# Aumento de la productividad en la gestión de proyectos, utilizando una metodología ágil aplicada en una fábrica de software en la ciudad de Guayaquil.

Julio Gamboa Manzaba

Facultad de Sistemas, Telecomunicaciones y Electrónica, Universidad de Especialidades Espíritu Santo, Km. 2.5 Vía Puntilla Samborondón, Ecuador
jgamboa@uees.edu.ec

**Resumen.** La ingeniería de software en la actualidad está llegando a cada organización. Oír hablar de ingeniería de software es un tema muy común, y esto se debe a que las organizaciones estánapuntando a como sobresalir o competir con las otras.
Las empresas que se dedican al desarrollo de software necesitan buscar nuevas maneras de desarrollar software y una de las opciones son las metodologías ágiles. Estas metodologías ágiles en la actualidad están dando excelentes resultados siempre y cuando se adopte la adecuada, según las necesidades de las organizaciones.
El objetivo de este estudio es poder demostrar que la implementación de una metodología ágil mejorará la productividad en la fábrica de software que se estudiará. Para esto se analizarán cuatro metodologías ágiles más utilizadas tales como Scrum, Extreme Programming, Kanban y Scrumban. De estas cuatro metodologías mencionadas se seleccionará una, aplicando una guía comparativabajo ciertos criterios de comparación, para luego implementarla en un proyecto piloto de la fábrica de software, con el fin de demostrar el impacto de competitividad en relación al tiempo de entrega del producto final.
Demostrando al final de este estudio por medios estadísticos y recolección de datos durante la aplicación del proyecto piloto, que la adopción de una metodología ágil aporta con el incremento de la productividad de la fábrica de software.

**Palabras clave:**ágil, Scrum, Kanban, Scrumban, metodología, manifiesto.

## 1. Introducción

En la actualidad escuchar hablar sobre gestión de proyectos de software es muy común, debido a que muchas organizaciones tienen la necesidad de actualizar sus sistemas o automatizar sus procesos, generando nuevos retos para empresas de desarrollo de software. Las empresas de desarrollo de softwareestán orientadas a mejorar su productividad, razón por la cual implementan nuevos métodos de innovaciones y una buena alternativason las metodologías ágiles, las mismas que están ayudando exitosamente a lograr éxitos en los desarrollos de software.Fred Villareal (2013) indica que de todos los beneficios con los que aportan las metodologías ágiles, producir resultados, el aumento de la productividad y la adaptación, son los beneficios principales que impactan a la competitividad de una organización de manera directa. La capacidad de una organización o empresa de sobresalir en el mercado donde se desarrolla, obteniendo una ventaja competitiva es lo que se conoce como competitividad (Porter, 2014).Existen varias metodologías ágiles, cada una tiene alguna característica distinta de la otra. Pero encontrar la que mejor se

adapte a las necesidades de la organización sería idóneo aunqueno garantice el éxito de un proyecto.

La adopción de una metodología ágil puede generar muchos beneficios para las organizaciones, pero también trae consigo varios obstáculos y dificultades.A pesar de todo, si se busca una mejora para la gestión de proyectos vale la pena hacer el esfuerzo. La reducción de tiempos es uno de los principales beneficios con los que aportan las metodologías ágilesen la productividad de las empresas y es lo que este trabajo pretende demostrar.

Uno de los principales problemas que enfrentan actualmente las empresas que se dedican al desarrollo de software, es la lucha constante contra el tiempo. El cliente siempre buscará calidad en el menor tiempo posible y para la disminución de tiempos se requiere de más personal capacitado. Por esto y más factores hoy las empresas que desarrollan software se ven en la necesidad de implementar métodos que ayuden en la gestión de sus proyectos.

Para llevar a cabo este trabajo una fábrica de software de la ciudad de Guayaquil, en su despertar, se da cuenta de que está involucrada en esta problemática general y decide incursionar en el tema de las metodologías ágiles. Dado que viene reportando una gran cantidad de llamados de atención, por atrasos en los cronogramas de trabajos, por parte de sus clientes.

Por lo cual este estudio previamente a la elección de una metodología ágil evaluará si la fábrica tiene un enfoque ágil, según los doce principios mencionados en el manifiesto ágil. Luego se propondrá a la fábrica cuatro metodologías ágiles (Scrum, Extreme Programming, Kanban, Scrumban), para ser analizadas en base a sus necesidades mediante una guía comparativa. Para finalmente seleccionar una de estas metodologías propuestas y aplicarla a un proyecto piloto de la fábrica, con la intensión de medir la productividad según los tiempos de entrega.

Por otro lado la fábrica de software en la que se realiza el estudio, es una organización que tiene tres años de vida y cuenta con un cliente principal líder del mercado, pero no es su único cliente. Mientras se desarrolló el estudio la fábrica contaba con cuatro proyectos diferentes con el cliente principal y dos proyectos con otros clientes.

La fábrica desde su creación a crecido en infraestructura, tecnología y personal, sin embrago también siente la necesidad de crecer en la manera de gestionar sus proyectos implementando temas de metodologías ágiles, para brindar a sus clientes un producto a tiempo y de calidad.

# 2 Fundamentación teórica

A continuación se abordarán los temas relacionados con el objeto de estudio, con la finalidad de mostrar y fundamentar a través de la literatura la relevancia del mismo.

## 2.1.Ingeniería de Software

La ingeniería del Software es el estudio de los principios y metodologías para el desarrollo y mantenimiento de sistemas de software (Zelkovitz, 1978).

La ingeniería de software también conocida como desarrollo de software es una aplicación del conocimiento científico en la construcción de programas (Software) y la documentación necesaria para desarrollar, operar y mantenerlos(Bohem, 1976).

La ingeniería de software es una disciplina de la ingeniería que tiene como meta el desarrollo costeable de sistemas de software. Éste es abstracto e intangible. No posee restricciones de materiales, o gobernado por leyes físicas o por procesos de manufactura, lo que hace que no existan limitaciones físicas del potencial del software (Sommerville, 2005).

La ingeniería de software es la aplicación del enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento de software, es decir, la aplicación de la ingeniería al software (IEEE, 1993).

Según estas definiciones se deduce que la ingeniería de software es una disciplina que estudia los procesos, método y herramientas involucradas en la producción de software de calidad, el mismo que debe de cumplir con los requerimiento por lo cual fue diseñado, ser confiable y fácil de usar para futuras mejoras.

### 2.2 Metodologías Ágiles

Ágil es el conjunto de buenos valores y buenas prácticas para el desarrollo de proyectos de software, también se lo denomina metodologías ágiles o métodos ágiles.

Todos los métodos ágiles están fundamentados en el manifiesto ágil, que es el resultado del trabajo de un grupo de expertos, siendo estos los mismos creadores de las metodologías ágiles.Teniendo como objetivo acordar y definir valores que ayuden a los equipos de trabajos a desarrollar software de manera eficiente, rápida y con adaptación a los cambios (Ágil, 2011).

El Manifiesto Ágil presenta 4 valores:

- Individuos e interacciones: sobre procesos y herramientas.
- Software funcionando: sobre documentación extensiva.
- Colaboración con el cliente: sobre negociación contractual
- Respuesta ante el cambio: sobre seguir un plan

Estos cuatros valores dieron origen a 12 principios ágiles que los podemos encontrar en el manifiesto ágil. En la Figura 1 se presentan los 12 principios:

**12 PRINCIPIOS**

- Sastifacer al cliente con Software de valor funcional.
- Aceptar el cambio (inclusive al final).
- Entregamos software funcional frecuentemente.
- Software funcionando es la principal medidad de progreso.
- Trabajo conjunto del negocio con el equipo de desarrollo.
- Brindar un entorno que soporte a un equipo motivado.
- Reflexionar sobre el trabajo a intervalos regulares.
- Simplicidad.
- Las mejores soluciones emergen de equipos auto-gestionados.
- Atención continúa a la excelencia técnica.
- Mantener un progreso sostenible.
- Comunicación cara a cara.

**Fig. 1.** Principios del Manifiesto Ágil.Disponible en la Manifiesto Ágil

Las metodologías ágiles que se van a estudiar son:

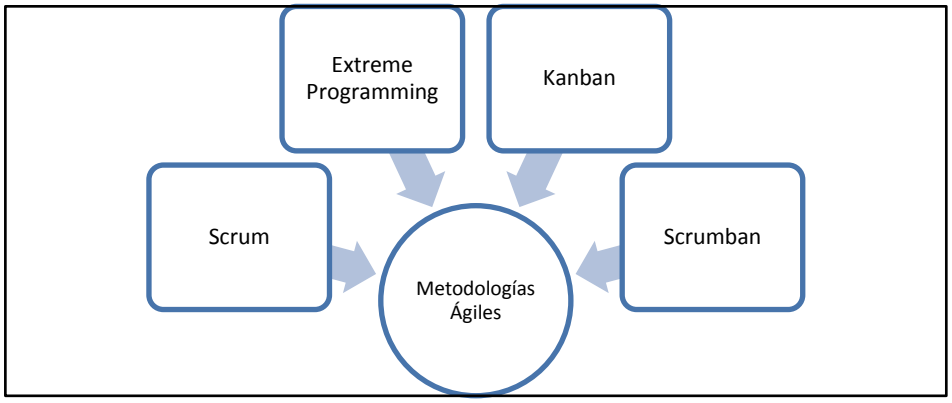



**Fig. 2.** Metodologías ágiles para el desarrollo de software

### 2.3.SCRUM

Scrum ha sido utilizado desde los años 90, usado para el trabajo de desarrollo y el mantenimiento de productos complejos. El uso de esta herramienta trae consigo problemas de adaptación, pero también ofrece beneficios como lograr entregar productos con un máximo valor, creatividad y productividad por parte de los equipos de trabajos.

Según los creadores de la guía de Scrum(Ken Schwaber y Jeff Sutherland, 2013) nos dicen que Scrum está basado en la teoria de control de procesos empíricos y que

emplea un enfoque iterativo e incremental para incrementar y optimizar la presivilidad del riesgo y el control. Scrum posee tres pilares que sostienen cada aplicación de control de proceso empírico: inspección, transparencia y adaptación.

### 2.3.1. Roles

En Scrum los equipos de trabajos deben tomar sus propias decisiones y ser multifuncionales, en otras palabras deben auto-organizarse. De esta manera se busca optimizar la flexibilidad, creatividad y la productividad(Ken Schwaber y Jeff Sutherland, 2013).

El equipo Scrum se divide en los siguientes roles:

- ProductOwner (Dueño del producto): Es el responsable de que el equipo de trabajo entienda claramente cuáles son los requisitos del proyecto, definir prioridades y maximizar el valor del producto de trabajo.
- DevelopmentTeam(El equipo de desarrollo): Son los profesionales del equipo quienes pueden gestionar su propio trabajo, queriendo obtener eficiencia y eficacia.
- Scrum Máster: Es el responsable de que el proceso de Scrum se entienda y de aplicar y hacer aplicar las reglas de Scrum. Es el líder ante el equipo de trabajo y debe asegurar que exista un lista priorizada de los requisitos, facilitar las reuniones de manera que sean productivas, enseñar al equipo a auto gestionarse, debe quitar los impedimentos y por ultimo proteger y aislar al equipo de interrupciones externas.

### 2.3.2. Eventos

Scrum es una herramienta que posee eventos prescritos, pretendiendo regularizar y minimizar la necesidad de programar reuniones no definidas. Scrum asegura de que se emplee una cantidad apropiada de tiempo de planificación, evitando desperdicio de tiempo utilizando time-boxes (bloques de tiempos) que tiene una duración máxima.

Cada uno de los eventos de Scrumestablecen una oportunidad para la inspección y adaptación de algún aspecto. Los eventos están diseñados específicamente para habilitar la vital transparencia e inspección(Villareal, 2013).

- **Sprint**: Es la parte principal de Scrum, es un bloque de tiempo (time-box) de un mes o menos durante el cual se crea un incremento del producto “Hecho”, utilizable y potencialmente entregable. Cada Sprint comienza con la reunión de planificación del sprint y termina con la revisión del sprint en la que el equipo de trabajo entrega el producto al propietario.
  El equipo de trabajo también se reúne después de cada sprint para realizar una reunión de retrospectiva, donde cada cual da su punto de vista de lo que se hizo o lo que se debió hacer con el fin de mejorar.

### 2.3.3. Artefactos de Scrum

La Guía de Scrum (2013), menciona que esta metodología dispone de tres artefactos o herramientas, con el objetivo de que los proyectos estén organizados:

- **ProductBacklog:** Es una lista dinámica de todos los requerimientos del producto quenecesariamente deben ir cambiando para tener un producto más adecuado, competitivo y siempre debe estar visible.
- **Seguimiento del Avance (Burndown):**Es un gráfico de trabajo pendiente a lo largo del tiempo, donde se muestra la velocidad con la que se están completando los objetivos del equipo.
- **Sprint Backlog:**Se visualiza en el tablero físico de Scrum, también llamado (Taskboard), donde el equipo de trabajo hace públicas sus actividades dividiéndolas en ítem (To Do – Doing – Done). Esta lista puede ir cambiando durante la ejecución del Sprint y es importante de que el equipo de trabajo constantemente actualice la lista, para poder ir estimando el trabajo por terminar.

## 2.4 EXTREME PROGRAMMING (XP)

El primer proyecto que se realizó con programación extrema se inició el 6 de marzo de 1996. XP resulta ser exitoso porque resalta la satisfacción del cliente. Este método se diferencia del modelo escala por que se base en la adaptabilidad en lugar de la predictibilidad.

Extreme Programming entre sus principales principios destaca en que cada iteración se determina el alcance de la próxima iteración, las entregas son frecuentes y continuas, se pone en producción rápidamente cada versión liberada, el cliente es integrado al equipo de desarrollo, se guía el desarrollo através de historias de usuarios, el cliente es quien define los casos de pruebas y se recomienda que se formen parejas de desarrolladores (Villareal, 2013).

### 2.4.1 Ciclo de Vida

La metodología de XP empieza con la planeación y las iteraciones consisten en cuatro fases básicas: planear, diseñar, codificar y probar. Se necesita de una comunicación continua entre los desarrolladores y el cliente, presentar soluciones simples, retroalimentación en las pruebas de aceptación y disponibilidad a enfrentar problemas y cambios en la fase de desarrollo, para que se realice el ciclo de vida en Extreme Programming(Villareal, 2013).

**RealsePlanning(Plan de liberación):**Esta es la etapa donde el cliente y el equipo de trabajo se reúnen, donde el cliente presenta los requerimientos funcionales y las historias de usuarios. A esta reunión se la conoce como PlanningGame.

**IterationPlanning(Planeación de la Iteración):**En esta etapa se realiza el plan de las tareas necesarias, para que el equipo de trabajo pueda entregar el producto final.

**Designing(Diseño):**El diseño es el inicio de las iteraciones en XP. Se requiere de un buen diseño para realizar cambios de manera simple al software sin afectar la funcionalidad. El diseño incremental debe ser simple, buscando hacer lo necesario sin agregar funcionalidad extra que no se necesita en ese momento.Esta no es una actividad que se realiza una sola vez en el proyecto, debido a que el equipo de desarrollo se reúne en sesiones rápidas de diseño antes de la iteración y revisiones de diseño.

**Codificación (Coding):**María Pérez (2012) menciona que codificar en XP es una forma de aprender, es tener un pensamiento y luego probarlo para saber si fue una buena idea.El código debe tener un formato estándar acordado por todo el equipo de trabajo, debe ser fácil de recordar y entender, para evitar problemas entre el equipo de trabajo.

En XP cada miembro del equipo siendo estos desarrolladores, verificadores y clientes,realizaráncontribuciones para obtener un producto de calidad.

- Desarrolladores: Realizando pruebas unitarias automatizadas en cada parte terminada. Con el objetivo de corregir cualquier error de codificación.
- El Cliente:Realizando pruebas funcionales e integrales, verificando que el producto va cumpliendo con lo que espera.
- Los Verificadores:Realizando pruebas de exploración, con el objetivo de identificar errores y junto con los desarrolladores analizar la situación para evitar problemas similares. Cuando las fallas son corregidas nuevamente los desarrolladores deben ejecutar las pruebas automatizadas para verificar que el problema efectivamente haya sido resuelto.

XP busca generar un producto de calidad, es por esta razón que implementa pruebas internas mientras se está en la etapa de codificación.

**Acceptance Test (Pruebas de Aceptación):** Son los clientes los que definen los escenarios de pruebas para cada historia de usuario y ellos mismo son los encargados de validar las pruebas y aprobarlas.

### 2.4.2. Roles

Hay diferentes roles en XP para diferentes tareas y propósitos durante el proceso y sus prácticas. A continuación, estos roles se presentan de acuerdo con(Abrahamsson, Salo, Ronkainen, & Warsta, 2002).

- **Customer(Clientes):** Responsables de definir qué funcionalidades debe de tener el software.
- **Programmers(Programadores):**Responsables de encontrar la forma más efectiva de entregar las historias dentro del plan, para esto deben hacer estimaciones de esfuerzo.

- **Testers(Verificadores):**Encargados de ayudar a los equipos XP a producir aplicaciones de calidad, deben colaborar con los clientes para determinar buenos escenarios de pruebas.
- **Coaches(Entrenadores):**Encargados de la motivación del equipo.

### 2.4.3. Valores en XP

Fred Villareal (2013) menciona que existen varios valores en XP y estos son:

- **Simplicidad:** Hacer exactamente lo que se ha pedido, no más.
- **Comunicación:**Debe existir una buena comunicación durante todo el proyecto.
- **Retroalimentación:** Siempre tener en cuenta la valoración del cliente una vez que se hace una entrega e intentar mejorar haciendo cambios en el proceso si es necesario.
- **Coraje:** Se trata que el equipo asuma la responsabilidad de su trabajo, tanto si es un éxito como un fracaso, además de ser emprendedor a la hora de implementar cambios en la aplicación.

## 2.5. KANBAN

Kanbanfue una técnica creada por Toyota, es una palabra japonesa que hace referencia a "tarjetas visuales", y se utilizó para controlar el avance del trabajo, en el contexto de línea de producción. Se ha convertido en un sinónimo de la aplicación justo a tiempo (JIT), que está diseñada para el control de inventario y reducir los tiempos. (Ordysiński, 2013).

Kanban tiene como objetivo principal gestionar de manera general como se van completando las actividades. Aunque Kanban no fue creada para la gestión de proyectos de software, en la actualidad se usa esta metodología dentro de esta área (Anderson, 2010).

### 2.5.1.Principios de la metodología Kanban

Ordysiński (2013) indica que el marco de trabajo de Kanban para el desarrollo de software puede ser presentado en 3 puntos principales:

1. Visualizar el flujo de trabajo.
2. Determinar el límite de trabajo en curso.
3. Medir el tiempo en completar una tarea

#### Visualizar el flujo de trabajo.

Para visualizar el trabajo, Kanban divide el trabajo en partes, cada una de estas partes se la escribe en un post-it y se lo pega en un tablero. Estos post-it contienen información como fecha de entrega de la actividad, fecha de inicio, descripción de la

tarea a realizar, etc. Los tableros tienen tantas columnas como estados o ciclos por los que puede pasar la tarea.


**Fig. 3.** Muro Kanban. Fuente David J. Anderson, 2010

El objetivo de esta visualización es tener claro el trabajo que se va a realizar, y los post-it indican el trabajo que está realizando cada integrante del equipo, para que el mismo pueda identificar las prioridades que se están dando. Las fases o etapas del muro se deciden según el caso, la figura 5 muestraun ejemplo general.(Anderson, 2010).

**Determinar el límite de trabajo en curso (Work in progress)**

Una de las principales ideas de Kanban es que el trabajo en curso debe ser limitado, es decir, que el número de tareas en cada fase del ciclo tiene que ser establecido y de conocimiento del equipo independientemente si el proyecto es grande o pequeño.Para poder agregar una nueva tarea es necesario que otra se haya terminado.

**Medir el tiempo en completar una tarea (Lead time)**

El tiempo en que se demora en finalizar una actividad o tarea se lo denomina "lead time", el lead time empieza desde que se realiza la petición hasta que se entrega la misma. También se suele utilizar otra métrica muy importante denominada "cycle time", el cycle time mide el rendimiento del proceso y se mide desde que se empieza a realizar la tarea hasta que se entrega la misma.

### 2.5.2.Roles

Como una característica que marca la diferencia entre Kanban y otras metodologías ágiles, está en que Kanban no utiliza roles.

Kanban entiende que la ausencia de roles es una ventaja para el equipo(Pérez, 2012).

Kanban nos permite saber el estado de los procesos y localizar rápidamente los problemas con un rápido vistazo. El éxito de esta metodología es que ayuda a los integrantes a auto-mejorar.

## 2.6.SCRUMBAN

AhmadKhan (2014) cita a (Pahuja, 2012) para definir que Scrumban nace utilizando la naturaleza prescriptiva de Scrumpara ser ágil y utiliza la mejora de los procesos de Kanban, por lo que Scrumban es una metodología derivada de Scrum y Kanban. Este término fue utilizado por primera vez por (Ladas, 2008) en su whitepaper denominado "Scrumban-EssaysonKanbanSystemsfor Lean Software Development".

AhmadKhan(2014) define las principales actividades que se realizan en Scrumban:

**Visualizar el flujo de trabajo:** Esta es una de las herramientas más importantes tomadas de Kanban y se aplica a Scrumban. Permite la visualización del flujo de trabajo. En un scrum normal, por lo general el equipo se inicia desde el sprint backlog y trabaja en los artículos y, finalmente, los mueve a la etapa de hecho. Sin embargo, en Scrumban la idea es visualizar el flujo de trabajo dentro y fuera de la Sprint. La visualización ayuda al equipo de trabajo, a identificar los cuellos de botellas. Además, la visualización ayuda a saber en lo que las personas están trabajando.

La visualización en la pizarra blanca o cualquier herramienta digital se hace dividiendo todo el tablero en diferentes etapas representadas por columnas. En la figura 4 se observa un ejemplo del flujo de trabajo visual utilizando Scrumban.

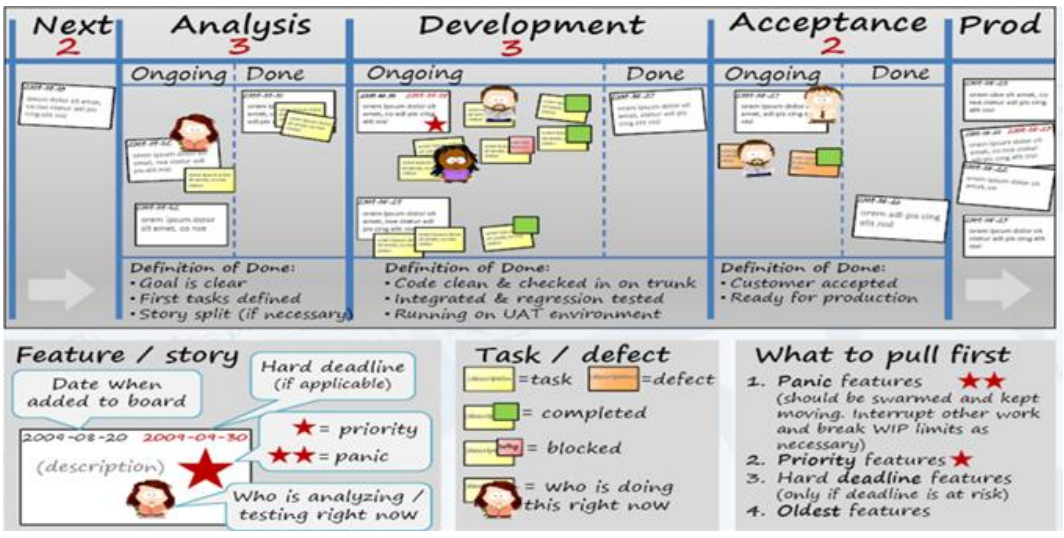



**Fig. 4.** Ejemplo de flujo de trabajo en Scrumban. Fuente:AhmadKhan, 2014

**Cola de Trabajo:** En Scrumban, el trabajo ingresa a una cola a diferencia del enfoque utilizado en un Scrum tradicional. Donde todo el trabajo que se realizará se le asigna

una fecha de inicio y una fecha de culminación para el sprint backlog. Esta cola contiene tareas que están pendientes de la cartera de pedidos, pero tienen alta prioridad. En esta cola no hay tareas vinculadas a alguna de las personas, pero tan pronto como alguien se desocupe, debe tomar una de estas tareas en lugar de recoger algo de la cartera general.

**Límite trabajo en progreso (WIP)**: Uno de los aspectos importantes de Scrumban es aplicar límites al trabajo en los puntos de progreso en todas las etapas, basado en la capacidad del equipo. Esto es un principioextraído de Kanban. EnScrum, significa limitar los elementos de la Pila de Producto (PBIS) que están en cursoen cualquier punto del tiempo, incluyendo el sprint backlog. El objetivo es ayudar al resto del equipo cuando un desarrollado haya terminado su actividad, logrando disminuir los cuellos de botellas.

**Reglas explícitas:** En Scrum la idea, es que los equipos son auto-organizados, trabajan y se coordinan a sí mismos, sin embargo, en la práctica existen siempre diferencias entre cómo un equipo debe organizarse y cómo están funcionando las cosas.

En Scrumban, las reglas del equipo se hacen explícitas para que todos en el equipo estén facultadospara auto-organizarse, con el fin delograr flujos de trabajos más suaves.

**Reuniones de Planificación:** A diferencia de Scrum, Scrumban tiene reuniones de planificación más cortas, con el fin de actualizar la cola de registro cuando sea necesario. El equipo siempre debe planificar para el período más corto por delante. Tener reuniones de planificación más largas no tienen sentido en el caso de que las prioridades cambien a menudo.

**Tabla 1**. Comparación de Scrum, Kanban y Scrumban

| | **Scrum** | **Kanban** | **Scrumban** |
|---|---|---|---|
| Visualización del trabajo | Parcial | Total | Total |
| Backlog | Sprint Backlog, ProductBacklog | Backlog con limites | Backlog con limites |
| Limite del WIP | No limitado | Limitado | Limitado |
| Cambios | Debe esperar hasta el siguiente Sprint | Según sea necesario | Según sea necesario |
| Roles | Si | No | Si |
| Estimaciones | Si | No | No |
| Iteraciones | Si | No | No |
| ScrumBoard | Se restablece después de cada Sprint | Tablero persistente | Depende de las decisiones de cada iteración |

Fuente: Ahmad Khan, 2014

## 3. Metodología

Este presente artículo académico experimental es de tipo correlacional debido que mide el efecto de relación entre la implementación de una metodología ágil y su impacto en la competitividad referente al tiempo de entrega del producto final. Los estudios correlaciónales miden dos o más variables que se pretende ver sí están o no relacionadas en los mismos sujetos y después se analiza la correlación (Roberto Sampieri, 2006).

Las variables que se midieron fueron las siguientes:

**Tabla 2.** Variables a medir.

| Variables | Sub-variables | Relación |
|---|---|---|
| Metodología Ágil | | Independiente |
| El grado de competitividad | Productividad referente al tiempo | Dependiente |

Como estrategia se han implementado los siguientes pasos:

En primero lugar, se ha implementado el método analítico realizando una investigación de cada una de las metodologías ágiles propuestas.

En segundo lugar,con las fuentes de datos investigadas, se ha seleccionado una metodología ágilmediante una guía comparativa, para aplicarla en un caso práctico, con el fin de poder reforzar la investigación.

A partir de lo anterior finalmente, mediante el procesamiento de datos obtenidos de la fábrica de software, se realizaron los respectivos análisis de correlación entre las variables definidas en la tabla 2, con el fin de cumplir con el objetivo del estudio.

Para la recolección de datos se utilizaron herramientas tales como: encuestas, entrevistas y reuniones grupales. La población de estudio fueron los integrantes de la fábrica de software tanto como el gerente, los líderes de proyectos y desarrolladores, tomando un solo proyecto como base con cuatro meses de duración según cronograma interno de la fábrica.

## 4. Implementación y análisis de resultados

Para brindar mayor soporte al estudio, primero se evaluará si la fábrica de software posee una orientación ágil, posteriormente se seleccionará una metodología ágil para implementarla en la misma fábrica y finalmente se analizarán los resultados de la implementación de la metodología seleccionada.

### 4.1. Orientación ágil

Antes de seleccionar una metodología ágil o de intentar adoptar una metodología, es necesario tener claro si la organización está orientada a estas. Por lo cual se va a relacionar cada principio ágil con la fábrica de software. El siguiente formulario ha

sido contestado por los cuatros líderes de proyectos que pertenecen a la fábrica de software.

**Valores de prioridad:**

**Tabla 3.** Valores de prioridad, para cada principio ágil.

| Valor Prioridad | Descripción Prioridad |
|---|---|
| 0 | Ninguna |
| 1 | Baja |
| 2 | Media |
| 3 | Alta |

**Tabla 4.** Principios del Manifiesto ágil.

| | Principio del Manifesto Ágil | Prioridad |
|---|---|---|
| **1** | Satisfacer al cliente mediante tempranas y continuas entregas de software que le aporte un valor. | |
| **2** | Aceptar cambios en cualquier etapa, para que el cliente tenga una ventaja competitiva. | |
| **3** | Entregar software frecuentemente funcional desde un par de semanas a un par de meses, con el menor intervalo de tiempo posible entre entregas. | |
| **4** | La gente del negocio y los desarrolladores deben trabajar juntos a lo largo del proyecto. | |
| **5** | Construir el proyecto en torno a individuos motivados. Darles el entorno y el apoyo que necesitan y confiar en ellos para conseguir finalizar el trabajo. | |
| **6** | El diálogo cara a cara es el método más efectivo para comunicar información dentro de un equipo de desarrollo. | |
| **7** | El software que funciona es la medida principal de progreso. | |
| **8** | Los procesos ágiles promueven un desarrollo sostenible. Los promotores, los desarrolladores y usuarios deberían ser capaces de mantener una paz constante. | |
| **9** | La atención continua a la calidad técnica y al buen diseño mejora la agilidad. | |
| **10** | La simplicidadesesencial. | |
| **11** | Las mejores arquitecturas, requisitos y diseños surgen de los equipos organizados por sí mismos. | |
| **12** | En intervalos regulares, el equipo reflexiona respecto a cómo llegar a ser más efectivo, y según esto ajusta su comportamiento | |

Fuente: Manifiesto ágil, 2012

Siendo 12 los principios ágiles y la prioridad más alta un valor de 3, entonces el valor más alto en cuanto al cumplimiento es de 36. Según el resultado obtenido $\Sigma$ *prioridades asignadas*, se deduce que las metas según el enfoque de la gerencia de proyectos de la fábrica se orientan en un $\Sigma$*prioridades asignadas* * 100/36 %al cumplimiento de los principios ágiles.

**Resultado Obtenido**

Para obtener datos, se ha realizado una encuesta a los líderes de proyectos que actualmente existen dentro de la fábrica de software.

**Tabla 5**. Resultado de la encuesta, realizada a los líderes de proyectos de la fábrica de software

| PRINCIPIO | L 1 | L 2 | L 3 | L 4 | PROM |
|---|---|---|---|---|---|
| La prioridad es satisfacer al cliente mediante tempranas y continuas entregas de software que le aporte un valor. | 3 | 3 | 3 | 2 | 2,75 |
| Dar la bienvenida a los cambios. Se capturan los cambios para que el cliente tenga una ventaja competitiva. | 1 | 3 | 2 | 2 | 2 |
| Entregar frecuentemente software que funcione desde un par de semanas a un par de meses, con el menor intervalo de tiempo posible entre entregas. | 3 | 2 | 3 | 2 | 2,5 |
| La gente del negocio y los desarrolladores deben trabajar juntos a lo largo del proyecto. | 2 | 2 | 3 | 2 | 2,25 |
| Construir el proyecto en torno a individuos motivados. Darles el entorno y el apoyo que necesitan y confiar en ellos para conseguir finalizar el trabajo. | 3 | 3 | 3 | 2 | 2,75 |
| El diálogo cara a cara es el método más efectivo para comunicar información dentro de un equipo de desarrollo. | 3 | 3 | 3 | 2 | 2,75 |
| El software que funciona es la medida principal de progreso. | 3 | 3 | 3 | 2 | 2,75 |
| Los procesos ágiles promueven un desarrollo sostenible. Los promotores, los desarrolladores y usuarios deberían ser capaces de mantener una paz constante | 2 | 2 | 3 | 2 | 2,25 |
| La atención continua a la calidad técnica y al buen diseño mejora la agilidad. | 1 | 3 | 3 | 2 | 2,25 |
| La simplicidad es esencial. | 2 | 2 | 2 | 2 | 2 |
| Las mejores arquitecturas, requisitos y diseños surgen de los equipos organizados por sí mismos. | 2 | 3 | 1 | 2 | 2 |
| En intervalos regulares, el equipo reflexiona respecto a cómo llegar a ser más efectivo, y según esto ajusta su comportamiento | 3 | 3 | 2 | 2 | 2,5 |
| **TOTAL** | | | | | **28,75** |

Fuente: Resultados propios, elaboración a partir del Manifiesto ágil, 2011

El resultado obtenido de la sumatoria de las respuestas de los cuatro líderes es ***28.75***, aplicando la formula obtendremos un porcentaje de ***80%***.

Se han obtenido datos objetivos que indican que la fábrica de software tiene un enfoque ágil del 80%, por tanto, el siguiente paso sería conocer qué metodología ágil, de entre las cuatro que se están evaluando en este estudio, se adapta mejor a la fábrica de software que se está estudiando.

### 4.2. Elección de la metodología ágil

Una vez teniendo claro que la fábrica está orientada al método ágil, procedemos a evaluar las cuatro metodologías ágiles propuesta (Scrum, XP, Kanban, Scrumban). Para esto vamos a utilizar una guía, para evaluar la forma de trabajo de la fábrica de software, basándose en los cuatro puntos de vistas de Iacovelli citado porMaría Pérez (2012).

- Uso: Reflejan el por qué utilizar las metodologías ágiles. Los atributos evalúan todos los beneficios que obtienen los equipo de trabajo y el cliente.
- Capacidad de agilidad: Representa cual es la parte ágil de la metodología. Los atributos de esta vista representan todos los aspectos del concepto de agilidad y su evaluación reflejan que aspectos están incluidos en las metodologías.
- Aplicabilidad: Representan cuando el entorno es favorable para la aplicación de metodologías ágiles.
- Procesos de productos: Representa como se caracterizan las metodologías. Los atributos caracterizarán a los procesos ágiles por dos dimensiones y listarán los productos de las actividades del proceso.

**Resultados Obtenidos utilizando la Guía Modelo: Fábrica de Software**

**Uso:** Refleja por qué utilizar metodologías ágiles.

**Tabla 6.** Resultados obtenidos de la evaluación, realizada todos los integrantes de la fábrica de software.

| | Premisa | Respuesta |
|---|---|---|
| **1** | Respeto de las fechas de entrega | V |
| **2** | Cumplimiento de los requisitos | V |
| **3** | Respeto al nivel de calidad | V |
| **4** | Satisfacción del usuario final | V |
| **5** | Entornosturbulentos | V |
| **6** | Favorable al Off shoring | F |
| **7** | Aumento de la productividad | V |

Fuente: María Pérez, 2012

**Capacidad de Agilidad:** ¿Cuál es la parte de agilidad incluida en el método?

**Tabla 7.** Resultados obtenidos de la evaluación, realizada todos los integrantes de la fábrica de software.

| | Premisa | Respuesta |
|---|---|---|
| **1** | Iteracionescortas | V |
| **2** | Colaboración | V |

| | | |
|---|---|---|
| **3** | Centrado en las personas | V |
| **4** | Refactoring político | F |
| **5** | Pruebapolítico | F |
| **6** | Integración de los cambios | V |
| **7** | De peso ligero | V |
| **8** | Los requisitos funcionales pueden cambiar | V |
| **9** | Los requisitos no funcionales pueden cambiar | V |
| **10** | El plan de trabajo puede cambiar | V |
| **11** | Los recursos humanos pueden cambiar | V |
| **12** | Cambiar los indicadores | V |
| **13** | Reactividad (AL COMIENZO DEL PROYECTO, CADA ETAPA, CADA ITERACIÓN) | ETAPA |
| **14** | Intercambio de conocimientos (BAJO, ALTO) | ALTO |

Fuente: María Pérez, 2012

**Aplicabilidad:** ¿Cuándo un ambiente es favorable para usar este método?

**Tabla 8.** Resultados obtenidos de la evaluación, realizada todos los integrantes de la fábrica de software.

| | Premisa | Respuesta |
|---|---|---|
| **1** | Tamaño del proyecto (PEQUEÑO, GRANDE) | GRANDE |
| **2** | La complejidad del proyecto (BAJA, ALTA) | ALTA |
| **3** | Los riesgos del proyecto (BAJO, ALTO) | ALTO |
| **4** | El tamaño del equipo (PEQUEÑO, GRANDE) | PEQUEÑO |
| **5** | El grado de interacción con el cliente (BAJA, ALTA) | BAJA |
| **6** | Grado de interacción con los usuarios finales (BAJA, ALTA) | ALTA |
| **7** | Grado de interacción entre los miembros del equipo (BAJA, ALTA) | ALTA |
| **8** | Grado de integración de la novedad (BAJA, ALTA) | BAJA |
| **9** | La organización del equipo (AUTO-ORGANIZACIÓN, ORGANIZACIÓN JERÁRQUICA) | AUTO ORGANIZACIÓN |

Fuente: María Pérez, 2012

**Procesos y productos:** ¿Cómo están caracterizados los procesos del método?

**Nivel de abstracción de las normas y directrices**

**Tabla 9.** Resultados obtenidos de la evaluación, realizada todos los integrantes de la fábrica de software.

| | Premisa | Respuesta |
|---|---|---|
| **1** | Gestión de proyectos | V |
| **2** | Descripción de procesos | V |
| **3** | Normas y orientaciones concretas sobre las actividades y productos | V |

Fuente: María Pérez, 2012

**Las actividades cubiertas por el método ágil**

**Tabla 10.** Resultados obtenidos de la evaluación, realizada todos los integrantes de la fábrica de software.

| | Premisa | Respuesta |
|---|---|---|
| **1** | Puesta en marcha del proyecto | V |
| **2** | Definición de requisitos | V |
| **3** | Modelado | F |
| **4** | Código | V |
| **5** | Pruebasunitarias | V |
| **6** | Pruebas de integración | V |
| **7** | Prueba del sistema | V |
| **8** | Prueba de aceptación | V |
| **9** | Control de calidad | V |
| **10** | Sistema de uso | V |

Fuente: María Pérez, 2012

**Productos de las actividades del método**

**Tabla 11.** Resultados obtenidos de la evaluación, realizada todos los integrantes de la fábrica de software.

| | Premisa | Respuesta |
|---|---|---|
| **1** | Modelos de diseño | V |
| **2** | Comentario del códigofuente | V |
| **3** | Ejecutable | F |
| **4** | Pruebasunitarias | V |
| **5** | Pruebas de integración | V |
| **6** | Pruebas de sistema | V |
| **7** | Pruebas de aceptación | V |
| **8** | Informes de calidad | V |
| **9** | Documentación de usuario | V |

Fuente: María Pérez, 2012

Los datos extraídos de estos formularios se compararán con la guía de clasificación de las cuatro metodologías que se muestran en el Anexo 1. Donde se asignará el valor de “1” si coinciden, caso contrario “0”.

**Tabla 12.** Resultados obtenidos de la comparación de los resultados de los formas anteriores con la guía del anexo 1.

| | | | METODOLOGIAS ÁGILES | | | |
|---|---|---|---|---|---|---|
| | | | **ORIENTADA AL DESARROLLO DEL SOFTWARE** | **ORIENTADA A LA GESTION DEL PROYECTO** | | |
| | | | **XP** | **SCRUM** | **KANBAN** | **SCRUMBAN** |
| **USO** | ¿Por qué utilizar una metodología ágil? | Respecto a las fechas de entrega | 0 | 1 | 0 | 0 |
| | | Cumplimiento de los requisitos | 1 | 1 | 1 | 1 |
| | | Respecto al nivel de calidad | 0 | 0 | 0 | 0 |
| | | Satisfacción del usuario | 0 | 1 | 0 | 0 |
| | | Entornosturbulentos | 1 | 1 | 1 | 1 |
| | | Favorable al Off shoring | 1 | 0 | 1 | 0 |
| | | Aumento de la productividad | 1 | 1 | 1 | 1 |
| **CAPACIDAD DE AGILIDAD** | ¿Cuál es laparte deagilidadincluida en elmétodo? | Iteracionescortas | 1 | 1 | 1 | 1 |
| | | Colaboración | 1 | 1 | 1 | 1 |
| | | Centrado en las personas | 1 | 1 | 1 | 1 |
| | | Refactoring político | 0 | 1 | 1 | 1 |
| | | Pruebapolítico | 0 | 0 | 1 | 0 |
| | | Integración de los cambios | 1 | 1 | 1 | 1 |
| | | De peso ligero | 1 | 1 | 1 | 1 |
| | | Los requisitos funcionales pueden cambiar | 1 | 1 | 1 | 1 |
| | | Los requisitos no funcionales pueden cambiar | 0 | 0 | 1 | 1 |
| | | El plan de trabajo puede cambiar | 1 | 0 | 1 | 1 |
| | | Los recursos humanos pueden cambiar | 1 | 0 | 1 | 1 |
| | | Cambiar los indicadores | 1 | 0 | 0 | 0 |
| | | Reactividad | 0 | 0 | 0 | 0 |
| | | Intercambio de conocimiento | 1 | 0 | 0 | 0 |
| **APLICABILIDAD** | ¿Cuándo un ambiente es favorable para usar este método? | Tamaño del proyecto | 0 | 1 | 0 | 1 |
| | | La complejidad del proyecto | 0 | 1 | 0 | 1 |
| | | Los riesgos del proyecto | 0 | 1 | 0 | 1 |
| | | El tamaño del equipo | 1 | 1 | 1 | 1 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | El grado de interacción con el cliente | 0 | 0 | 1 | 1 |
| | | Grado de interacción con los usuarios finales | 0 | 1 | 0 | 0 |
| | | Grado de interacción entre los miembros del equipo | 1 | 1 | 0 | 1 |
| | | Grado de integración de la novedad | 0 | 0 | 1 | 0 |
| | | La organización del equipo | 1 | 1 | 1 | 1 |
| **PROCESOS Y PRODUCTOS** | ¿Cómo esta caracterizados los procesos del método? | **Nivel de abstracción de las normas y directrices** | | | | |
| | | Gestión de proyectos | 0 | 1 | 0 | 1 |
| | | Descripción de procesos | 1 | 0 | 0 | 0 |
| | | Normas y orientaciones concretas sobre las actividades y productos | 1 | 0 | 0 | 0 |
| | | **Las actividades cubiertas por el método ágil** | | | | |
| | | Puesta en marcha del proyecto | 0 | 0 | 0 | 0 |
| | | Definición de requisitos | 1 | 1 | 0 | 1 |
| | | Modelado | 0 | 0 | 1 | 1 |
| | | Código | 1 | 1 | 1 | 1 |
| | | Pruebasunitarias | 1 | 1 | 1 | 1 |
| | | Pruebas de integración | 1 | 1 | 1 | 1 |
| | | Prueba del sistema | 1 | 1 | 1 | 1 |
| | | Prueba de aceptación | 0 | 0 | 0 | 0 |
| | | Control de calidad | 0 | 0 | 0 | 0 |
| | | Sistema de uso | 0 | 0 | 0 | 0 |
| | | **Productos de las actividades del método** | | | | |
| | | Modelos de diseño | 0 | 1 | 0 | 1 |
| | | Comentario del códigofuente | 1 | 1 | 1 | 1 |
| | | Ejecutable | 0 | 0 | 0 | 0 |
| | | Pruebasunitarias | 1 | 1 | 1 | 1 |
| | | Pruebas de integración | 1 | 1 | 1 | 1 |
| | | Pruebas de sistema | 1 | 0 | 1 | 1 |
| | | Pruebas de aceptación | 0 | 0 | 0 | 0 |
| | | Informes de calidad | 0 | 0 | 0 | 0 |
| | | Documentación de usuario | 0 | 0 | 0 | 0 |
| **TOTAL** | | | 28 | 29 | 28 | **32** |

En los resultados se observa que Scrumban ha obtenido mayor puntaje, por la cual sería una buena opción implementarla.

### 4.3. Implementación de la metodología ágil

Aunque la metodología ágil que más se aplica según la guía comparativa para el caso de la fábrica de software que estamos analizando es Scrumban, la fábrica ha optado por adoptar Kanban en uno de los proyectos que actualmente están realizando, con un equipo de trabajo de seis personas.

Se decidió implementar Kanban por una razón muy importante dentro de la fábrica de software. Se debe a que siendo una organización con tres años de vida y aunque tienen una organización jerárquica, la fábrica no trabaja cumpliendo roles jerárquicos específicos. Esto quiere decir que el líder de proyectos también es un desarrollador y los desarrolladores a la ausencia del líder asumen el cargo de líderes eventuales. Al implementar XP, Scrum o Scrumban como parte de la implementación será necesario cambiar la cultura actual del personal.

La fábrica de software en la actualidad no ha incursionado en ninguna herramienta ágil. Para este proyecto piloto todos los integrantes del equipo tienen conocimiento de que están participando en un proyecto piloto y todos están prestos a participar. En la figura 5 podemos ver la estructura actual del equipo de trabajo.

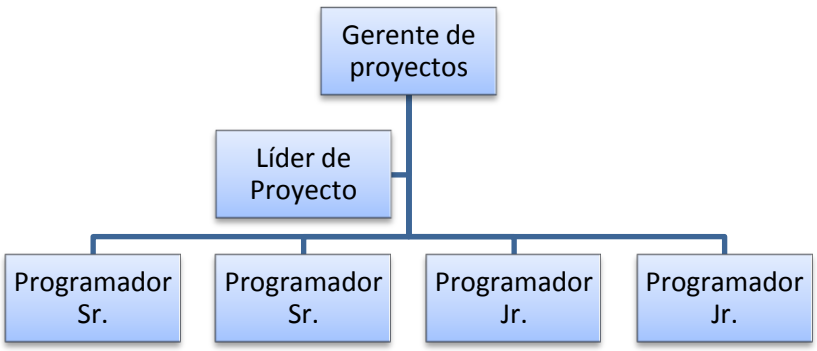



**Fig. 5.** Diseño organizacional de la fábrica de software

#### 4.3.1. Preparación

Para la preparación de la adopción de la metodología Kanban se llevaron a cabo las siguientes actividades:

1. **Capacitación en la metodología Kanban:** Esta actividad se realiza para que todo el equipo conozca el nuevo marco de trabajo y se familiarice.

2. **Creación del tablero de Kanban y la distribución de cada ciclo:** En esta actividad el grupo de trabajo elaboró el tablero Kanban, según la forma de trabajo en la fábrica de software. En la figura 6 podemos observar el trabajo final.

| DDS DERCAS | | DESARROLLO | | PRUEBAS INTERNAS | | PRUEBAS USUARIO | | PASE PRODUCCION | | VERIFICACION POST PRODUCCIOIN | | CAPACITACION MANUALES | | ACTIVIDADES PERMANENTES | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IN PROGRESS | DONE | IN PROGRESS | DONE | IN PROGRESS | DONE | IN PROGRESS | DONE | IN PROGRESS | DONE | IN PROGRESS | DONE | IN PROGRESS | DONE | IN PROGRESS | DONE |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | |

**Fig. 6.** Muro Kanban

3. **Diseño de los post-it:** En esta actividad, se llegó al acuerdo de que los post-it tendrían el siguiente diseño:

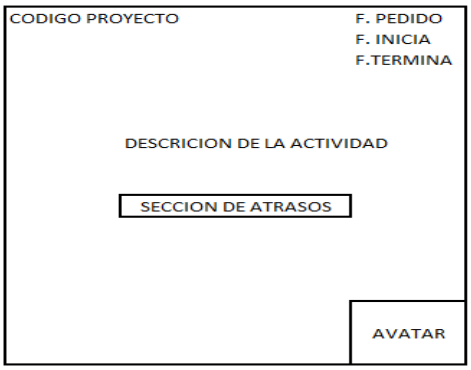



**Fig.7.**Post-it para el tablero Kanban

En la figura 7 observamos el diseño estructural de cómo se dividirá cada post-it, donde:

- Código Proyecto: Es el identificador del proyecto
- F. Pedido: Fecha en la que se realiza el pedido de la actividad. También se utiliza para la medición.
- F. Inicia: Fecha en la que se comienza a realizar la actividad. También se utiliza para la medición.

- F. Termina: Fecha en la que se finaliza la actividad. También se utiliza para la medición.
- Descripción de la actividad.
- Sección de atrasos: En este espacio se pondrá un punto rojo por cada día de atraso de la actividad, sirviendo como indicador para las revisiones diarias. La figura 9 nos muestra un ejemplo.
- Avatar: Sirve para identificar al integrante del equipo.

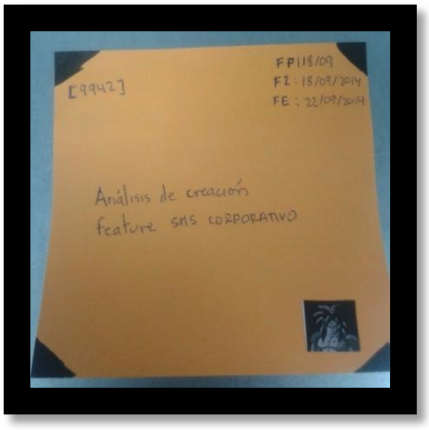



**Fig.8**. Post-itde trabajo para el tablero Kanban

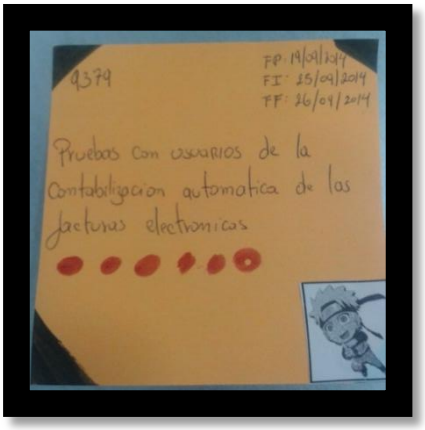



**Fig.9.**Post-it de trabajo para el tablero Kanban, reportando atrasos

**Creación de políticas internas:** En esta actividad se establecieron las siguientes políticas:

- El límite de las tareas en la pizarra será de 10.
- Las reuniones diarias serán de 20 minutos por reloj, terminado el tiempo se levanta la sesión y continuará el próximo día.
- El post-it que posea puntos rojos "señal de atraso", tendrá prioridad y se comentará al respecto en las reuniones diarias.

### 4.3.2. Actividades del equipo

En esta etapa del proyecto, cada integrante del equipo debe tener una lista de las actividades que se van a realizar y tenerla priorizadas, cada integrante del equipo debe auto-organizarse y realizar esta actividad. Una vez identificadas las actividades por parte de los integrantes del equipo, se procede a ir ubicando su actividad según la etapa o ciclo en la que se encuentre.

### 4.3.3. Reuniones de seguimiento

Parte importante de la ejecución del proyecto piloto fueron las reuniones diarias de 20 minutos, donde al principio las reuniones quedaban incompletas debido a que los 20 minutos no abastecían. Pero con el transcurso de los días esto fue mejorando y las reuniones pudieron terminar exactamente con los 20 minutos establecidos. En las reuniones se comentaban sobre los avances de cada integrante y los posibles inconvenientes que se presentaban y se ponía toda la atención a las actividades que presentaban puntos rojos "señal de atrasos".

Como resultado principal se puede decir que estas reuniones diarias, ayudaron a más de un integrante en sus inconvenientes, debido a que todo el equipo conocía los problemas de cada integrante, estos podían dar soluciones según su experiencia adquirida.

### 4.3.4. Medición de la productividad

Se puede realizar una medición tomando como referencia a cada integrante del equipo. Donde según los post-it finalizados podemos medir:

- Carga de trabajo.
- Actividades Otorgadas.
- Actividades Finalizadas con atrasos.

### Monitorizar

Para poder realizar estas mediciones es necesario revisar constantemente las actividades en el tablero he ir ingresando en un registro la información obtenida de cada pos-it.

Este trabajo utilizará estas mediciones y será desarrollado en la sección de análisis de resultados.

### 4.4 Resultados

Seleccionada la metodología ágil para la fábrica de software se demostrará que mediante la implementación de la metodología ágil Kanban, se puede mejorar la productividad de los equipos de trabajos en la fábrica de software, reduciendo tiempos de entregas y errores.

#### 4.4.1. Métricas del proyecto

**Carga de trabajo**
Para partir de un punto inicial se presentará la carga de trabajo de cada integrante del grupo, según los post-it pegados en el Backlog. La tabla 13 nos describe el número de actividades que cada integrante del equipo tiene en todo el proyecto.

**Tabla 13.** Actividades por integrantes

| Numero de Tareas | | | | |
|---|---|---|---|---|
| **Integrantes** | **Julio** | **Agosto** | **Septiembre** | **Octubre** |
| **Desarrollador 1** | 20 | 20 | 10 | 5 |
| **Desarrollador 3** | 10 | 5 | 20 | 6 |
| **Desarrollador 3** | 3 | 10 | 9 | 1 |
| **Desarrollador 4** | 6 | 3 | 1 | 2 |
| **Desarrollador 5** | 1 | 20 | 6 | 4 |
| **Desarrollador 6** | 4 | 5 | 3 | 6 |

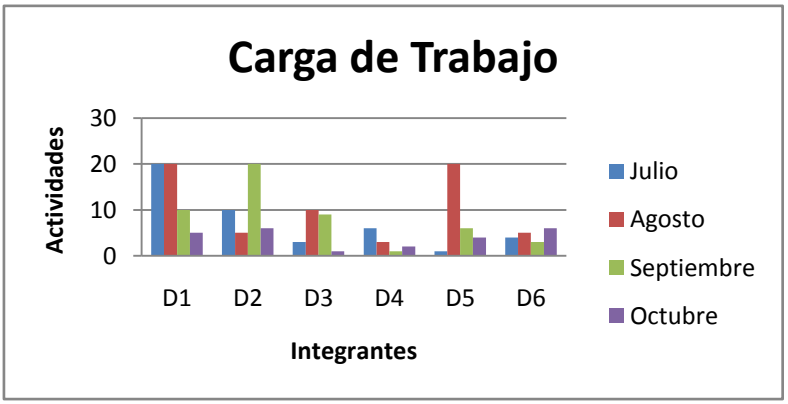



**Fig. 10.** Cargar de trabajo

En la figura 10 se observa la carga de trabajo de cada uno de los integrantes del equipo.

**Actividades**

Este indicador se calcula mediante las actividades entregadas por cada uno de los integrantes, comparando con las actividades que debían entregar en las fechas correspondientes.

**Tabla 14.** Actividades entregadas, agrupadas por mes.

| Julio | | Agosto | | Septiembre | | Octubre | |
|---|---|---|---|---|---|---|---|
| **Se espera** | **Se entrego** | **Se espera** | **Se entrego** | **Se espera** | **Se entrego** | **Se espera** | **Se entrego** |
| 20 | 6 | 20 | 25 | 10 | 18 | 5 | 6 |
| 10 | 8 | 5 | 5 | 20 | 22 | 6 | 6 |
| 3 | 2 | 10 | 9 | 9 | 10 | 2 | 2 |
| 6 | 6 | 3 | 3 | 1 | 1 | 1 | 2 |
| 1 | 1 | 20 | 18 | 6 | 8 | 4 | 4 |
| 4 | 2 | 5 | 4 | 3 | 6 | 6 | 6 |

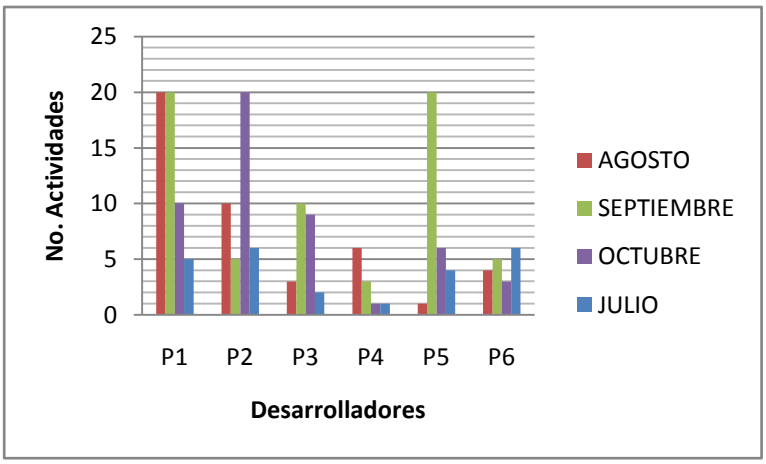



**Fig. 11.** Actividades por entregar

En la Figura 11 se observa la cantidad de actividades que el desarrollador debe entregar en el mes.

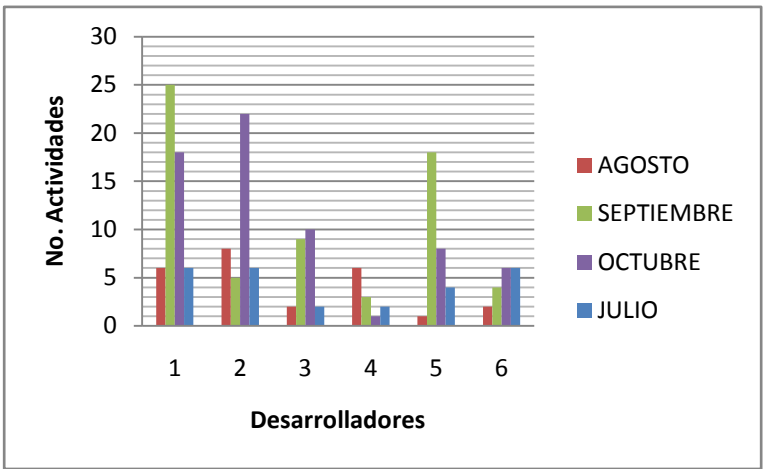



**Fig. 12.**Actividades entregadas

En la Figura 12 se observa la cantidad de actividades entregadas por el desarrollador en el mes.

**Atrasos**

Este indicador se calcula mediante el registro de atraso registrado de las actividades en cada mes.

**Tabla 15.**Atrasos registrados, agrupados por mes.

| Julio | Agosto | Septiembre | Octubre |
|---|---|---|---|
| **4** | 3 | 0 | 0 |

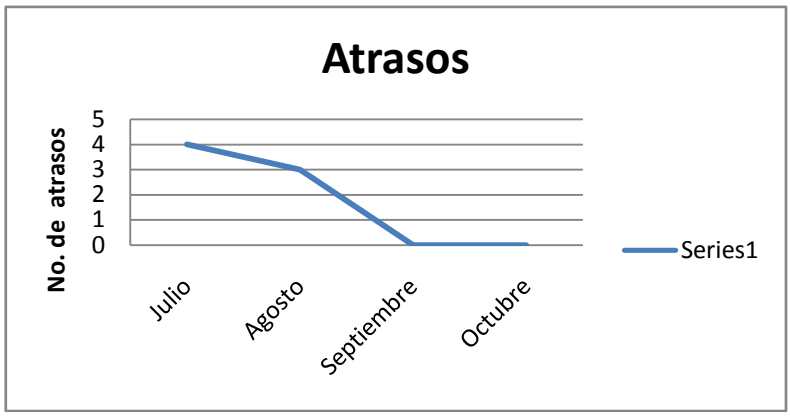



**Fig. 13.** Atrasos por mes

En la Figura 13 se observa cómo van disminuyendo los atrasos de las actividades. En es transcurso de los meses.

Para brindar un soporte adicional al estudio, se ha realizado una encuesta final a los integrantes del equipo del proyecto piloto, después del cierre del proyecto. El objetivo es poder medir la satisfacción de equipo entre la metodología ágil implementada en comparación con el método de gestión tradicional que venían implementando. El modelo de la encuesta está disponible en el Anexo 2.

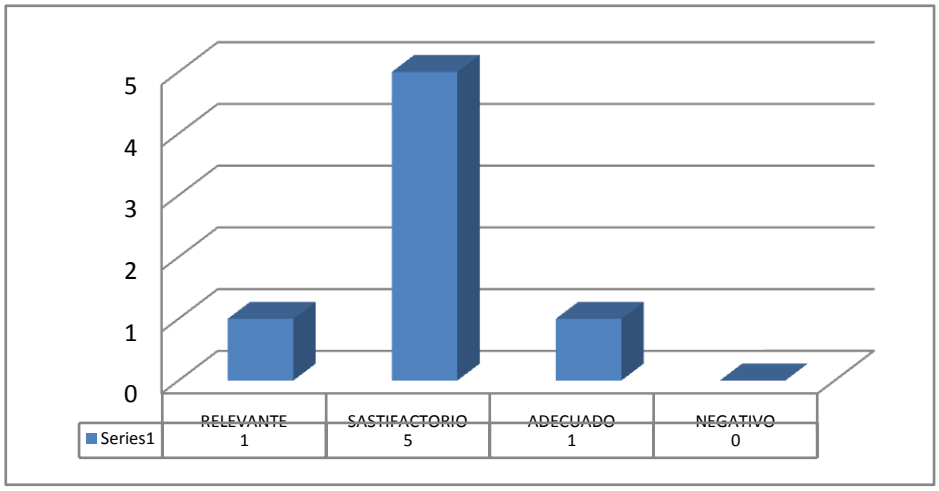



**Fig. 14.**Resultado de usar Kanban para la entrega del producto en menor tiempo

En la Figura 14 se observa el resultado de satisfacción del equipo de trabajo al implementar Kanban, referente a la entrega del producto.

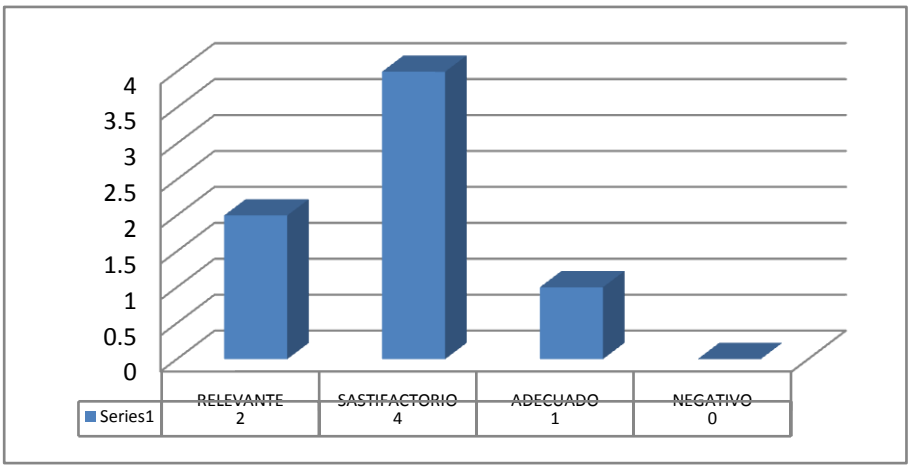



**Fig. 15.**Impacto en la productividad del equipo al usar Kanban

En la Figura 15 se observa el resultado de satisfacción del equipo de trabajo al implementar Kanban, referente al nivel de productividad del equipo.

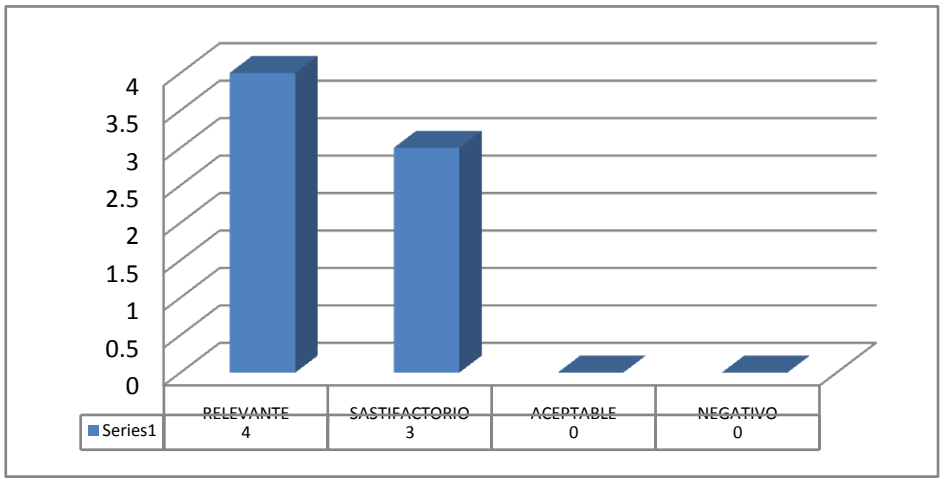



**Fig. 16.** Mejora de la calidad del producto final al usar Kanban

En la Figura 16 se observa el resultado de satisfacción del equipo de trabajo al implementar Kanban, referente a la calidad del producto final.

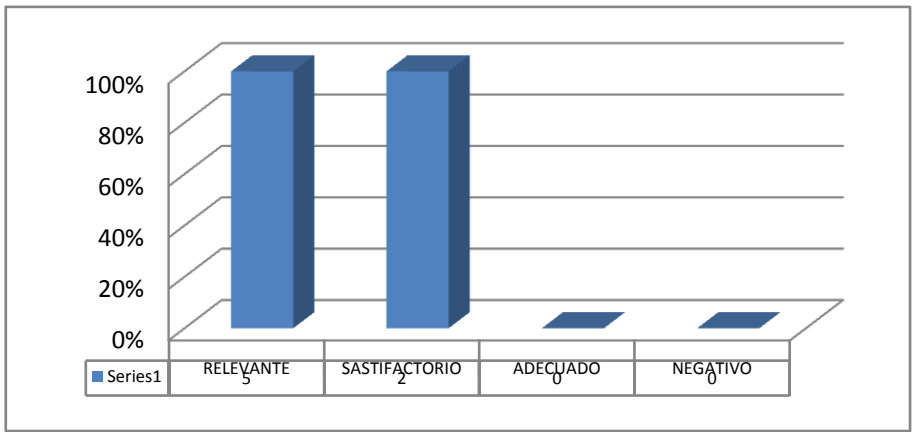



**Fig.17.** Resultados obtenidos al usar Kanban en el proyecto

En la Figura 17 se observa el resultado de satisfacción del equipo de trabajo al implementar Kanban.

## 5. Conclusiones

El objetivo de este trabajo consistió en poder incrementar la productividad de la fábrica de software. Para esto en este artículo, se propusieron cuatro metodologías ágiles y se seleccionó una de ellas, para aplicarla en un proyecto piloto con el fin de poder medir los tiempos de entregas de cada actividad y el tiempo de entrega final del producto. Con la adopción de esta metodología se logró cumplir con el objetivo, demostrando en la sección de análisis y resultados los siguientes puntos:

- Efectividad: Entregando al cliente el producto final con días de anticipación y entregando cada actividad en las fechas acordadas según los cronogramas de trabajos.
- Productividad: Esto tiene que ver con el desempeño del equipo de trabajo, quienes mostraron interés y motivación en este proyecto piloto, dando como resultado la efectividad del trabajo.
- Satisfacción del cliente: Este es el punto más relevante en este proyecto, debido a que la fábrica de software en los otros proyectos siempre ha terminado entregado el producto al cliente con días de atrasos. En este proyecto piloto el cliente quedo satisfecho con el trabajo por el cumplimiento de las fechas establecidas.

En este estudio se demuestra que usar una metodología ágil como Kanban, para la gestión de un proyecto de software es una opción viable y sobre todo innovadora, que ofrece beneficios a las organizaciones que las implementan. También es una nueva

forma de salir de la rutina de programación y ayuda a la motivación de los integrantes de los equipos de trabajo. Por otro lado también se demuestra que al aplicar una metodología ágil como Kanban, la fábrica de software experimentó un aumento en su productividad, entregando el producto final en los tiempos acordados de cierre de proyecto, entre la fábrica y el cliente. Por lo que podemos decir que la implementación de la metodología ágil Kanban aumentó la productividad de la fábrica de software, en referencia a los tiempos de entrega.

Se ha realizado este estudio con el objetivo de poder brindar una solución a la gestión de proyectos de la fábrica de software, donde se le proporcionó una base y experiencia en la implementación, para que puedan aplicar a los demás proyectos y mejorar su calificación como organización por parte de sus clientes. A su vez este trabajo pretende ser una referencia práctica, para otras organizaciones, sirviendo como apoyo a la hora de comenzar a trabajar de manera ágil.

Es necesario tener en cuenta que una metodología ágil cualquiera que sea no nos va a garantizar el éxito o fracaso del proyecto, sino la predeterminación y el cambio en la cultura de las personas.

## 6. Limitaciones

La fábrica de software tiene un contrato con el cliente de dar soporte hasta seis meses después de la entrega del producto final, por lo que no se puede presentar un porcentaje exacto de la calidad del producto, pero si se puede estimar, debido a cada éxito en las pruebas con el cliente.

## 7. Recomendaciones

En este estudio no se implementaron todas las prácticas de adaptación de Kanban por cuestión de tiempo, por lo que se recomienda realizarlas todas para una mejor apreciación de los resultados.

De los varios proyectos de trabajos se seleccionó uno para el plan piloto, considerando el tamaño, duración y nivel de experiencia del equipo de trabajo.

## 8. Agradecimientos

Este trabajo forma parte del trabajo de titulación presentado a la UEES. Se agradece a la Fábrica de Software, por haber contribuido de maneta participatica, con este proyecto.

## 9. Referencias


Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (2002). Agile Software Development Methods. Review and Analysis. *VTT* .

Ágil, M. (2011). *Manifiesto Agil.* Recuperado el 1 de 10 de 2014, de http://agilemanifesto.org/iso/es/

Ahmad Khan, Z. (23 de Mayo de 2014). Scrumban – Adaptive Agile Development Process (Tésis de Maestría). University of Applied Sciences Metropolia.

Anderson, D. J. (2010). *Kanban Successful Evolutionary Change for Technology Organizations.* Sequin, Washington.

Bohem, B. (12 de diciembre de 1976). Software Engineering. *IEEE Transaction on Computers, C-25, num 12* .

Bourque, P., & Fairley, R. E. (Edits.). *Guide to the Software Engineering Body of Knowledge, Versión 3.0, IEES Computer Society, 2014.*

Canós, J. H., Letelier, P., & Penadés, M. C. (15 de Enero de 2006). *Métodologías Ágiles en el Desarrollo de Software.* Recuperado el 15 de Agosto de 2014, de http://noqualityinside.com/nqi/nqifiles/XP_Agil.pdf

Hernández Sampieri, R., Ferández Collado, C., & Baptista Lucio, P. (2006). *Metodología de la Investigación* (Cuarta ed.). Mexico: McGraw-Hill.

Joskowicz, J. (10 de febrero de 2008). *Instituto de Ingeniería Eléctrica.* Recuperado el 5 de Octubre de 2014, de Reglas y Prácticas en extreme programming: http://iie.fing.edu.uy/~josej/docs/XP%20-%20Jose%20Joskowicz.pdf

Ken Schwaber and Jeff Sutherland. (Julio de 2013). *Scrum Guides.* Recuperado el 15 de Septiembre de 2014, de http://www.scrumguides.org/scrum-guide.html

Kniberg, H., & Skarin, M. (2010). *Kanban and Scrum making the most of both* (On line ed.). (D. Plesa, Ed.)

Mendes Calo, K., Estevez, E., & Fillottrani, P. (2010). *Evaluación de Metodologías Ágiles para Desarrollo de Software.* Recuperado el 15 de Octubre de 2014, de http://sedici.unlp.edu.ar/bitstream/handle/10915/19546/Documento_completo.pdf?sequence=1

ORDYSIŃSKI, T. (2013). KANBAN BASED INFORMATION MANAGEMENT IN ORGANIZATION. Studia I Materialy Polskiego Stowarzyszenia Zarzadzania Wiedza / Studies & Proceedings Polish Association For Knowledge Management, (63), 76-85.

Pérez, M. (2012). *uva biblioteca.* Recuperado el 29 de Agosto de 2014, de Guía Comparativa de Metodologías Ágiles: http://uvadoc.uva.es/handle/10324/1495

Poppendieck, M., & Poppendieck, T. (2003). *Lean Software Development: An Agile Toolkit for Software Development.* Addison Wesley.

Porter, M. E. (2014). Competitive Advantage.

Sommerville, I. (2005). *Ingeniería de Software* (Séptima Edición ed.). Madrid: Pearson Education, S.A.

Tamayo y Tamayo, M. (2001). *El proceso de la investigación científica.* Editorial Limusa.

Villareal, F. D. (1 de Agosto de 2013). Adopción de una metodología ágil para proyectos de software (Tésis de Maestría). Universidad Europea Miguel de Cervantes. México.

Weitzenfeld, A. (2005). *Ingenieria de Software Orientada a Objetos: Teoría y Práctica con UML y Java.* Thomson Paraninfo.

Wells, D. (8 de Octubre de 2013). *Extreme Programming*. Recuperado el 15 de Agosto de 2014, de http://www.extremeprogramming.org/

## 10. Anexos

8.1. Guía comparativa

| METODOLOGIAS ÁGILES | | | | | | |
|---|---|---|---|---|---|---|
| | | | **ORIENTADA AL DESARROLLO DEL SOFTWARE** | **ORIENTADA A LA GESTION DEL PROYECTO** | | |
| | | | **XP** | **SCRUM** | **KANBAN** | **SCRUMBAN** |
| **USO** | ¿Por qué utilizar una metodología ágil? | Respecto a las fechas de entrega | FALSO | VERDADERO | FALSO | FALSO |
| | | Cumplimiento de los requisitos | VERDADERO | VERDADERO | VERDADERO | VERDADERO |
| | | Respecto al nivel de calidad | FALSO | FALSO | FALSO | FALSO |
| | | Satisfacción del usuario | FALSO | VERDADERO | FALSO | FALSO |
| | | Entornosturbulentos | VERDADERO | VERDADERO | VERDADERO | VERDADERO |
| | | Favorable al Off shoring | FALSO | VERDADERO | FALSO | VERDADERO |
| | | Aumento de la productividad | VERDADERO | VERDADERO | VERDADERO | VERDADERO |
| **CAPACIDAD DE AGILIDAD** | ¿Cuál es la parte de agilidad incluida en el método? | Iteracionescortas | VERDADERO | VERDADERO | VERDADERO | VERDADERO |
| | | Colaboración | VERDADERO | VERDADERO | VERDADERO | VERDADERO |
| | | Centrado en las personas | VERDADERO | VERDADERO | VERDADERO | VERDADERO |
| | | Refactoring político | VERDADERO | FALSO | FALSO | FALSO |
| | | Pruebapolítico | VERDADERO | VERDADERO | FALSO | VERDADERO |
| | | Integración de los cambios | VERDADERO | VERDADERO | VERDADERO | VERDADERO |
| | | De peso ligero | VERDADERO | VERDADERO | VERDADERO | VERDADERO |
| | | Los requisitos funcionales pueden cambiar | VERDADERO | VERDADERO | VERDADERO | VERDADERO |
| | | Los requisitos no funcionales pueden cambiar | FALSO | FALSO | VERDADERO | VERDADERO |
| | | El plan de trabajo puede cambiar | VERDADERO | FALSO | VERDADERO | VERDADERO |
| | | Los recursos humanos pueden cambiar | VERDADERO | FALSO | VERDADERO | VERDADERO |
| | | Cambiar los indicadores | VERDADERO | FALSO | FALSO | FALSO |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | Reactividad (AL COMIENZO DEL PROYECTO, CADA ETAPA, CADA INTERACIÓN) | INTERACIÓN | INTERACIÓN | INTERACIÓN | INTERACIÓN |
| | | Intercambio de conocimiento (BAJO, ALTO) | ALTO | BAJO | BAJO | BAJO |
| **APLICABILIDAD** | ¿Cuándo un ambiente es favorable para usar este método? | Tamaño del proyecto (PEQUEÑO, GRANDE | PEQUEÑO | GRANDE / PEQUEÑO | PEQUEÑO | GRANDE / PEQUEÑO |
| | | La complejidad del proyecto (BAJA, ALTA) | BAJA | ALTA | BAJA | ALTA |
| | | Los riesgos del proyecto (BAJO, ALTO) | BAJO | ALTO | BAJO | ALTO |
| | | El tamaño del equipo (PEQUEÑO, GRANDE) | PEQUEÑO | PEQUEÑO | PEQUEÑO | PEQUEÑO |
| | | El grado de interacción con el cliente (BAJA, ALTA) | ALTA | ALTA | BAJO | BAJO |
| | | Grado de interacción con los usuarios finales (BAJA, ALTA) | BAJO | ALTO | BAJO | BAJA |
| | | Grado de interacción entre los miembros del equipo (BAJA, ALTA) | ALTA | ALTA | BAJA | ALTA |
| | | Grado de integración de la novedad (BAJA, ALTA) | ALTA | ALTA | BAJA | ALTA |
| | | La organización del equipo (AUTO-ORGANIZACIÓN, ORGANIZACIÓN JERÁRQUICA) | AUTO - ORGANIZACIÓN | AUTO - ORGANIZACIÓN | AUTO - ORGANIZACIÓN | AUTO - ORGANIZACIÓN |
| **PROCESOS Y PRODUCTOS** | ¿Cómo esta caracterizados los procesos del método? | **Nivel de abstracción de las normas y directrices** | | | | |
| | | Gestión de proyectos | FALSO | VERDADERO | FALSO | VERDADERO |
| | | Descripción de procesos | VERDADERO | FALSO | FALSO | FALSO |
| | | Normas y orientaciones concretas sobre las actividades y productos | VERDADERO | FALSO | FALSO | FALSO |
| | | **Las actividades cubiertas por el método ágil** | | | | |
| | | Puesta en marcha del proyecto | FALSO | FALSO | FALSO | FALSO |
| | | Definición de requisitos | VERDADERO | VERDADERO | FALSO | VERDADERO |
| | | Modelado | VERDADERO | VERDADERO | FALSO | FALSO |
| | | Código | VERDADERO | VERDADERO | VERDADERO | VERDADERO |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | Pruebasunitarias | VERDADERO | VERDADERO | VERDADERO | VERDADERO |
| | | Pruebas de integración | VERDADERO | VERDADERO | VERDADERO | VERDADERO |
| | | Prueba del sistema | VERDADERO | VERDADERO | VERDADERO | VERDADERO |
| | | Prueba de aceptación | FALSO | FALSO | FALSO | FALSO |
| | | Control de calidad | FALSO | FALSO | FALSO | FALSO |
| | | Sistema de uso | FALSO | FALSO | FALSO | FALSO |
| | | **Productos de las actividades del método** | | | | |
| | | Modelos de diseño | FALSO | VERDADERO | FALSO | VERDADERO |
| | | Comentario del códigofuente | VERDADERO | VERDADERO | VERDADERO | VERDADERO |
| | | Ejecutable | VERDADERO | VERDADERO | VERDADERO | VERDADERO |
| | | Pruebasunitarias | VERDADERO | VERDADERO | VERDADERO | VERDADERO |
| | | Pruebas de integración | VERDADERO | VERDADERO | VERDADERO | VERDADERO |
| | | Pruebas de sistema | VERDADERO | FALSO | VERDADERO | VERDADERO |
| | | Pruebas de aceptación | FALSO | FALSO | FALSO | FALSO |
| | | Informes de calidad | FALSO | FALSO | FALSO | FALSO |
| | | Documentación de usuario | FALSO | FALSO | FALSO | FALSO |

Fuente: María Pérez, 2012

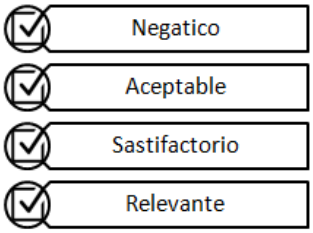
### 8.2 Anexo 2: Encuesta satisfacción de la metodología.

**Encuesta**

1. ¿Cuál ha sido el resultado de usar Kanban para entrega del producto en menor tiempo?


2. ¿Cuál ha sido el impacto en la productividad del equipo al usar Kanban?


3. ¿En qué grado se mejoró la calidad del producto final en el uso de Kanban como proceso?




4. ¿Cómo califica los resultados obtenidos de usar Kanban?