarea se comunica a los miembros de soporte mediante la propia herramienta, correo electrónico o personalmente.

7.3.4. Testeo de desarrollos

Los agentes de Help Desk participarán en las labores de Testing. Se reservarán horarios de baja afluencia de peticiones de clientes para realizar estas tareas.

- El agente de Help Desk accederá a la pila de tareas pendientes de Testing
- Seleccionará una tarea y se la asignará. Se utilizará una herramienta de software para esta gestión. En el caso de utilizar una pizarra física se realizará la asignación de la tarea mediante un identificador en la tarea
- En caso de existir dudas sobre la funcionalidad a testear se pondrá en contacto directamente con el equipo de desarrollo. Las tareas deberían estar totalmente documentadas en cuanto al requerimiento que representan y el desarrollo realizado.
- Se documentarán las pruebas realizadas
- Si las pruebas son satisfactorias se pasará la tarea a estado de “Terminado”. De lo contrario se pasará de nuevo a estado “Pendiente” informando de los errores encontrados.

7.3.5. Evaluación y reporte de los BUG

Los BUG se reportarán al equipo de desarrollo

- Diariamente se realizará una reunión del equipo de soporte y el coordinador
- Se revisarán los BUG reportados por los clientes. Se comentarán y se establecerá la prioridad de cada uno de ellos.
- El coordinador de soporte en la reunión semanal que se realiza entre los diferentes coordinadores de soporte y el Scrum Master pondrá de manifiesto la necesidad de priorizar los BUG que se consideren oportunos
- Si alguno de los BUG detectados tienen carácter de urgente se notificará de inmediato al Scrum Master sin esperar a la reunión semanal

7.3.6. Solución de BUG priorizados

Se describe en este proceso como el departamento de desarrollo prioriza los BUG reportados

- En la reunión diaria se revisan los BUG que hayan sido reportados por los coordinadores de soporte
- Si priorizan dichos BUG intercalándose con las tareas que componen la pila de Sprint. Si es posible se les da salida cuanto antes
- Cuando se solucionan estos BUG se documenta en herramienta seleccionada para tal efecto. Se informa al coordinador de soporte para que a su vez el equipo de soporte esté informado y se pueda comunicar a los clientes

7.3.7. Documentación de usuario

Se realizará documentación de usuario en el formato que se quiera (documentos, wiki, documentación web, etc). Dicha documentación será realizada por el equipo de soporte

- Cuando un paquete funcional esté totalmente probado y finalizado se procederá a realizar la documentación de usuario
- La documentación la realizarán los agentes de soporte que realizan labores de Testing

- La documentación será depositada en un repositorio al que tienen acceso los clientes

7.3.8. Documentación técnica

Se realizará documentación técnica disponible para los equipos de soporte en el formato que se quiera (documentos, wiki, documentación web, etc). Dicha documentación será realizada por el equipo de desarrollo.

- Cuando un paquete funcional esté totalmente finalizado se procederá a realizar la documentación técnica
- La documentación la realizará un miembro del equipo de desarrollo

7.3.9. Formación a clientes

Periódicamente se realizarán formaciones a clientes mediante una herramienta de eLearning. Será responsabilidad del coordinador de soporte que se lleven a cabo estas formaciones. La formación la realizará un miembro del equipo de soporte.

- Se realizará la convocatoria de los clientes a la formación. Dicha convocatoria se realizará por todos los medios posibles de comunicación con los clientes (correo electrónico, web corporativa, etc)
- Se realizará cada formación en dos sesiones diferentes, a diferentes horas para facilitar la asistencia de mas público
- Al finalizar la formación se dejará abierto un espacio de ruegos y preguntas
- Se grabará la formación en un vídeo que se dejará a disposición de los clientes en el repositorio al que tienen acceso los clientes

7.3.10. Formación a soporte

Periódicamente se realizarán formaciones para el departamento de soporte. Serán impartidas por el departamento de desarrollo. Será conveniente hacerlo antes de que las releases sean desplegadas en los clientes para que el departamento de soporte sea conocedor de las nuevas funcionalidades. Será responsabilidad del Scrum Master que se lleven a cabo estas formaciones.

- Se realizará la convocatoria de la formación a las áreas de soporte

- La formaciones se grabarán y se dejarán disponibles para todos los miembros de los equipos en la herramienta colaborativa de compartición de conocimiento

8. Evaluación

La metodología se ha evaluado siendo aplicada en el caso de estudio que se detalla en el presente trabajo: una unidad de negocio de una distribuidora de farmacia que entre otros servicios comercializa un software de gestión de oficina de farmacia.

8.1. Equipos de trabajo y roles

Los equipos de trabajo definidos y los roles después de aplicar la metodología son:

Equipo de Help Desk

- Coordinador de Help Desk
- Agentes de Help Desk
- Roles: coordinador, agente soporte, tester, formador

Equipo de Asistencia Técnica

- Coordinador de Asistencia Técnica
- Técnicos de calle
- Roles: coordinador, técnico, formador

Equipo de desarrollo

- Scrum Master
- Programadores
- Roles: Scrum Master, programadores, tester, formador

8.2. Herramientas utilizadas

Los equipos utilizan las siguientes herramientas:

- JIRA: herramienta para reporte de tareas del equipo de desarrollo. Se utiliza el pluggin JIRA Agile. Se utiliza la pizarra de SCRUM de JIRA. No se utiliza la pizarra física ya que no todos los miembros del equipo están ubicados en las mismas localidades. JIRA es también utilizado para la gestión de la pizarra Kanban y las tareas planificadas de los equipos de soporte

- Confluence: herramienta de la misma familia que JIRA (Atlassian) que se utiliza como herramienta de colaboración. En Confluence se registran procedimientos de trabajo, documentación de problemas, etc. Se mantiene por todos los miembros del equipo

- Alfresco: gestor documental utilizado para el almacenamiento de la documentación de usuario y técnica generada (manuales, etc)

- WebEx: herramienta para formaciones online tanto a clientes como a nivel interno

- Kayako: herramienta de soporte para gestión de tickets con los clientes. Mediante esa herramienta los clientes tiene una visibilidad exterior del estado de sus incidencias pudiendo consultarlo en cualquier momento

- Web del producto: se ha construido por parte del departamento una web para los clientes donde se pone a su disposición documentación, vídeos de formaciones, etc

- PhpList: gestión de colas de correo mediante el cual se hacen comunicaciones masivas a los clientes mediante correo electrónico

8.3. Puesta en marcha de la metodología

Se aplica la metodología de trabajo siguiendo estos pasos:

- Formación a todos los miembros de los equipos de la nueva metodología que se va a aplicar: roles, conceptos, procesos, artefactos, etc

- Formación a todos los miembros de las herramientas a utilizar

- Planificación de los eventos a realizar: se realiza calendario de los diferentes eventos que se van a producir en las siguientes cinco semanas, periodo estipulado de duración inicial del Sprint

- Se van aplicando los nuevos procesos de trabajo supervisados por los coordinadores de soporte y el Scrum Master
- Al cabo de cuatro Sprint de haber aplicado la nueva metodología se realizan cálculos de métricas y encuestas de satisfacción tanto a clientes como a miembros de los equipos de trabajo

8.4. Resultados

A continuación se detallan las métricas aplicadas y encuestas de satisfacción

8.4.1. Nivel de Servicio

Se evalúa el nivel de servicio después de la aplicación de la metodología. La percepción de los agentes es que las farmacias llaman menos para preguntar ya que se les ofrece información mediante otros medios.

La media del nivel de servicio se encuentra en el 70%, 7 puntos por encima del primer estudio. El objetivo al que se quiere llegar es al 75%.

Los resultados se valoran positivamente.

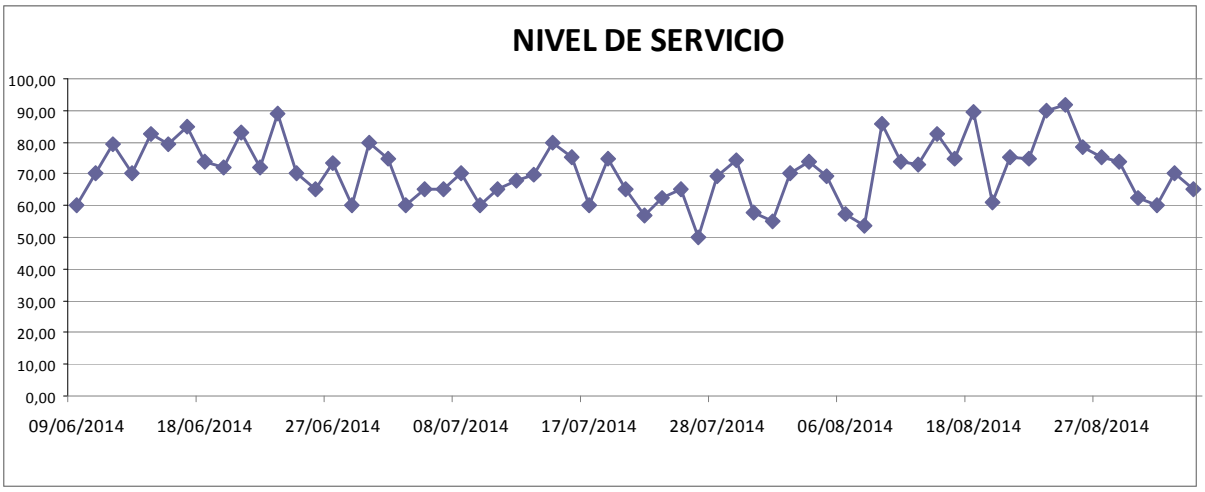

Figura 8 – Gráfica de Nivel de Servicio situación actual (Propia)

8.4.2. Clasificación de las llamadas

En cuanto a la clasificación de las llamadas este es el resultado:

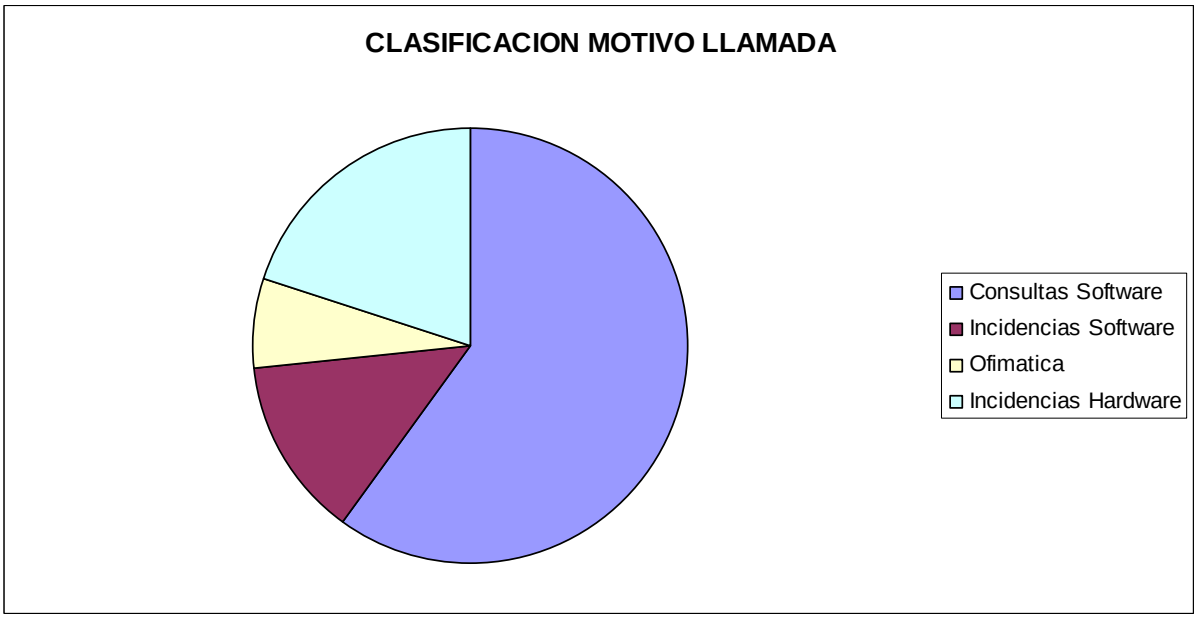

Figura 9 – Gráfica de clasificación de motivo de llamada situación actual (Propia)

Baja la proporción de consultas y de incidencias de Software. Se mantienen el resto.

8.4.3. Opinión de los miembros del equipo

En las retrospectivas después de cada Sprint se plantea al equipo de desarrollo y a los representantes de los agentes de soporte la sensación respecto a la nueva metodología. En general la opinión es positiva destacando estas fortalezas:

- Incremento del sentimiento de equipo global entre los tres departamentos
- Acciones de colaboración entre departamentos valoradas muy positivamente
- Incremento de la motivación de los miembros del equipo de soporte debido a tomar otros roles como los de tester y formador
- Mejora de la visión de los problemas del cliente por parte de los desarrolladores. Visión proporcionada por los compañeros de soporte

8.4.4. Opinión de los clientes

Se repite una encuesta de satisfacción a los clientes. El resultado global de la encuesta es:

- Valoración de los clientes de la velocidad de resolución

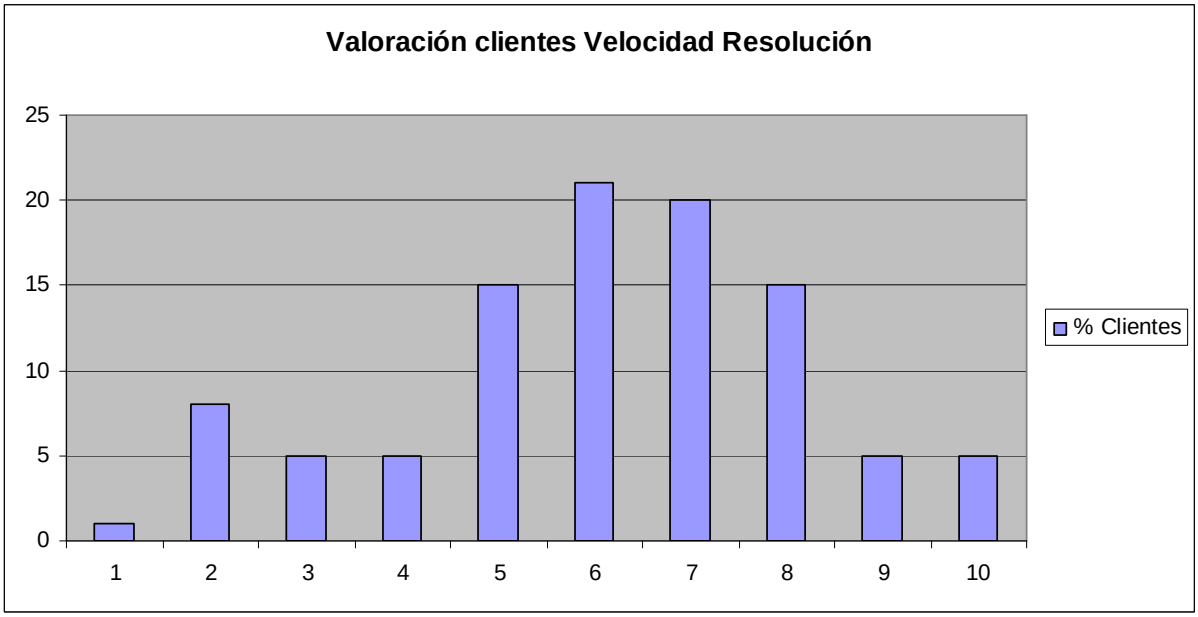

Figura 10 – Gráfica de valoración de clientes de velocidad resolución soporte situación actual (Propia)

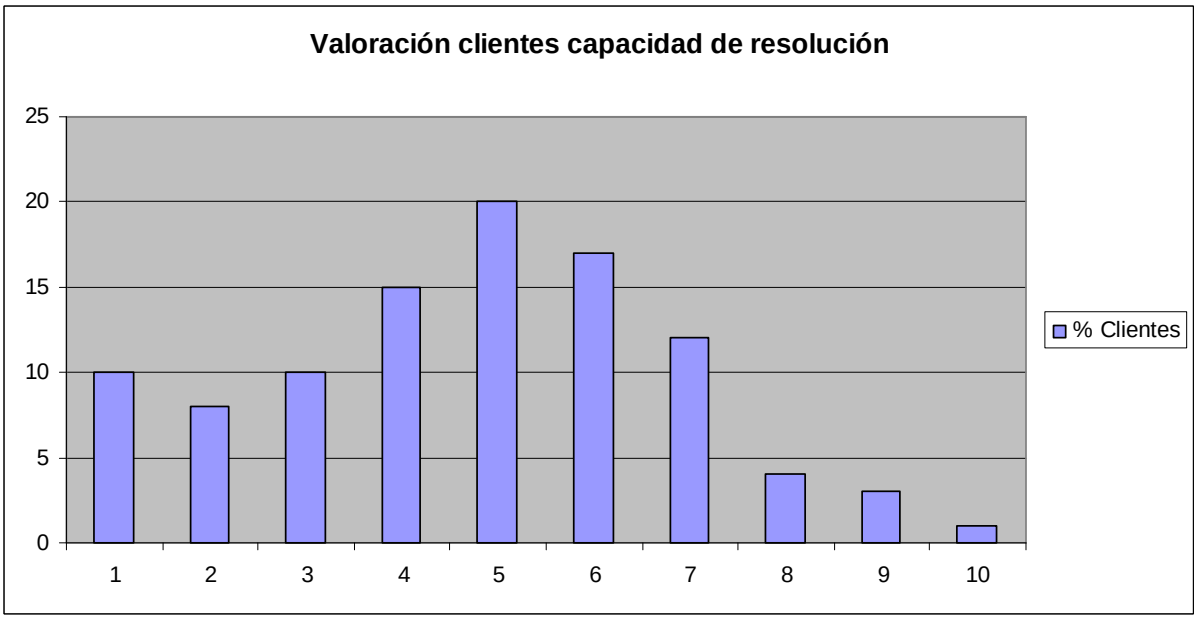

Figura 11 – Gráfica valoración de clientes de capacidad resolución soporte situación actual (Propia)

La opinión de los clientes ha mejorado. Consideran al equipo de soporte mas resolutivo ya que ellos son en todo momento la cara del negocio y son ellos los que les solucionan los

problemas, aunque haya sido escalando la incidencia o generando una nueva release con el problema resuelto.

8.5. Revisión de los objetivos

Las conclusiones revisando los objetivos y los resultados obtenidos son las siguientes:

- Se incrementa el nivel de servicio del equipo de Help Desk disminuyendo el número de llamadas provocadas por consultas ya que se incrementa la formación del usuario
- Se incrementa la formación del los equipos de soporte ya que se involucran en procesos de formación a clientes y en labores de Testing conociendo de primera mano las nuevas funcionalidades realizadas por el equipo de desarrollo
- Se incrementa el nivel de formación de los clientes con la formación continua que va realizando el equipo de soporte así como con la realización de formaciones organizadas
- Se incrementa la formación del equipo de Help Desk en cuanto a resolución de problemas debido a las formaciones internas realizadas
- Se disminuye el nivel de BUG en producción por el aumento de las tareas de Testing y por la detección temprana del BUG

9. Conclusiones y trabajo futuro

En el presente trabajo se ha abordado la posibilidad de aplicar prácticas de metodologías ágiles habitualmente empleadas solo en equipos de desarrollo en otras unidades funcionales también ligadas al desarrollo de software pero más enfocados al soporte a clientes. La elección de este tipo de metodología es debido a que las metodologías ágiles son un referente a la hora de ser aplicadas en entornos altamente cambiantes, premisa de la que parte también el presente estudio.

Las necesidades de crear esta metodología se encuentran en la observación de ciertos sistemas que crean y mantienen un producto software y al no tener procedimientos y métodos de trabajo compartidos entre las diferentes áreas funcionales no consiguen un producto y servicio con el valor requerido por el cliente.

La evaluación posterior a la aplicación de la metodología demuestra que este tipo de procedimientos y buenas prácticas puede ser aplicado con éxito en modelos de unidades de negocio similares a los expuestos en este trabajo.

En este estudio se ha aplicado con éxito la metodología en el caso de estudio expuesto. Una línea de trabajo futuro sería extenderlo a otras áreas diferentes, que no tuvieran que ver con el software al que hace referencia el caso de estudio. Una nueva línea de investigación se podría dirigirse hacia la normalización de las labores de Testing integradas en el desarrollo y personal de soporte aplicando técnicas del tipo de automatización de pruebas.

10. Bibliografía

- Poppendieck M., Poppendieck T.(2009) “Leading Lean Software Development: Results Are not the Point”. Addison-Wesley
- Poppendieck M., Poppendieck T.(2003) “Lean Software Development. AnAgile Toolkit”. Alistair Cockburn and Jim Highsmith, Serie Editors
- Agilemanifesto.org, (2001) “Manifiesto por el desarrollo Ágil del Software” http://agilemanifesto.org/iso/es/
- Huda, Preston, (1992) “Software Quality Journal” 1, 9-26 Recuperado el 9 de Septiembre de http://link.springer.com/article/10.1007/BF01720166#page-1
- Sommerville I (2005) “Ingeniería del Software”. Pearson Education
- Garzás (2010). “Las Leyes de la Evolución del Software”. Recuperado el 8 de Septiembre de 2014 de http://www.javiergarzas.com/2010/07/leyes-evolucion-software.html
- Wikipedia (2014). “Manifiesto Ágil”. Recuperado el 10 de Septiembre de http://es.wikipedia.org/wiki/Manifiesto_%C3%A1gil
- Garzas (2012), “Lean Software Development”. Recuperado el 9 de Septiembre 2014 de http://www.javiergarzas.com/2012/01/lean-software-development.html
- Martin (2008), “Clean Code”. Prentice Hall
- Womack, Jones (2000). “Lean Thinking”. Gestión 2000

- Wikipedia (2010). “Scrum”. Recuperado el 11 de Septiembre de http://es.wikipedia.org/wiki/Scrum
- Scrum.org (2013) “What is Scrum?”. Recuperado el 20 de Septiembre de https://www.scrum.org/resources/what-is-scrum/
- Crispin, Gregory (2009) – “Agile Testing “. Pearson Education
- Bruton (2013) “Why IT support can’t ignore the call of Agile, Kanban, Scrum, Sprint, and DevOps” Recuperado el día 16 de Septiembre de http://www.servicedesk360.com/why-it-user-support-cant-ignore-the-call-of-agile-kanban-scrum-sprint-and-devops-by-noel-bruton/
- Weinstock, Herman (2009) Applying Kanban to IT Processes (Part 2). Recuperado el 16 de Septiembre de http://blogs.lessthandot.com/index.php/ITProfessionals/ITProcesses/applying-kanban-to-it-processes-part-2/
- Cohn (2004) “User Stories and Planning y User Stories Applied for Agile Software Development”. Addison-Wesley
- Schwaber (2011). “Scrum is, Scrum is NET” – Recuperado de http://kenschwaber.wordpress.com/2011/08/11/scrum-is-scrum-is-not-2/ el 21 de Septiembre 2014
- Schawaber (1995) “System Development Process” . Recuperado el 19 de Septiembre de http://navegapolis.net/files/Scrum_Development_Process.pdf
- Garzás (2012) “Como lograr el mejor WIP de un Kanban”. Recuperado el 18 de Septiembre de http://www.javiergarzas.com/2012/04/el-mejor-wip-de-un-kanban.html
- Kniberg (2011) “Lean from the Trenches; Managing Large-Scale Projects with Kanban”. Kay Keppler
- Kniberg (2007) “Scrum y XP desde las trincheras”. InfoQ
- Systems At LitheSpeed (2010). “kanban for Service Desks”. Recuperado el 15 de Septiembre de http://lithespeed.com/kanban-for-service-desks/