UNIVERSIDAD TECNOLÓGICA ISRAEL

TRABAJO DE TITULACIÓN EN OPCIÓN AL GRADO DE:

INGENIERO EN SISTEMAS

TEMA: “ESTUDIO DE LA EFECTIVIDAD DE LA APLICACIÓN DE LA METODOLOGÍA ÁGIL DE DESARROLLO SCRUM”

AUTOR/ A: CHRISTIAN FELIPE REY GUEVARA

TUTOR: MSC. OSWALDO BASURTO

AÑO: 2017

UNIVERSIDAD TECNOLÓGICA ISRAEL

PLAN DEL PROYECTO INTEGRADOR DE CARRERA

CARRERA / PROGRAMA:	Ingeniería en Sistemas Informáticos
AUTOR:	Rey Guevara Christian Felipe
TEMA DEL TT:	**Estudio de la efectividad de la aplicación de la metodología ágil de desarrollo Scrum**
ARTICULACIÓN CON LA LÍNEA DE INVESTIGACIÓN INSTITUCIONAL:	Tecnología aplicada a la producción y sociedad
SUBLÍNEA DE INVESTIGACIÓN INSTITUCIONAL:	Desarrollo y automatización de procesos
FECHA DE PRESENTACIÓN DEL PLAN:	30 de noviembre del 2016

DECLARACIÓN Y AUTORIZACIÓN

Yo, **Christian Felipe Rey Guevara**, CI: 1713293015 autor del trabajo de graduación: **Estudio de la efectividad de la aplicación de las Metodología ágil de desarrollo Scrum**, previo a la obtención del título de **Ingeniería en Sistemas Informáticos** en la UNIVERSIDAD TECNOLÓGICA ISRAEL.

1.- Declaro tener pleno conocimiento de la obligación que tienen las instituciones de educación superior, de conformidad con el Articulo 144 de la Ley Orgánica de Educación Superior, de difundir el respectivo trabajo de graduación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la SENESCYT a tener una copia del referido trabajo de graduación, con el propósito de generar un repositorio que democratice la información, respetando las políticas de propiedad intelectual vigentes.

Quito, enero del 2017

Atentamente.

Christian Felipe Rey Guevara.
C.I. 1713293015

CARTA DE TUTORÍA

En mi calidad de Tutor del Trabajo de Graduación certifico:

Que el trabajo de graduación "**ESTUDIO DE LA EFECTIVIDAD DE LA APLICACIÓN DE LA METODOLOGÍA ÁGILE DE DESARROLLO SCRUM"**, presentado por **CHRISTIAN FELIPE REY GUEVARA,** estudiante de la carrera de INGENIERÍA **EN SISTEMAS INFORMÁTICOS**, reúne los requisitos y méritos suficientes para ser sometido a la evaluación del Tribunal de Grado, que se designe, para su correspondiente estudio y calificación.

Quito, diciembre del 2016

TUTOR

Msc. Oswaldo Basurto

DEDICATORIA

A Dios

A mi Esposa por ser Fuente de Inspiración y apoyo constante

Al Msc. Oswaldo Basurto por su orientación académica y profesional para el desarrollo y culminación del presente trabajo.

AGRADECIMIENTO

Deseo agradecer a Dios
A mi Esposa por su comprensión y apoyo constante.
A mis profesores por sus conocimientos que recibí en las aulas de clase.
A mi Director de tesis Msc. Oswaldo Basurto por su guía profesional en la elaboración de este documento.

Con estima y aprecio
Christian.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

En Ecuador la mayor parte de las empresas pymes de desarrollo de software lamentablemente no cuentan con una metodología de desarrollo de software que permita entregar un producto de software confiable y flexible con estándares de calidad. Es así, que al no adoptar metodologías claras para el desarrollo de software lo que promueven es que exista frecuentemente problemas de conceptualización y diseño al instante de realizar programas, causando no solamente fallas en el producto que se entrega, sino también, desgastando al equipo de trabajo y generando los respectivos costos económicos y financieros en la empresa.

Por tal motivo se propone exponer las presentes metodologías de desarrollo ágil como una alternativa eficiente y eficaz para desarrollar software especialmente a pequeñas empresas de software donde los equipos de trabajo son pequeños y los costos de operación para una pyme son elevados, brindado para ello procedimientos esenciales en el inicio y final del ciclo de desarrollo de software.

Descriptores: Desarrollo de software, metodología, desarrollo ágil, calidad.

ABSTRACT

In Ecuador, the majority of SMEs software development companies unfortunately do not have a software development methodology that allows to deliver a product of reliable and flexible software quality standards. Thus, by not adopting clear methodologies for software development as they promote is that problems of conceptualization and design there often when making programs, causing not only faults in the product delivered, but also, wearing team work and generating the respective economic and financial costs in the company.

Therefore, it is proposed to explain this agile development methodology as an efficient and effective way to develop software especially small software companies where teams are small and operating costs for an SME alternative are high, provided for this procedure essential in the beginning and end of the software development cycle.

Descriptors: Software development, methodology, agile software development, quality.

CAPÍTULO 1

1. INTRODUCCIÓN

Toda empresa de desarrollo de software enfrenta en cada uno de sus proyectos desafíos tecnológicos, organizacionales y estructurales; el éxito o fracaso de cada proyecto está fundamentalmente basado en la metodología de desarrollo seguido por la pericia del talento humano involucrado en el proyecto, su tecnología, estructura de organización y como esta organización responde antes los cambios y desafíos ante la demanda (Group, 2012).

A nivel mundial el 89% de la tecnología de información y comunicación se encuentra enfocada a desarrollar software de servicios con una proyección de ingresos al 2015 de 36,6 mil millones de dólares (DATAMONITOR, 2014).

En Ecuador existen aproximadamente 633 empresas, a nivel nacional que participan en el sector de 'Programación, consultoría y actividades conexas'. De éstas 610 empresas se dedican a la adaptación de programas informáticos a las necesidades del cliente y el resto en aéreas involucradas a la recuperación de desastres informáticos e instalación de programas informáticos (AESOFT, Estudio de mercado de hardware y software en Ecuador, 2012).

El 80% se encuentra ubicado en las principales ciudades de Quito y Guayaquil con ingresos totales 550 millones de dólares según el Servicio de Rentas Internas (SRI). El valor de los ingresos en los próximos cuatro años en el sector reporta una tasa de crecimiento del 5% en el área de desarrollo de software (Services, 2010). De estos únicamente el 35% de las empresas desarrolla software libre en cambio el 65% es desarrollado con licenciamiento por terceras empresas (AESOFT, Estudio de mercado de hardware y software en Ecuador, 2012).

1.1. Problema

Un gran número de pymes tecnológicas ecuatorianas de desarrollo de software lamentablemente no cuentan con metodologías de desarrollo de software que les facilite entregar un software confiable y flexible con estándares de calidad. Es así, que al no adoptar metodologías claras para el desarrollo de software lo que se promueve es que exista frecuentemente problemas de diseño, conceptualización y arquitectura en sus sistemas, causando no solamente fallas en el producto que se entrega, sino también, desgastando al equipo de trabajo y generando los respectivos costos económicos y financieros en la empresa.

Muy pocas de éstas son las que utilizan un adecuado manejo en el proceso de metodología de desarrollo de software. La mayoría delegan al jefe de sistemas o que realice este tipo de desarrollo o en su defecto a la persona con más experiencia. La práctica habitual es hacer que el jefe a cargo de este proyecto asuma toda la responsabilidad y la presión para que el proyecto siga su curso. Este tipo de prácticas sin una metodología de desarrollo se la conoce como desarrollo heroico donde el gestor y el programador utilizan sus propias codificaciones y su propio empirismo de desarrollos anteriores para hacer frente al proyecto asignado. Este tipo de proyecto trae como consecuencia una deficiente planificación, mal seguimiento en el desarrollo, falta de liderazgo, alta presión en el grupo de desarrollo y por consecuencia retrasos o no satisfacción del sistema al cliente, ocasionando pérdidas económicas y financieras en la empresa.

1.2. Objetivos

1.2.1. Objetivo General:

Comparar la eficiencia y efectividad en los proyectos de desarrollo de software utilizando metodologías de desarrollo ágiles.

1.2.2. Objetivos Específicos:

- Comparar los procesos que integra un proyecto de desarrollo de software ágil
- Aplicar un sistema de pruebas para el desarrollo de las metodologías ágiles
- Presentar los resultados realizados
- Documentar estas metodologías para desarrollo de futuros proyectos
- Aplicar las metodologías de desarrollos ágiles a cuatro talleres de actualización como: buenas prácticas de programación, java avanzado, metodología Scrum y metodología PMP.

1.3. Ideas a sostener en el proceso investigativo

- La presente investigación generará la eficiencia y eficacia en el desarrollo de proyectos de software

- La presente investigación no generará la eficiencia y eficacia en el desarrollo de proyectos de software

1.4. Alcance

El presente proyecto se limita a encontrar una solución alternativa y eficiente para pymes de desarrollo de software que busquen una solución flexible, el presente estudio no pretende realizar paralelos y métricas de otras metodologías de desarrollo que utilizan la mayoría de industrias de software, pero si enfatiza los aspectos principales del uso de estas herramientas, sus buenas prácticas de software, procesos y organización del talento humano enfocado a realizar las comparaciones entre estas metodologías y donde se pueda extraer las mejores experiencias.

CAPÍTULO 2

2. MARCO TEÓRICO

2.1. Manifiesto de desarrollo ágil

El desarrollo de software ha evolucionado tanto en los últimos tiempos demandando en la construcción mucha pericia técnica en su desarrollo, llegando a convertirse en un elemento estratégico competitivo clave a nivel mundial y un factor dominante en las economías del mundo industrializado.

El desarrollo ágil de software es ahora considerado como una especialización técnica de alta demanda pues un programador con este tipo de destreza técnica a más de saber programar es además un conceptualizador estratégico de todo el sistema informático a su cargo.

Según el manifiesto de los creadores de estas metodologías lo enuncian de esta manera:

“Descubrir formas mejores de desarrollar software tanto por nuestra propia experiencia como ayudando a terceros. A través de este trabajo hemos aprendido a valorar:
Individuos e interacciones sobre procesos y herramientas.
Software funcionando sobre documentación extensiva.
Colaboración con el cliente sobre negociación contractual.
Respuesta ante el cambio sobre seguir un plan.
Esto es, aunque valoremos los elementos de la derecha, valoramos más los de la izquierda” (Sutherland et at, 2001).

Por lo enunciado anteriormente, las personas más influyentes en desarrollo de software establecieron doce principios de cómo se debe guiar el desarrollo de software, estos son:

“1. Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.
2. Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos Ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente.
3. Entregamos software funcional frecuentemente, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible.
4. Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.
5. Los proyectos se desarrollan en torno a individuos motivados. Hay que darles el entorno y el apoyo que necesitan, y confiarles la ejecución del trabajo.
6. El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara.
7. El software funcionando es la medida principal de progreso.
8.Los procesos Ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios debemos ser capaces de mantener un ritmo constante de forma indefinida.
9. La atención continua a la excelencia técnica y al buen diseño mejora la Agilidad.
10. La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.
11. Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados.

12. A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para a continuación ajustar y perfeccionar su comportamiento en consecuencia" (Sutherland et at, 2001).

Por lo que podemos decir que las metodologías de desarrollo ágil de software son formas eficientes y eficaces de desarrollar software en un entorno organizativo que enfrenta el cambio con relativa simplicidad a un menor costo.

2.2. Metodologías de desarrollo ágil

La metodología de desarrollo ágil de software nació a partir en la década de los '90 intentando obtener software de calidad optimizando tiempo y costo en su desarrollo.

La solución partió con pequeños grupos de programadores con herramientas que generaban código automático, sintaxis de alto nivel y una adecuada organización en cada una de sus fases. A más de utilizar conceptos e ideas ya presentadas por IBM de la década del '70 en su libro Chief Programmer este tipo de "hacer las cosas" dio origen a una nueva forma de gestión como la metodología XP (eXtreme Programming) arquetipo para la evolución de futuras metodologías ágiles.

Entre las metodologías ágiles de desarrollo destacadas se encuentran:

- XP (eXtreme Programming)
- XBreed
- Cristal Clear
- ASD (Adaptive Software Development)
- Scrum

XP (Extreme Programming)

Programación extrema busca adaptarse a las incertidumbres de la gestión de proyectos en el desarrollo de software en el inicio y fin de un plan propuesto. Se considera que los cambios no previstos son naturales e inevitables en todo proyecto.

Su creador Kent Beck, dio origen a la primera metodología de desarrollo ágil en su libro Extreme Programming Explained: Embrace Change (1999) detallando "un conjunto de prácticas y procesos de desarrollo liviano de software para enfrentar ambientes muy cambiantes en donde la planificación, el desarrollo y el diseño van junto de la mano en cada fase de cambio" (Beck, 2004). Para ello un equipo pequeño motivado con capacidad de aprender y con una alta capacitación es capaz de enfrentarse a entornos complejos en corto tiempo aportando soluciones simples a un menor costo.

XBreed

Este tipo de metodología es la fusión entre la Programación Extrema y la metodología Scrum, utiliza Scrum para la gestión de software y para la adaptación de las versiones utiliza XP. Su objetivo es desarrollar software reutilizable en tiempo record creando bibliotecas que pueden ser fácilmente insertadas en nuevos proyectos de software, el proceso de desarrollo depende del equipo un alto conocimiento y capacidad para mantener el ciclo del proyecto.

Crystal Clear

La presente metodología establece dos dimensiones en su desarrollo tamaño y complejidad centrados en ejes como: aspecto, tamaño del equipo, comunicación, políticas a seguir, espacio físico de trabajo. Se enfatiza mucho en la comunicación manejando iteraciones cortas en cada iteración y feedback para los usuarios/clientes.

La metodología es pragmática y se orienta a lo que se puede customizar durante el proceso, los desarrolladores escogen aquellos principios básicos que les resultan efectivos, lo agregan o lo remueven a los procesos de desarrollo de software.

SCRUM

Se conoce al método Scrum como “un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos.” (Agiles, 2014)

ASD (Adaptive Software Development)

Es un modelo que toma conceptos del ciclo de la calidad de Deming donde el Comando /Control es reemplazado por un modelo Liderazgo/ Colaboración y de inteligencia artificial sobre Sistemas Adaptativos Complejos donde enfatiza que la dificultad de la estructura de software es por naturaleza complejo.

Esta metodología propone administrar proyectos mediante tres ejes: especulación, colaboración y aprendizaje estos procesos generan un alto cambio y rápido desarrollo en procesos que se encuentren al borde del caos.

2.3. Metodología de desarrollo Scrum

Según el método Scrum se orienta en que todo proyecto desde su fase inicial hasta su entrega final es imposible prever todos los cambios a realizarse, para ello Scrum advierte que este tipo de cambios son naturales y obedece a la dinámica de evolución de un proyecto que está sujeto a variables que son a veces imposibles de controlar.

La bondad de **Scrum** es que facilita la formación de equipos de desarrollo organizados impulsados por un liderazgo y una visión del proyecto que mediante la interdependencia y la comunicación informal entre todos los involucrados puedan afrontar los cambios e imprevistos originados en su gestión.

La característica clase de Scrum es como el cliente influye constantemente en la idea y en los cambios en el proyecto sobre lo que quieren y lo que necesitan, de esta manera se estrechan los vínculos de trabajo y hace más predecible la entrega de un producto confiable y de calidad. Es por esta razón que Scrum posee una metodología pragmática asumiendo como primera instancia que la magnitud del problema no puede ser completamente definido ni entendido a lo cual debe responder de manera rápida y flexible los requisitos emergentes.
Las implementaciones para gestionar Scrum son diversas que van desde pequeñas notas amarillas "post-it", pizarras de actividades y paquetes de software
Una ventaja de este tipo de metodología es su fácil aprendizaje y se requiere ser auto disciplinado para beneficiarse de ello

2.3.1. Origen del Método Scrum

En cuanto al origen del método Scrum, nace en marzo de 2001, y reúne a 17 críticos de los modelos de producción basados en procesos, para tratar a profundidad el término “Métodos Ágiles” y definir aquellas que estaban emergiendo como una alternativa de las metodologías tradicionales: “CMM-SW, (precursor de CMMI) PMI, SPICE (proyecto inicial de ISO 15504)” (Scrum, 2014), consideradas convencionales y rigurosas por su naturaleza normativa y de una acentuada dependencia en planificaciones al detalle , antes de su desarrollo.

Estos integrantes se reunieron y formaron cuatro postulados que ha dado forma al “Manifiesto Ágil” (Sutherland et at, 2001) que son los valores donde se sustenta estos métodos.

“Hasta 2005, entre los defensores de los modelos de procesos y los de modelos ágiles fueron frecuentes las posturas radicales, más ocupadas en descalificar al otro, que en estudiar sus métodos y conocerlos para mejorar los propios” (Scrum, 2014).

Dentro de este manifestó, el método Scrum es considerado dentro del modelo general de desarrollo ágil el mismo que se caracteriza por:

- “Adoptar una estrategia de desarrollo incremental, en lugar de la planificación y ejecución completa del producto.
- Basar la calidad del resultado más en el conocimiento tácito de las personas en equipos auto organizados, que en la calidad de los procesos empleados.
- Solapamiento de las diferentes fases del desarrollo, en lugar de realizar una tras otra en un ciclo secuencial o de cascada” (Scrum, 2014).

El ingeniero Ikujiro Nonaka e HirotakaTakeuchi a principios de los 80, analizaron como las grandes industrias de productos lanzaban al mercado nuevos productos de manufactura tecnológica como: Hewlett-Packard, Canon, Fuji, Xerox, Honda, 3M y Epson.

En estudios preliminares determinaron una nueva gestión de trabajo, su estrategia era de avance en formación de los jugadores de Rugby, a este tipo de habilidad se la denominó “Scrum”.

A pesar que esta manera de orientar el trabajo se debe principalmente a empresas de productos tecnológicos le hace apropiada a proyectos que posean requisitos inestables para lo cual se necesita rapidez de respuesta como flexibilidad en diferentes situaciones en el desarrollo del software.

Ken Schwaber en 1995 presentó “Scrum Development Process” (Schwaber, 1995) , “es un conjunto de reglas de desarrollo de software” (Schwaber, 1995) fundamentado en los principios de Scrum donde él había empleado esta estrategia en empresas como Delphi y otro visionario como Jeff Sutherland en la compañía Easel Corporation.

2.3.2. Operatividad del método Scrum

Se enmarca Scrum, en un conjunto integrado de buenas prácticas de programación y reglas que servirán para dar una solución a los diferentes procesos de desarrollo ágil:

- El proceso es incremental y evolutivo en su avance no predictiva
- Se trabaja orientándose a la calidad del producto, involucrando a las personas como los principales actores para el éxito del proyecto.
- Su táctica es de incremento continuo por iteraciones (Sprint) y revisiones al avance de cada proceso.
- Conceptualizar la visión general de las necesidades y requisitos funcionales que tiene el cliente, la construcción del software de manera incremental a través de iteraciones continuas que comprenden procesos de investigación y revisión. Los Sprint o saltos en estas iteraciones van repitiéndose continuamente hasta que el usuario o cliente de por terminada la evolución del software.

Este método inicia con:

La conceptualización.

Es la instantánea del resultado a obtenerse, se puntualiza en detalle cuáles son las actividades del proyecto donde se dará mayor prioridad y mayor impacto en obtener una cantidad menor tiempo y de costos.

Iteración o Sprint.

Es el ciclo o fase de desarrollo donde terminará con un entregable funcional del software (incremento). Esta duración va desde una hasta varias semanas recomendándose que no vaya a excederse a más de un mes.

Reuniones de equipo y monitoreo

El equipo supervisa y controla el desarrollo de cada salto, a través de reuniones breves diarias, tiene el propósito de revisar las diferentes actividades realizadas por cada integrante del equipo que van desde el día anterior y el posterior. Las reuniones diarias no pueden exceder de 15 minutos, se las denomina “reunión de pie” o “Scrum diario”, debido a que se lleva un seguimiento en una pizarra donde se puntualizan todas las tareas de la iteración como el trabajo pendiente.

Gestión de evolución del proyecto

El método Scrum gestiona de manera incremental e iterativa la evolución del proyecto a través de las tácticas siguientes:

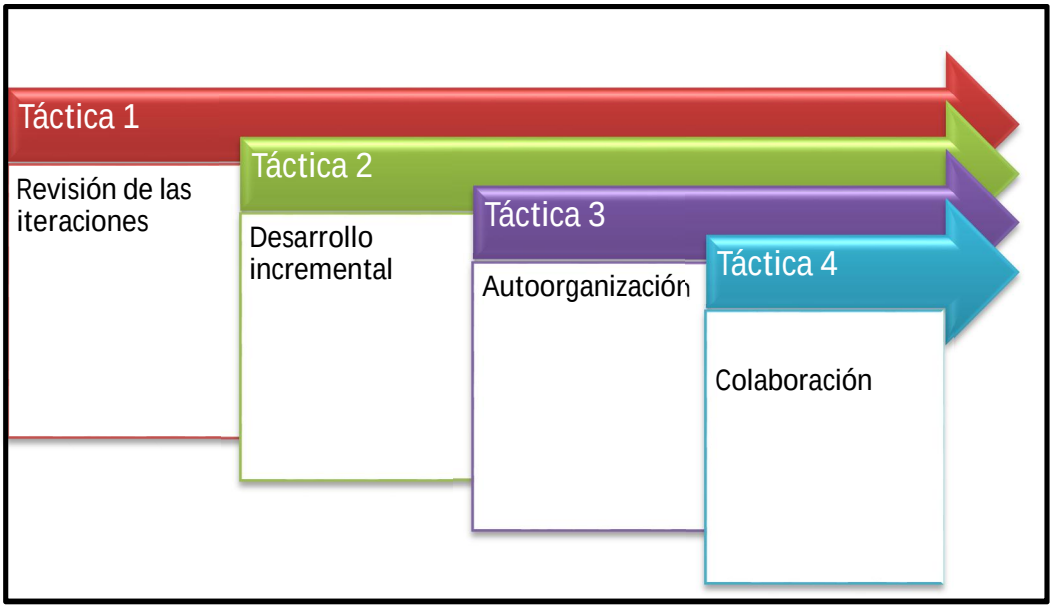


Figura 1.-Tácticas del método Scrum

Elaborado por: Autor

Revisión de las iteraciones

Al término de cada sprint se inspección a la funcionalidad del resultado, donde intervienen todos los miembros del proyecto. Es ahí donde se descubrirán posibles planteamientos erróneos, que pueden ser sujetos de mejora o malas interpretaciones en el desarrollo del sistema.

Desarrollo incremental

“Es la entrega al final de cada salto o iteración que forma parte de un conjunto operativo funcional , que se puede probar, supervisar y valorar” (Scrum, 2014), es ahí donde se nota la aplicabilidad del método Scrum, porque resulta propicio donde la incertidumbre en este tipo de proyectos son naturales , debido a que se considera a la incertidumbre como un indicio, para adoptar metodologías de trabajo que faciliten el desarrollo sin menospreciar y comprometer el diseño, la calidad y que puedan permitir que evolucione.

“Durante la construcción se depura el diseño y la arquitectura, y no se cierran en una primera fase del proyecto. Las distintas fases que el desarrollo en cascada realiza de forma secuencial, en Scrum se solapan y realizan de forma continua y simultánea” (Scrum, 2014).

Auto organización

Es muy común que dentro de proyectos de desarrollo existan gran cantidad de factores impredecibles, es por eso que el asignar un Gestor del Proyecto, fomenta a que los equipos sean auto organizados, con amplio margen de maniobra para que puedan tomar decisiones oportunas.

Colaboración

Compromiso, responsabilidad y colaboración son los principios éticos de todos los miembros del equipo, sin importar sus capacidades, su rol o puesto dentro del proyecto es un componente importante y necesario dentro del método Scrum, con el propósito de ofrecer el soporte necesario y lograr los mejores resultados.

2.3.3. Marco técnico del método Scrum

El marco técnico de Scrum está formado por los siguientes elementos:

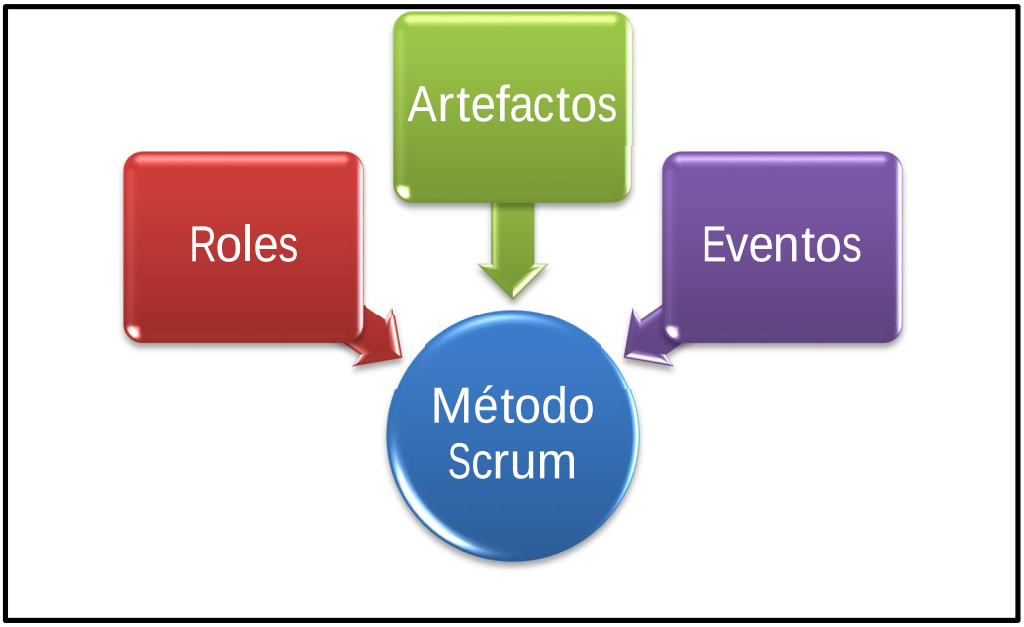


Figura 2.- Marco Técnico del Método Scrum

Elaborado por: Autor

Realizando un estudio rápido a cada elemento junto con sus componentes, se tiene:

2.3.3.1. Roles

2.3.3.1.1 El equipo Scrum

Está conformado por los técnicos que ejecutan el trabajo de entregar la parte funcional del software, también conocido como “Hecho”, al terminar cada Sprint.

2.3.3.1.2. El Scrum Master

Es el líder del equipo Scrum, y entre sus funciones es la de ser el gestor de todo el proceso iterativo del sistema y de que los integrantes del equipo conozcan y operen las diferentes fortalezas de esta técnica.

2.3.3.2. Artefactos

Dentro de “Artefactos” (Scrum, 2014), los desarrolladores de Scrum Management, establecen los siguientes componentes:

2.3.3.3. Pila del producto

Se conoce a este componente como los requerimientos que solicita el usuario a través de una lista de requerimientos funcionales, la misma que se desarrolla en la visión o etapa inicial del proyecto.

2.3.3.4. Pila del sprint

Es la pila de sprint es una lista de los trabajos que es elaborada por el equipo de trabajo la cual generará el incremento previsto que servirá para cada sprint desarrollado.

2.3.3.5. Sprint

Es este elemento se refiere a cada iteración de desarrollo, es decir, es el motor que genera avance del proyecto de acuerdo a los tiempos prefijados. “Se denomina sprint a cada ciclo o iteración de trabajo que produce una parte del producto terminada y funcionalmente operativa” (Scrum, 2014).

2.3.3.6. Incremento

Es el producto generado por cada sprint.

2.3.3.7. Reunión de planificación del sprint

Estas no son más que reuniones de trabajo, las mismas que se llevan a cabo antes de empezar un sprint; cada reunión tiene la meta de establecer los objetivos y las tareas que se deben realizar para conseguirlos.

2.3.3.8. Scrum diario

Estas son reuniones de duración corta que se llevan diariamente con los integrantes del equipo, éstas reuniones tienen que tratar principalmente de: cuál

fue el trabajo realizado el día anterior, el plazo para realizarlo y lo que se necesita para realizar dicho trabajo.

2.3.3.9. Control del Sprint

En este elemento se pretende cubrir el análisis e inspección del incremento formado en el sprint, se prevé también el análisis de si amerita un ajuste a la pila del producto.

2.3.3.10. Medición y Seguimiento del Sprint

Dentro del modelo Scrum, es imprescindible realizar un seguimiento a la gestión por los miembros del equipo y esto va desde los niveles operativos de programadores hasta los niveles superiores de la organización.

Es así que el modelo, plantea una escala de tres aspectos que deben ser monitoreados, medidos y evaluados, estos son:

- **Solución, desarrollo y gestión del proyecto** - Este hace referencia al seguimiento que se le dará al rendimiento, satisfacción del cliente, eficiencia y desempeño del equipo y del producto entregado.
- **Gestión de proyecto**. - A medida que el proyecto va desarrollándose los avances se miden a través de porcentajes, donde se podrá incluir indicadores de: esfuerzo, coste, tamaño, entre otros.
- **Autoorganización. -** Este último es uno de los fundamentales, porque tiene que ver netamente con el nivel de satisfacción laboral, es aquí donde se podrá abarcar aspectos como complejidad del diseño, disponibilidades, densidad de fallos, entre otros.

Figura 3.- Monitoreo y Seguimiento dentro del método Scrum

Elaborado por: Autor

Es necesario se contemple un seguimiento y una medición flexible, basado siempre en el objetivo del método:

El objetivo de un equipo scrum es ofrecer la mejor relación valor / simplicidad

2.4. Metodología de desarrollo PMP (Project Management Professional)

PMP (Project Management Professional) aparece a finales de 1969 como una certificación sobre gestión de proyectos, es patrocinada por el Project Management Institute de los EE.UU, esta credencial se alcanza desde los 3 a 5 años de experiencia y al rendir un examen en su “sitio web” (PMBOK, 2004) para obtener una acreditación.

PMP basa su metodología “en un conjunto de fases del ciclo de vida del proyecto” (PMBOK, 2004), cada fin e inicio de cada ciclo está definida por una entrega técnica de productos entregables que permiten verificar si los objetivos se cumplieron a base de los resultados conseguidos. Para ello se debe identificar:

- Director es el responsable del proyecto
- Cliente/usuario el ente u organización que utilizará el producto
- Equipo de proyecto el que realiza el desarrollo del sistema
- Equipo de Dirección del proyecto son las personas que participan directamente en el proyecto
- Patrocinador es la persona que proporciona los recursos financieros
- Influyentes son las personas que no forman parte del proyecto, pero influyen positiva o negativamente en cada uno de los ciclos del proyecto,

2.4.1. Procesos en el PMP

PMP se ancla principalmente en la gestión de calidad PHVA de Deming en las siguientes etapas: planear, hacer, verificar y actuar.

Todo el proceso está divido en 5 fases o ciclos, estos son:

2.4.2. Procesos de Iniciación

Este proceso es por lo general es establecer las necesidades y requisitos y se establecen las descripciones claras de los objetivos del proyecto. Contiene los alcances y limitaciones como responsabilidades en la dirección dentro del equipo de trabajo, para ello se realiza un acta de constitución y un enunciado preliminar de alcance del mismo.

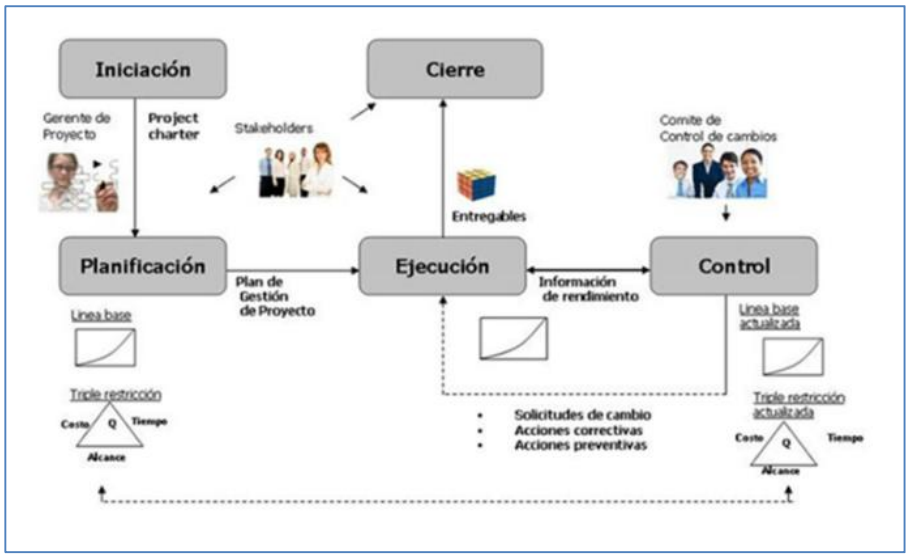


Figura 4.-Gestión de proyecto PMP (PMBOK, 2004)

Fuente: Guía de los Fundamentos de la Dirección de Proyectos

2.4.3. Proceso de planificación

En esta etapa definen los procesos para desarrollar, planificar y gestionar todo el proyecto definiendo el coste del proyecto y detallando en un plan de actividades todas las dependencias, como un análisis FODA dentro del proyecto instrumentándose bucles de retroalimentación que permitan avanzar su desarrollo.

Para ello también se detalla el alcance, las entradas y salidas, las actividades, la secuencia de actividades, la estimación de recursos, la duración de actividades, el cronograma, los costes, el presupuesto, la calidad, el talento humano, las comunicaciones y los riesgos.

2.4.4. Procesos de Ejecución

En este ciclo se involucra el talento humano técnico necesario para realizar el proyecto el cual debe establecer un alcance definido e implementar cambios aprobados. La mayor parte del presupuesto del proyecto se invertirá en esta fase de desarrollo por lo cual es indispensable tener presente los siguientes aspectos en esta fase:

- Dirigir y gestionar la ejecución del proyecto es con el fin de saber exactamente qué productos son entregables por sobre las actividades que han realizado.
- Analizar y asegurar la calidad del software es con el fin de satisfacer los requerimientos del cliente
- Establecer un equipo técnico disciplinado que inicie y termine el proyecto
- Distribución de la información con el fin de alimentar y retroalimentar los avances o cambios en su desarrollo.
- Vendedores son las personas que apoyan indirectamente en el proyecto y son aquellas encargadas de entregar información, presupuestos, cotizaciones, etc.

2.4.5. Procesos de Seguimiento y Control

Es aquella que puede observar, medir e intervenir en los posibles problemas que existen en la ejecución del proyecto, la ventaja de esto es que permite medir las variaciones con respecto al avance del proyecto, de esta manera

se tiene un diagnóstico sobre la situación particular y general a más de las desviaciones que pondrían en peligro la factibilidad de su ejecución.

En esta fase es necesario considerar los siguientes aspectos:

- Supervisar el proyecto se mide el rendimiento y se realiza la evaluación de mediciones para mejorar el proceso
- Control de cambios sirve para asegurarse que los cambios no afecten los resultados previstos y sean beneficiosos para el proyecto
- Verificar el alcance determina exactamente los productos entregables
- Control del cronograma sirve para asegurarse que los tiempos estén adecuados a las tareas y resultados esperados.
- Control de costes se limita a no inflar el presupuesto acordado y controlar los cambios
- Control de calidad determina si los resultados específicos cumplen con los estándares de calidad.
- Gestión del talento humano es un seguimiento de cada integrante del equipo enfocado a encontrar soluciones, coordinar y comunicar los diferentes cambios en cada inicio y final del proyecto.
- Control de riesgos identifica los riesgos y ejecuta planes adecuados para reducirlos.
- Administración del contrato es la gestión entre comprador y vendedor en donde la relación contractual está supeditada a las especificaciones técnicas y funcionalidad del sistema para el cliente.

2.4.6. Procesos de Cierre

Son aquellos utilizados para finalizar las fases de un proyecto, entregar los productos y cerrar definitivamente el proyecto global.

En esta etapa se debe considerar dos aspectos:

- Cerrar el proyecto que indica que para finalizar se debe cumplir con todas las especificaciones, no se podrá cerrar si alguna fase del proyecto quedo como no concluida.
- Cerrar el contrato que indica completar y aprobar cada contrato con su cierre respectivo en cada inicio y fin de cada proceso o fase como del proyecto total.

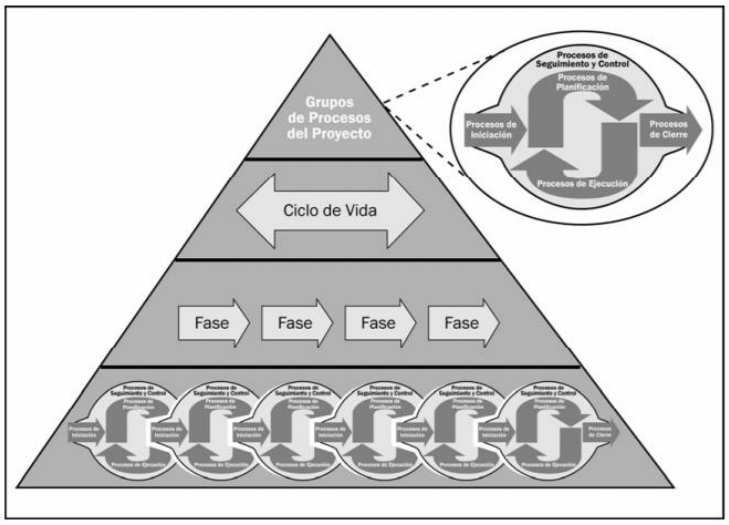


Figura 5.-. Triangulo de Procesos de Proyecto PMP (PMBOK, 2004)

Fuente: Guía de los Fundamentos de la Dirección de Proyectos

2.5. Metodología de desarrollo en Cascada o Lineal

Es conocido como modelo tradicional o secuencial y es el origen ideado por Winston Royce en 1970 el primer pionero en la ingeniería de software y de denominarlo con ese nombre.

Es un conjunto de etapas secuenciales donde el software es desarrollado para su entrega final al cliente.

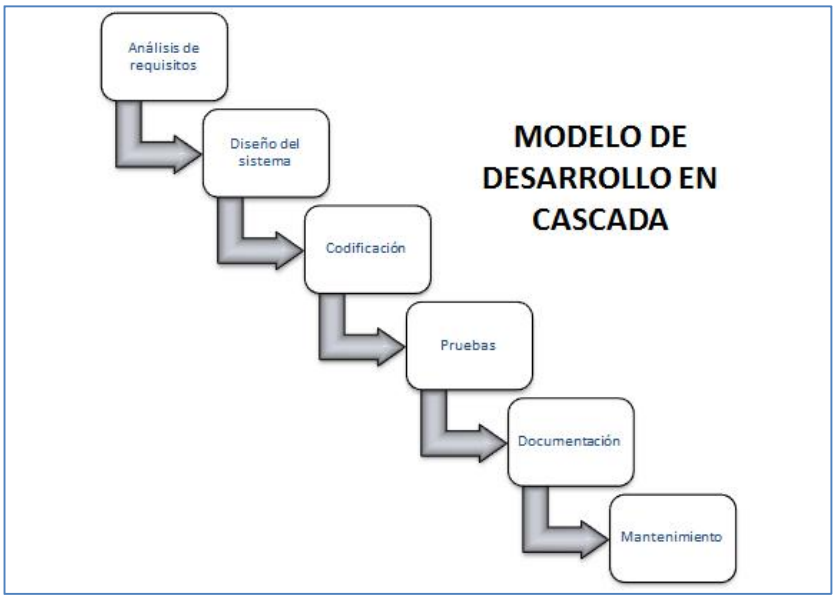


Figura 6.-Modelo en Cascada

Elaborador por: Autor

2.1.5.1 Requerimientos y análisis

Se debe analizar y consensuar todos los requerimientos del sistema juntamente con las necesidades del cliente. Esta etapa es importante porque no deben surgir cambios profundos a la mitad del desarrollo de software.

2.1.5.2 Diseño del sistema

El desarrollo se realiza por equipos de trabajo alimentando mediante un documento de diseño del software. Se irán desarrollando por módulos sobre la estructura global del sistema y la especificación de cada una de sus partes.

2.1.5.3 Codificación

Es donde se realiza la escritura del código fuente haciendo uso de plantillas o prototipos de tal manera se pueda ir corrigiendo errores en el transcurso del desarrollo
Aquí se realiza la reutilización de código fuente como el uso de bibliotecas.

2.1.5.4 Pruebas

El software es probado en su correcto funcionamiento y si cumple con los requerimientos funcionales antes de ser entregado al beneficiario final. Es esta etapa realizan pruebas integrales para verificar que el sistema no falle.

2.1.5.5 Documentación

Esta etapa incluye el manual de usuario del sistema, el funcionamiento, las recomendaciones, así como el mantenimiento en general del sistema.

2.1.5.6 Mantenimiento

Se realiza los respectivos cambios al sistema principalmente de parte del usuario final, esta etapa es de alto riesgo porque se invierte una gran cantidad de tiempo en satisfacer las expectativas del cliente

2.6. Diferencias entre proyecto convencional y ágil

- La metodología de desarrollo ágil es una gestión evolutiva mientras que el convencional es predictiva.
- El desarrollo ágil utiliza una estrategia incremental mientras que el convencional es completa.

- El desarrollo ágil utiliza una táctica incremental mientras que el convencional es por plan.
- El desarrollo ágil se centra en fases solapadas en el trabajo en cambio el convencional es por cascada.
- El conocimiento en el desarrollo ágil se centra en la personas y procesos el convencional en procesos.

Figura 7.-. Diferencias entre la gestión de proyectos convencional y desarrollo ágil

Fuente: Gestión de proyectos Scrum Manager

CAPÍTULO 3

3. DIAGNÓSTICO

Según la consultora internacional The Standish Group (Group, 2012) únicamente el 34% de los proyectos terminan con éxito, los presupuestos en proyectos de IT poseen una desviación del 189% y un 222% en sus retrasos, donde el proyecto ha rebasado los costos y no han cumplido con las expectativas del cliente. Es así que una gran mayoría de empresas de desarrollo desconocen cómo establecer la eficiencia productividad de software.

De cuatro empresas Pymes consultadas en desarrollo de software en Quito (Pymes, 2013), todas ellas no poseen una metodología de desarrollo definida y desarrollan según las necesidades del cliente, del proyecto y asumen los riesgos asociados que con lleva a sobredimensionar los costos del proyecto. Sin embargo, saben de la importancia de tener una metodología adecuada a sus necesidades.

Para continuar con nuestra propuesta de investigación sobre estas metodologías de desarrollo se procedió a consultar a los estudiantes de Ingeniería en Sistemas Informáticos para ello se utilizó como técnica de recolección de datos una encuesta aplicada con la cual se pretende analizar si los estudiantes adoptan estas metodologías en el desarrollo de sus proyectos estudiantiles o en sus lugares de trabajo.

3.1. Instrumentos y técnicas para la recolección de datos

Nuestro universo de estudio está constituido por un solo tipo de población, la cual estuvo formada por los estudiantes de ingeniería en sistemas informáticos de la Universidad Israel de octavo nivel a noveno nivel con un total de 22 personas encuestadas, la muestra de esta población es pequeña en relación al universo de estudiantes de toda la facultad.

El método de muestreo a emplearse es el no probabilístico, todas las personas fueron elegidas cumpliendo con las características específicas necesarias para este tipo de investigación.

3.1.1 Tamaño de la muestra

$$n = \frac{N \cdot Z^2 \cdot \sigma^2}{(N-1) \cdot e^2 + Z^2 \cdot \sigma^2}$$

N =22 (número total de encuestados)

Z = 95% = 1.96 (nivel de confianza)

e = 5% (error de muestreo)

σ = 0.5

n = tamaño de la muestra

n = (1.96^2*22*0.5^2) / (0.05^2*(22 - 1) + 1.96^2*0.5^2)

n = 20.86 = 21

3.1.2. Encuesta

La encuesta tendiente a descubrir sus hábitos como programador y conocer el ambiente en desarrollo de software donde se desenvuelve para tener una idea del nivel de conocimiento de desarrollo de software.

El tipo de encuesta es mixto entre el dicotómico en donde el consultado tiene opciones de verdadero o falso y nominal polifónica donde tienen más opciones desordenadas y por llenar.

3.1.3. Procesamiento de la Información

En el presente trabajo se aplicó la encuesta a los estudiantes con el propósito de descubrir el tipo de conocimiento en desarrollo de software y como lo gestionan en su ambiente estudiantil y laboral.

Pregunta Nº1

¿Tiene algún conocimiento teórico sobre metodologías de desarrollo de software ágil?

Tabla 1.- Resultados Pregunta 1

Si	1
No	21
Total	22

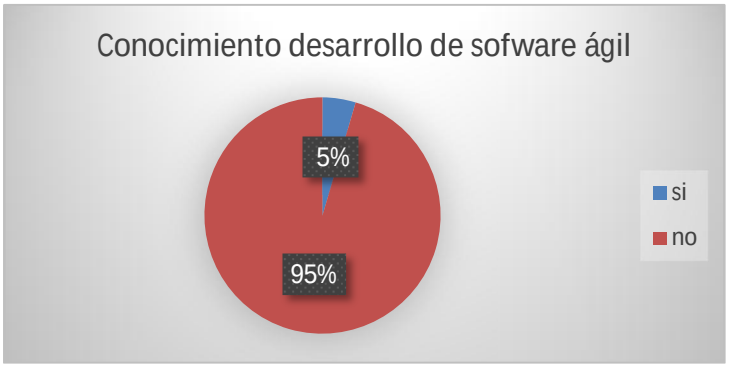


Figura 8.- Conocimiento teórico de desarrollo de software

Elaborado por: Autor

La gran mayoría de estudiantes afirman que no poseen un conocimiento en desarrollo de software ágil.

Pregunta Nº 2

¿Tiene algún conocimiento práctico sobre metodologías de desarrollo de software?

Tabla 2.- Resultados Pregunta 2

Si	2
No	20
Total	22

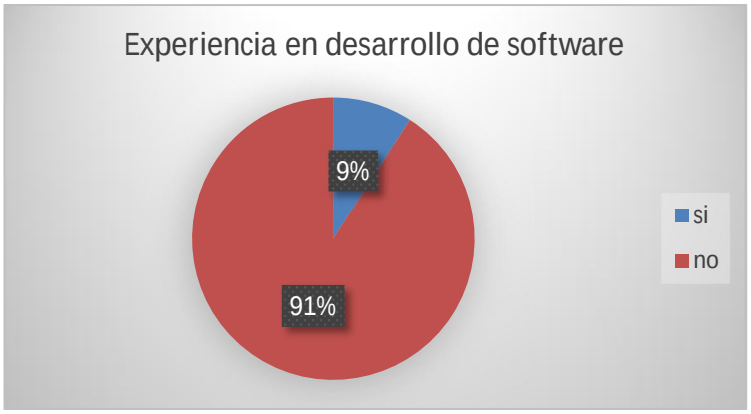


Figura 9.- Experiencia práctica desarrollando software

Elaborado por: Autor

Es evidente que los estudiantes poseen una baja experiencia en desarrollar software.

Pregunta Nº 3

¿Usted ha trabajado antes en una empresa desarrollando software?

Tabla 3.- Resultados Pregunta 3

Si	4
No	18
Total	22

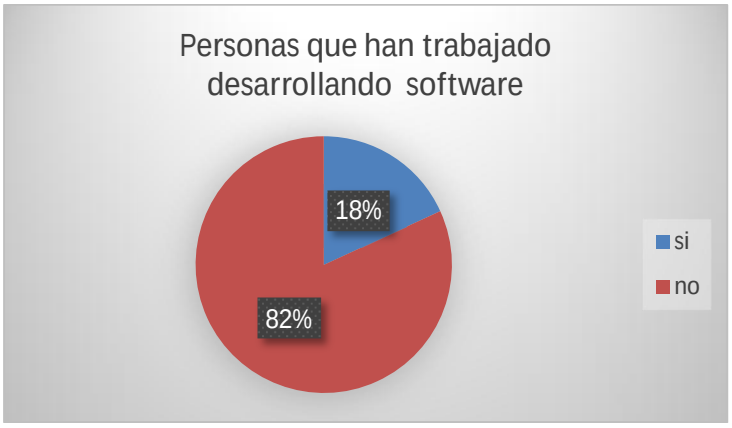


Figura 10.- Experiencia práctica desarrollando software en su trabajo

Elaborado por: Autor

De acuerdo con los resultados existe una baja experiencia en el campo laboral desarrolando software.

Pregunta Nº 4

¿Al programar en una aplicación realiza comentarios de la funcionalidad en una aplicación?

Tabla 4.- Resultados Pregunta 4

Si	6
No	16
Total	22

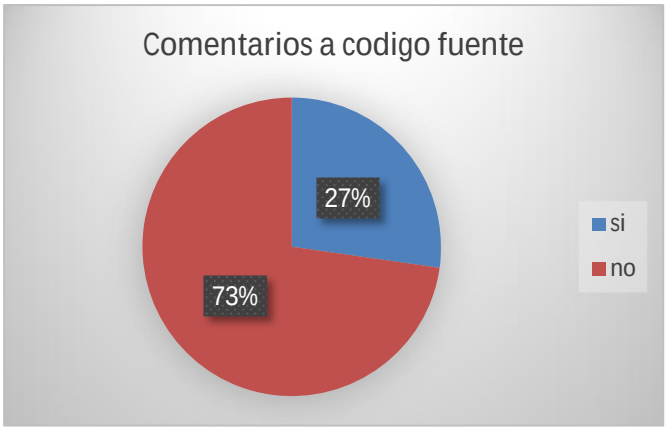


Figura 11.-Porcentaje de personas que realizan comentarios a código fuente modificado

Elaborado por: Autor

De acuerdo con los resultados la gran mayoría no realiza comentarios al código fuente.

Pregunta Nº 5

¿En una aplicación cuando el código donde programa es obsoleto (no sirve)? ¿Usted lo borra?

Tabla 5.- Resultados Pregunta 5

Si	8
No	14
Total	22

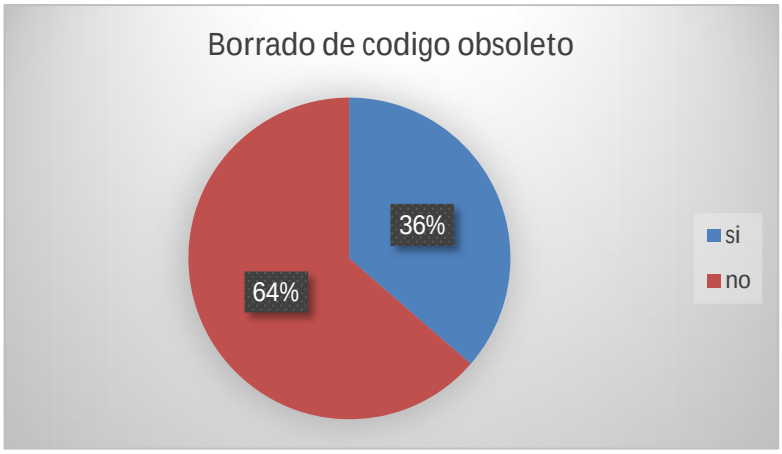


Figura 12.-Porcentaje de personas que borran código obsoleto

Elaborado por: Autor

De acuerdo con los resultados eexiste una gran mayoría de estudiantes que preservan el código en sus aplicaciones.

Pregunta Nº 6

¿Usted ha trabajado desarrollando software en equipo?

Tabla 6.- Resultados Pregunta 6

Si	1
No	21
Total	22

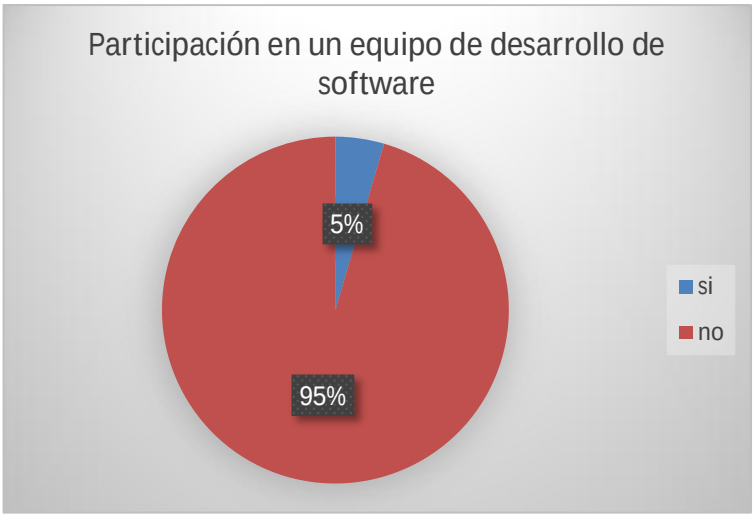


Figura 13.-Participación en algún proyecto de software en equipo

Elaborado por: Autor

De acuerdo con los resultados existe un pequeño número de estudiantes que ha podido participar en equipos de proyectos de software.

Pregunta Nº 7

¿Tiene conocimientos de base de datos?

Tabla 7.- Resultados Pregunta 7

Básico	14
Medio	7
Avanzado	1
Total	22

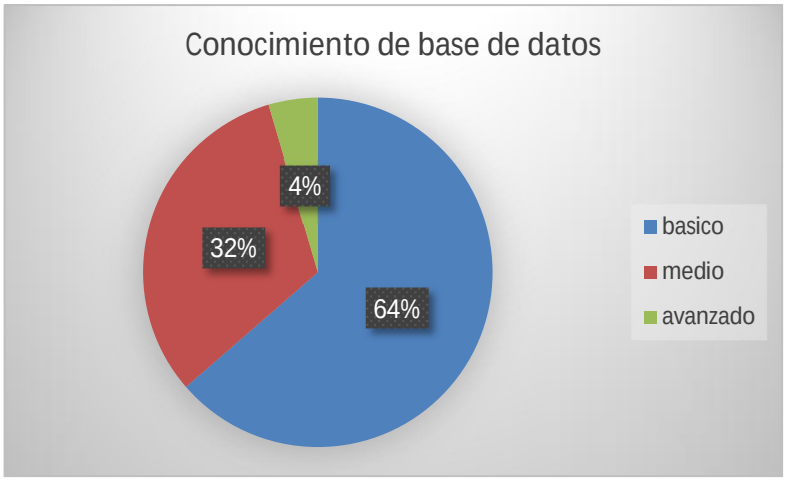


Figura 14.-Porcentaje de conocimiento de base de datos

La gran mayoría posee un conocimiento medio de base de datos.

Pregunta Nº 8

¿Tiene conocimientos de java, básico, medio, avanzado?

Tabla 8.- Resultados Pregunta 8

Básico	12
Medio	9
Avanzado	1
Total	22

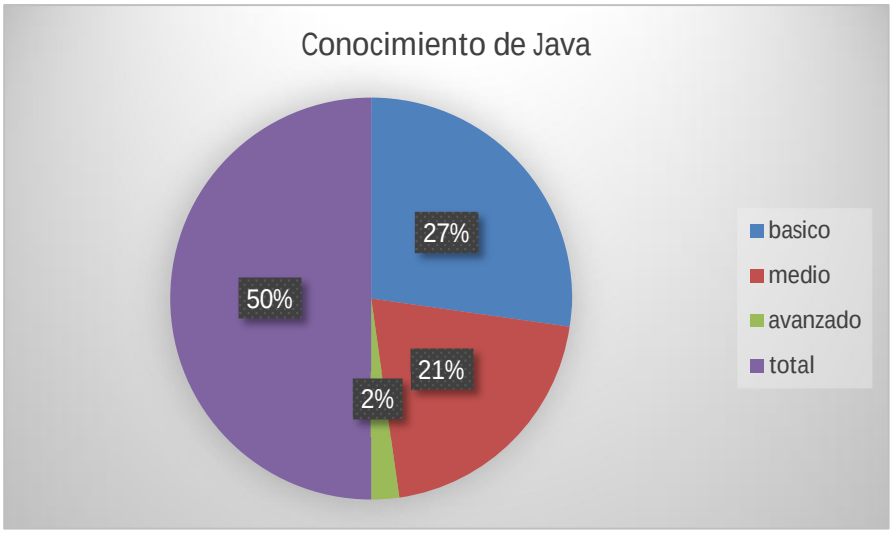


Figura 15.-Porcentaje de conocimiento de lenguaje de programación Java

Elaborado por: Autor

De acuerdo con los resultados la gran mayoría posee un conocimiento básico del lenguaje de programación de java.

Pregunta Nº 9

¿Al programar en una aplicación utiliza buenas prácticas de programación?

Tabla 9.- Resultados Pregunta 9

Si	3
No	19
Total	22

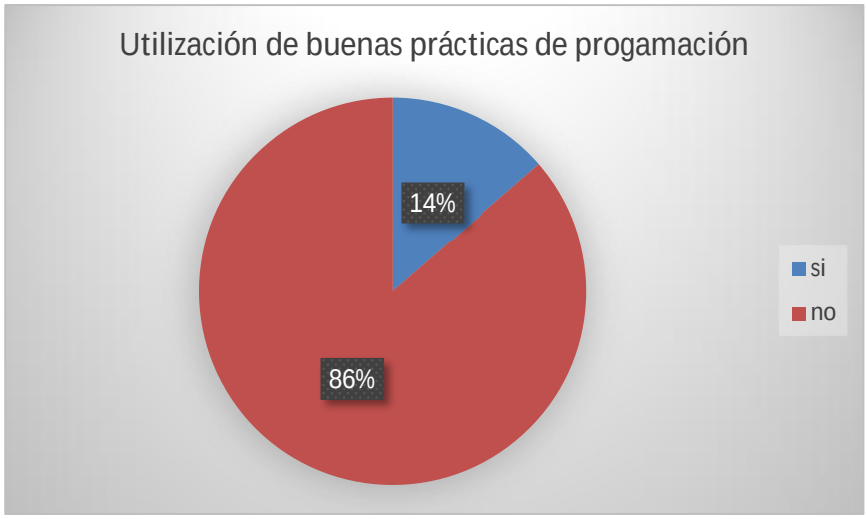


Figura 16.-Porcentaje sobre el uso de buenas prácticas de programación

Elaborado por: Autor

De acuerdo con los resultados la mayoría no utiliza buenas prácticas de programación.

Pregunta Nº 10

¿Cuándo Ud. Programa. la aplicación está orientada a satisfacer las necesidades?

Tabla 10.- Resultados Pregunta 10

Si	20
No	2
Total	22

Figura 17.-Orientación del aplicativo al cliente

Elaborado por: Autor

De acuerdo con los resultados la mayoría de los estudiantes sabe que la satisfacción del usuario/cliente es muy importante.

Pregunta Nº 11

¿Cuándo Ud. programa cuales son los problemas al desarrollar software?

Tabla 11.- Resultados Pregunta 11

Falta de conocimientos basicos y medios del lenguaje	9
Mala documentación en los requerimientos funcionales	3
Pésima gestión en el alcance de proyecto	8
Falta de estándares de programación	1
Coordinación de trabajo en equipo	1
Total	22

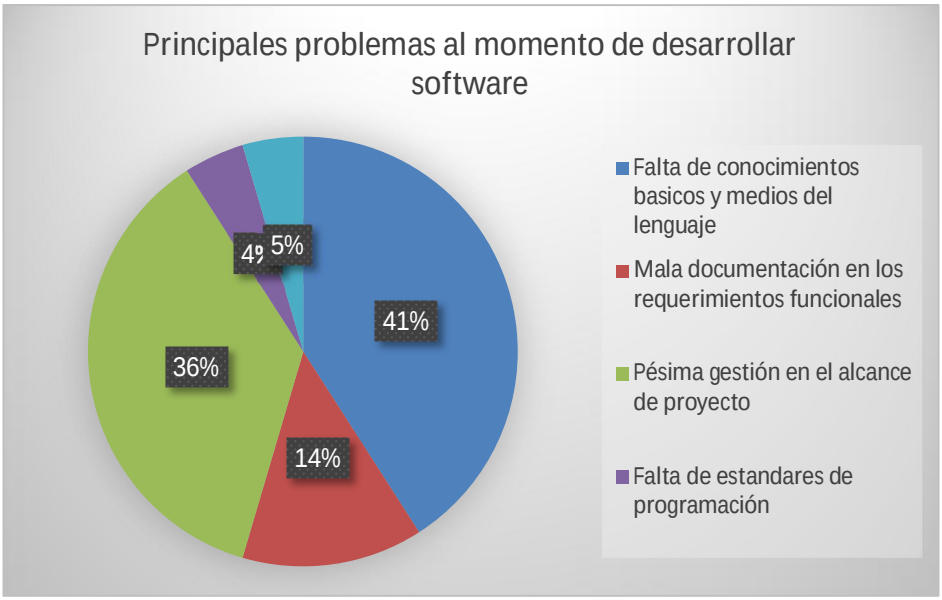


Figura 18.-Principales problemas al desarrollar software

Elaborado por: Autor

El resultado indica que las mayores dificultades están en conocer el lenguaje y en la gestión del proyecto**.**

Pregunta Nº 12

¿Utiliza Ud. Pruebas unitarias para demostrar que su aplicativo sirve?

Tabla 12.- Resultados Pregunta 12

Si	14
No	8
Total	22

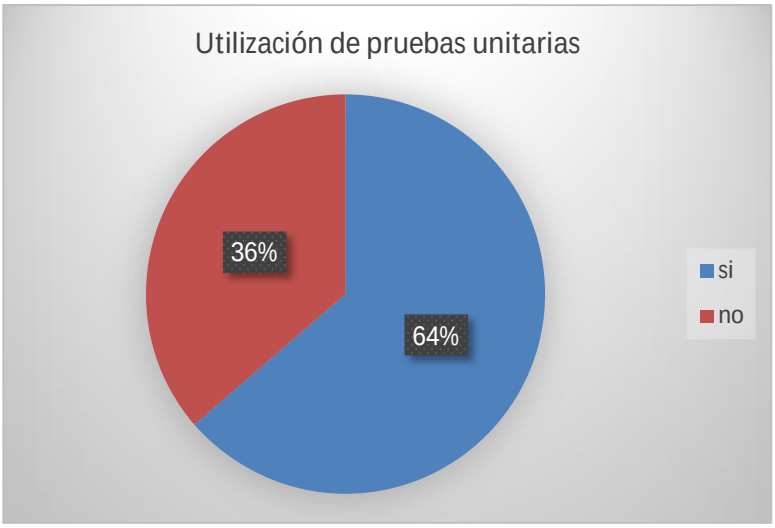


Figura 19.-Utilización de pruebas unitarias

Elaborado por: Autor

Un gran porcentaje de estudiantes realiza sus propias pruebas unitarias para verificar la funcionalidad del aplicativo.

Pregunta Nº 13

¿El desarrollo de software es en sí, es un proceso complejo?

Tabla 13.- Resultados Pregunta 13

Si	20
No	2
Total	22

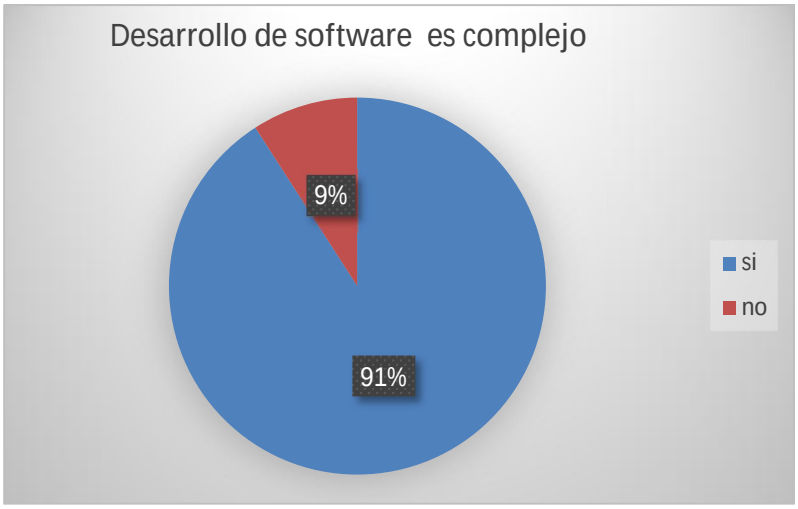


Figura 20.-Desarrollar software es complejo

Elaborado por: Autor

El resultado indica que los estudiantes piensan que es complejo desarrollar software

Pregunta Nº 14

¿El desarrollo de una metodología de software elimina los riesgos?

Tabla 14.- Resultados Pregunta 14

Reduce los riesgos	16
No reduce los riesgos	6
Total	22

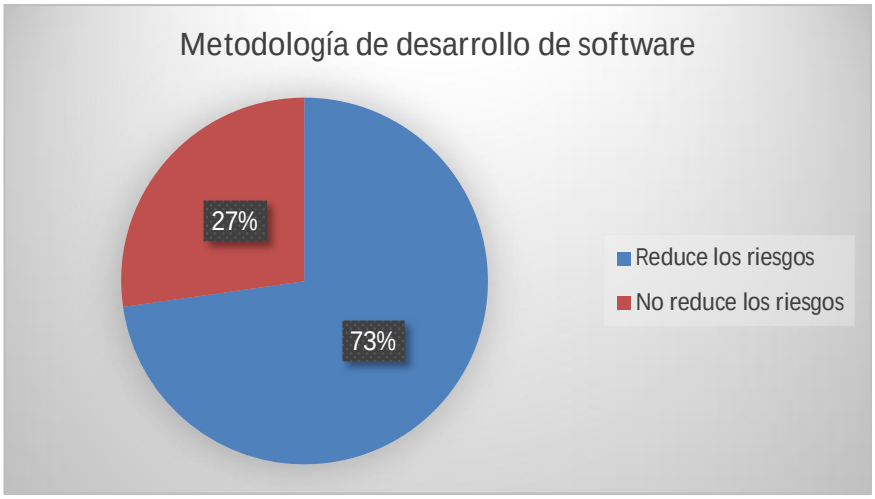


Figura 21.-. Reducción de riesgos al utilizar metodología de desarrollo

Elaborado por: Autor

La mayoría de estudiantes piensa que el utilizar metodologías ayuda a reducir los riesgos durante el proceso de desarrollo.

3.1.4. Interpretación de los resultados

La encuesta demostró una deficiencia en desarrollo de software como buenas prácticas de programación y conocimiento de lenguaje de programación java, por lo cual se procederá a capacitar a un grupo de estudiantes en los aspectos citados anteriormente para poder empezar con las comparaciones de la eficiencia del desarrollo de software ágil en los laboratorios de la Universidad Israel.

CAPÍTULO 4

4. TEORÍA DE LAS HERRAMIENTAS

4.1. Entorno de desarrollo IDE NetBeans

Para el desarrollo del aplicativo se trabajara con el entorno de desarrollo NetBeans hecho principalmente para el lenguaje de programación Java, Php, Android entre otros. Existe además una amplitud de componentes para poder extenderlo. El IDE NetBeans es un producto de software libre.

El IDE NetBeans es un entorno integrado de desarrollo - Es una herramienta de lenguaje para desarrolladores enfocada en escribir, compilar, depurar y ejecutar programas. Su codificación esta en Java, pero pueden ir agregándose componentes de otros lenguajes, su ventaja principal es un software de uso libre y sin restricciones.

El NetBeans IDE soporta el desarrollo de todos los tipos de aplicación Java (J2SE, web, EJB y aplicaciones móviles). Su característica se encuentra en un sistema basado en proyectos en Ant, control de versiones y refactoring.

La versión a aplicarse es la NetBeans IDE 8.0.2

4.2. Sistema de gestión de base MySQL

Tendremos una base de datos en Mysql que permitirá crear todo el esquema de la información requerida Mysql. Se utilizará esta base de datos relacionales MySQL, por su fácil conexión con IDE Netbeans y en aplicaciones web, como Drupal o phpBB, en plataformas (Linux/Windows-Apache-MySQL-PHP/Perl/Python), y por herramientas de seguimiento de errores como Bugzilla. MySql está estrechamente ligada a Php donde su conexión es relativamente fácil.

MySQL es una base de datos muy eficiente en la lectura cuando utiliza un motor no transaccional MyISAM, pero puede provocar problemas de integridad en entornos de alta concurrencia en la modificación. En las aplicaciones web existe una baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL una base de datos ideal para estas aplicaciones. Es recomendable supervisar el rendimiento de la base para poder detectar y corregir errores de SQL en programación.

Es un sistema de gestión de bases datos relacional multiusuario y multihilo y con más de ocho millones de instalaciones.

Su licencia es GNU GPL para cualquier uso, sin embargo para aquellas empresas que necesiten sus productos privativos deberán comprar su licencia para su uso.

La versión a utilizarse para esta investigación es la Mysql 5.6

4.3. Modelo vista controlador MVC

Es un patrón de arquitectura de software que permite separar la lógica de negocio de las vistas con los datos de una base relacional. Este patrón se utiliza principalmente por la lógica de reutilización de código y la separación de conceptos a lo que posterior se define un adecuado mantenimiento y un desarrollo eficiente de código en las aplicaciones.

Modelo: es la que gestiona todos los accesos a la información del sistema como actualizaciones, privilegios de acceso según la lógica de negocio y envía las vistas de usuario en cada momento que le sea solicitada para ser mostrada al usuario todas estas peticiones de acceso llegan a través del controlador.

Controlador: es aquel que responde a todos los eventos y peticiones del modelo sobre peticiones. El controlador hace de intermediario entre la vista y el modelo

Vista: es aquel que muestra la información en un formato concreto para interactuar con el usuario e interactúa con el modelo y presenta las vistas o interfaz del usuario como mecanismo de salida.

4.4. Lenguaje de programación JAVA

Como lenguaje de programación utilizaremos java para plataforma de escritorio para todo el desarrollo de la aplicación sus puntos fuertes: “es un lenguaje de programación orientado a objetos fundamentado en clases que fue diseñado específicamente para tener pocas dependencias de implementación como fuera posible. Su objetivo es permitir que los desarrolladores escriban el programa una vez y lo ejecuten en cualquier dispositivo (conocido en inglés como *WORA*, o "*write once, run and where*"), lo que quiere decir que el código ejecutado en una plataforma no tiene que ser recompilado para correr en otra” (Augusto, 2008). Java es, uno de los lenguajes más predilectos ahora de programar aplicaciones de cliente-servidor de web.

JAVA fue creado por James Gosling, Patrick Naughton, Chis Warth, Ed Frank y Mike Sheridan en Sun Microsystem en 1991, este surgió de la necesidad de diseñar un lenguaje de programación de equipos de baja potencia de cálculo y memoria, los mismos que generen un código de baja potencia de cálculo y memoria.

El lenguaje Java se centra según Córdova en la “creación, manipulación y construcción de objetos. El mundo real está lleno de objetos, todo objeto tiene unas propiedades y un comportamiento. Cualquier concepto que se desee implementar en un programa Java debe ser encapsulado en una clase” (Augusto, 2008).

4.5. Modelo de la presentación propuesta

4.5.1. Validación

El presente proyecto de investigación, estará basado en el Método de Investigación Experimental, el mismo que según José Cegarra Sánchez, no es más que: “un método que posee una unidad conceptual y operativa, según el problema planteado se pueden apreciar algunas diferencias entre sus diferentes variedades” (Sánchez, 2011).

Para el desarrollo del proyecto se contrastará dos ensayos utilizando los mismos requerimientos funcionales de desarrollo. La primera se realizará sin metodología de desarrollo (desarrollo heroico) donde el equipo construirá su proyecto a su consideración.

La segunda se utilizará la metodología SCRUM como marco de trabajo y desarrollo de proyectos ágiles basada en un proceso iterativo e incremental utilizado comúnmente en entornos basados en el desarrollo ágil de software.

4.5.2. Análisis

En el laboratorio de la Universidad Israel en la Facultad de Sistemas Informáticos se realizará la investigación respectiva con la participación de un equipo de estudiantes para desarrollar la metodología de desarrollo ágil Scrum; para demostrar el uso de esta metodología en el ambiente de pruebas utilizaremos un ejemplo de software enfocado a la actividad veterinaria. El software se encuentra diseñado en lenguaje de programación en java, con base de datos Mysql.

Es indispensable realizar un mini taller sobre esta metodología de desarrollo Scrum para que las personas involucradas en el proyecto a construir puedan desarrollar el aplicativo sin inconvenientes.

Se medirá el nivel de organización, dirección, ejecución del proyecto del aplicativo. El control se lo realizará en la eficiencia y efectividad de entregar un producto entregable en máximo 8 horas de programación.

Se realizará únicamente un salto o sprint para demostrar la eficiencia del desarrollo, la ventaja en este caso es el tiempo y aprovechamiento de todos los recursos disponibles para alcanzar el salto respectivo.

La etapa de pruebas tiene una duración de una semana.

4.5.3. Encuesta

Se realizará y se tomará un cuestionario a 22 estudiantes de la Facultad de Ingeniería en Sistemas Informáticos de la Universidad Israel, encuesta tendiente a descubrir sobre su entorno de trabajo como desarrollador y conocer su ambiente en desarrollo de software para determinar el nivel de conocimiento de desarrollo de software. El tipo de cuestionario es mixto entre el dicotómico en donde el consultado tiene opciones de verdadero o falso y nominal polifónica donde tienen más opciones desordenadas y por llenar.

Desarrollo heroico (sin metodología)

Para demostrar la eficiencia y eficacia de la metodología iniciaremos con una prueba sin metodología de desarrollo donde los participantes aplicarán su propio método para poder integrar el sistema únicamente en el módulo de administración hasta ese nivel se integrará el producto entregable. Así mismo la duración de esta prueba tendrá una duración de dos días. Se determinará la productividad, líneas de código y buenas prácticas de programación.

4.5.4. Desarrollo de la metodología

La segunda prueba es la utilización de la metodología de desarrollo Scrum donde se utilizarán todos los instrumentos y buenas prácticas de desarrollo

como: buenas prácticas de codificación, establecimiento de MVC, organización de carpetas en todo el proyecto, reutilización de código entre otras.

Se realizó un mini taller de capacitación para el manejo apropiado de esta metodología como:

- Programación heroica vs desarrollo ágil
- Buenas prácticas de programación
- Programación java advanced
- Metodología de desarrollo ágil Scrum, TPM y Lineal

4.5.5. Diseño

Definición de requerimientos funcionales

a. El programa tendrá que ofrecer las respectivas consultas, búsquedas, creación, borrado, actualización (CRUD- Create, Read, Update and Delete) BD:

    1. Administración

b. El programa tendrá que ofrecer diferentes usuarios para el ingreso al sistema:

    1. Administrador sistema

c. El lenguaje a utilizarse será JAVA software libre para la aplicación y conectividad con la BD.

d. El hardware utilizado para el software será desde un intel Pentium IV hasta un core i7.

4.5.6. Diagrama de casos de usos

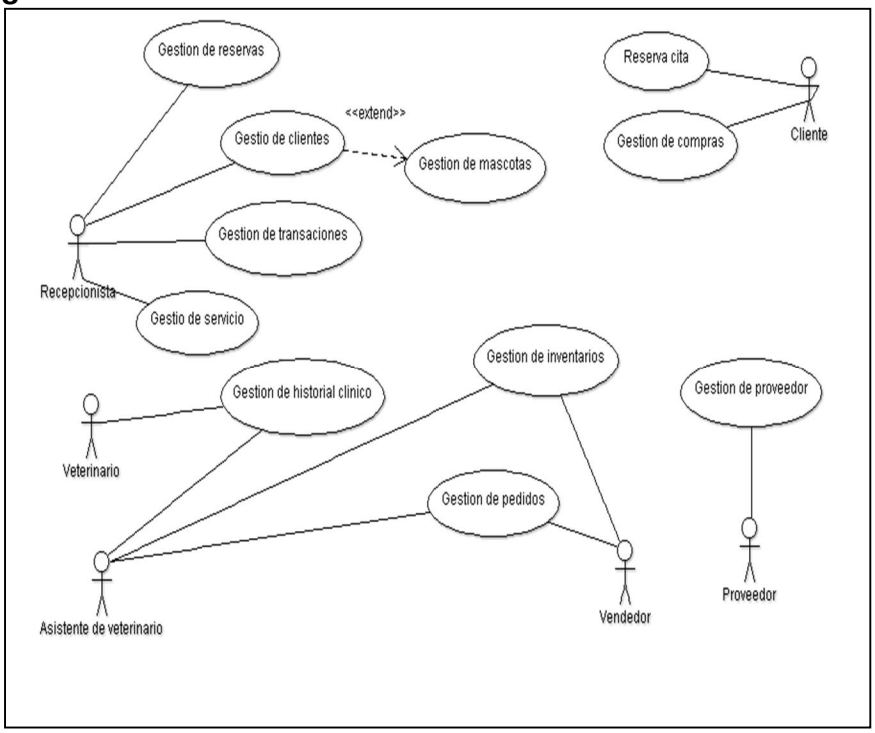


Figura 22.-Casos de uso

Elaborado por: Autor

4.5.7. DIAGRAMA ITERACION

HISTORIAL CLINICO

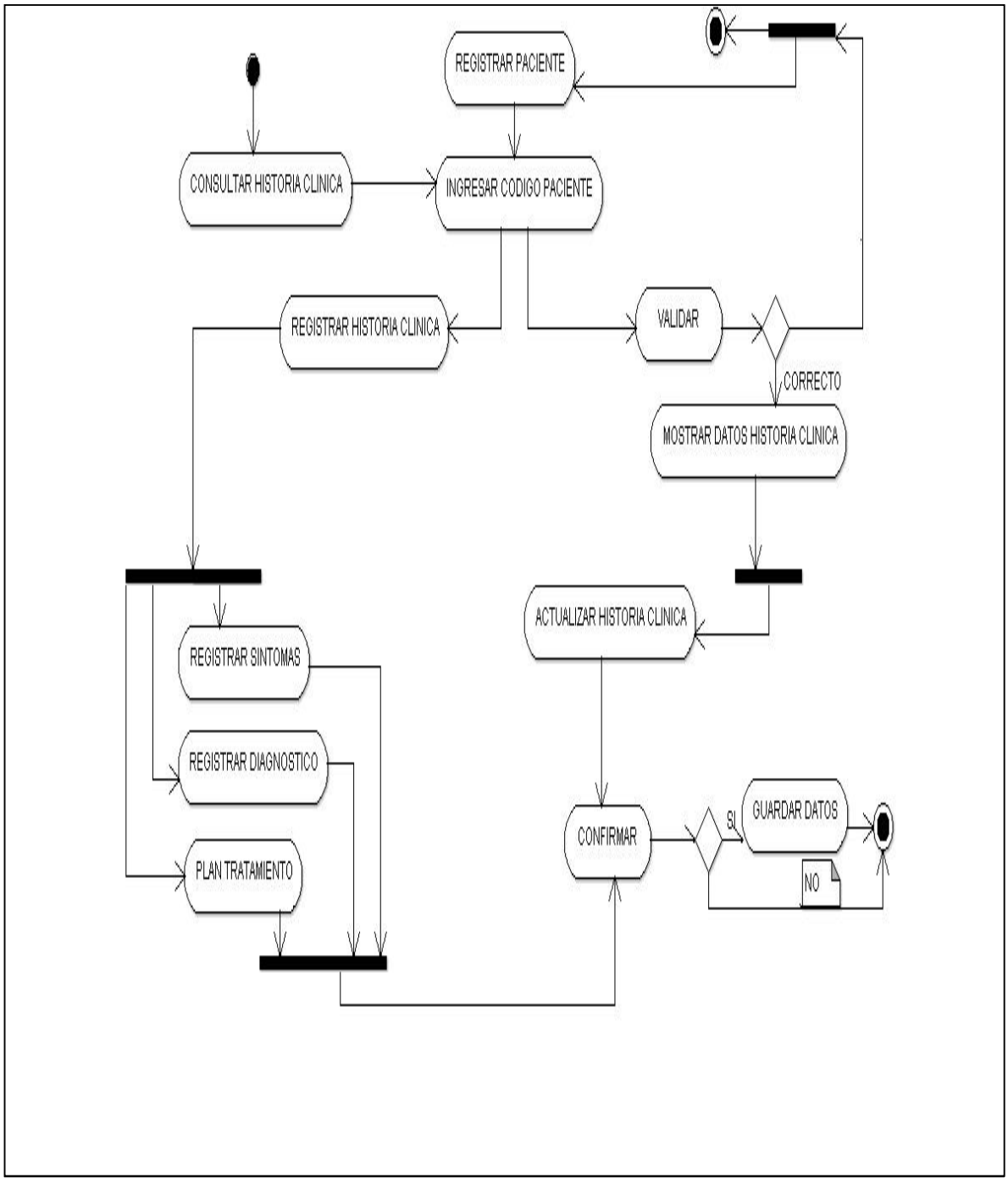


Figura 23.-Diagrama de iteración atención veterinaria

Elaborado por: Autor

4.5.7. Diagrama de secuencia
DEL HISTORIAL CLINICO

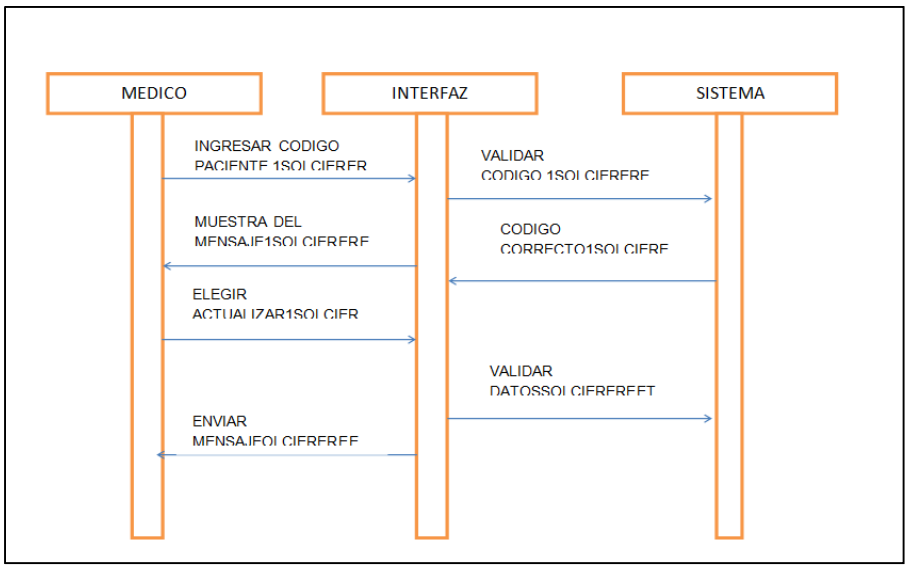


Figura 24.-Diagrama de Secuencia de atención veterinaria

Elaborado por: Autor

4.5.8. Esquema de base de datos

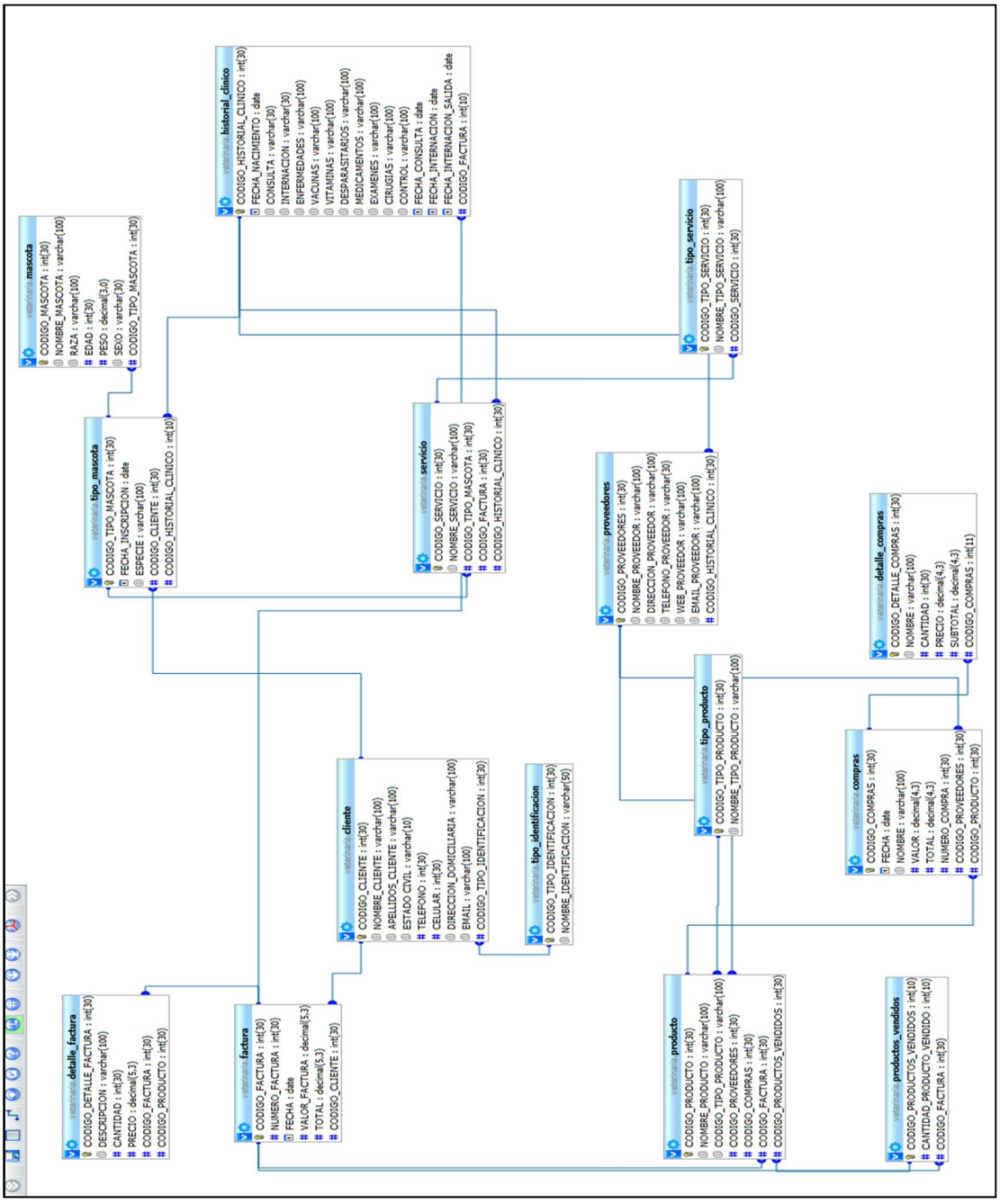


Figura 25.-Diagrama de Base de Datos

Elaborado por: Autor

4.5.9. Vistas

Figura 26.-Diagrama de Vistas

Elaborado por: Autor

Figura 27.-Diagrama de Vistas

Elaborado por: Autor

Figura 28.-Diagrama de Vistas

Elaborado por: Autor

4.5.10. Desarrollo de las metodologías

El desarrollo tomo unas seis horas completando todas las funcionalidades del aplicativo, reorganizando completamente las carpetas en el proyecto de NetBeansIDE, establecimiento de conexiones, manejo lógico CRUD, mejores prácticas de programación y reutilización de código fuente.

El proceso se inició con una planificación estableciendo:

1. Los roles principales:
    a. Los involucrados en el proyecto
        - Scrum Master, director de PMP y Lineal
        - Team (Equipo de trabajo)

    b. Los comprometidos en el proyecto
        - Usuario/cliente

Administración

Se estableció el primer sprint que es el módulo de administración que contendrá los requerimientos de crear, actualizar, borrar usuarios en el sistema

PRIORIDAD	ITEM	ESTIMACION	ESTADO	CRITERIOS DE ACEPTACION
1	ADMINISTRACION	8	EN CURSO	El sistema permitirá registrar a usuarios y clientes en el sistema. El sistema funcionará en equipos con S.O. Windows. El lenguaje de la aplicación es Java Entorno de desarrollo IDE NetBeans 8.0.2 El sistema brindara seguridad a usuarios registrados en el sistema.

Tabla 15.- Sprint

Elaborado por: Autor

Se identificaron las siguientes historias

PRIORIDAD	HISTORIA DE USUARIO	ESTIMACION	IMPORTANCIA	DESCRIPCIÓN
1	Creación de perfiles de usuario	2	EN CURSO	Manejo de usuarios
2	Iniciar sesión en el sistema	2	EN CURSO	Brindar seguridad en el acceso de la aplicación
3	Administración de usuarios	4	EN CURSO	Administrar usuarios en perfiles

Tabla 16.- Historias

Elaborado por: Autor

Se definió el equipo de trabajo Scrum

ROL	EQUIPO	TAREA
PRODUC OWNER	Cliente	ADMINISTRACION NEGOCIO
SCRUM MASTER	Líder 1	ADMINISTRACION SCRUM /PRUEBAS
TEAM	EQUIPO DESARROLLO PRIORIDAD 1 PROGRAMADOR1 PRIORIDAD 2 PROGRAMADOR 2 PRIORIDAD 3 PROGRAMADOR 3	CODIFICACION Y PRUEBAS

Tabla 17.- Roles y equipos

Elaborado por: Autor

Se definió el equipo de trabajo para PMP

ROL	EQUIPO	TAREA
Iniciación	Gerente de proyecto	VISION ADMINISTRACION NEGOCIO PRODUCTOS PRUEBAS CIERRE
Planificación	Líder 1/ Gerente de proyecto	ADMINISTRACION PRODUCTOS PRUEBAS CIERRE
Ejecucion	EQUIPO DESARROLLO PRIORIDAD 1 PROGRAMADOR1 PRIORIDAD 2 PROGRAMADOR 2 PRIORIDAD 3 PROGRAMADOR 3	CODIFICACION Y PRUEBAS PRODUCTOS PRUEBAS CIERRE
Control	PRIORIDAD 1 PROGRAMADOR1 PRIORIDAD 2 PROGRAMADOR 2 PRIORIDAD 3 PROGRAMADOR 3	 PRODUCTOS PRUEBAS CIERRE

Tabla 18.- Equipo de PMP sus funciones y roles

Elaborado por: Autor

Se definió el equipo de trabajo para Lineal o Cascada

ROL	EQUIPO	TAREA
Análisis de requisitos	EQUIPO DESARROLLO PRIORIDAD 1 PROGRAMADOR1 PRIORIDAD 2 PROGRAMADOR 2 PRIORIDAD 3 PROGRAMADOR 3	Análisis de requisitos
Diseño del sistema	EQUIPO DESARROLLO PRIORIDAD 1 PROGRAMADOR1 PRIORIDAD 2 PROGRAMADOR 2 PRIORIDAD 3 PROGRAMADOR 3	Conceptualización del sistema
Codificación	EQUIPO DESARROLLO PRIORIDAD 1 PROGRAMADOR1 PRIORIDAD 2 PROGRAMADOR 2 PRIORIDAD 3 PROGRAMADOR 3	Codificación y pruebas

Pruebas	PRIORIDAD 1 PROGRAMADOR1 PRIORIDAD 2 PROGRAMADOR 2 PRIORIDAD 3 PROGRAMADOR 3	PRUEBAS
Mantenimiento	PRIORIDAD 1 PROGRAMADOR1 PRIORIDAD 2 PROGRAMADOR 2 PRIORIDAD 3 PROGRAMADOR 3	

Tabla 19.- Equipo y gestión para Lineal o Cascada

Elaborado por: Autor

4.5.11. Presentación de los resultados

El desarrollo del aplicativo se lo realizó en los laboratorios de la Universidad Tecnológica Israel comprendiendo un periodo de 12 días laborables comprendiendo cuatro días para la respectiva capacitación en la metodología de desarrollo ágil.

En el desarrollo del aplicativo se realizaron las siguientes pruebas al equipo de desarrollo: la primera sin metodología de desarrollo, la segunda con la metodología de desarrollo en cascada o lineal, la tercera con la metodología Scrum y por último se gestionó el proyecto con la metodología PMP la cual contó con todas las herramientas de gestión y requerimientos señalados para el desarrollo del aplicativo.

De este proceso de pruebas se logró obtener los respectivos resultados:

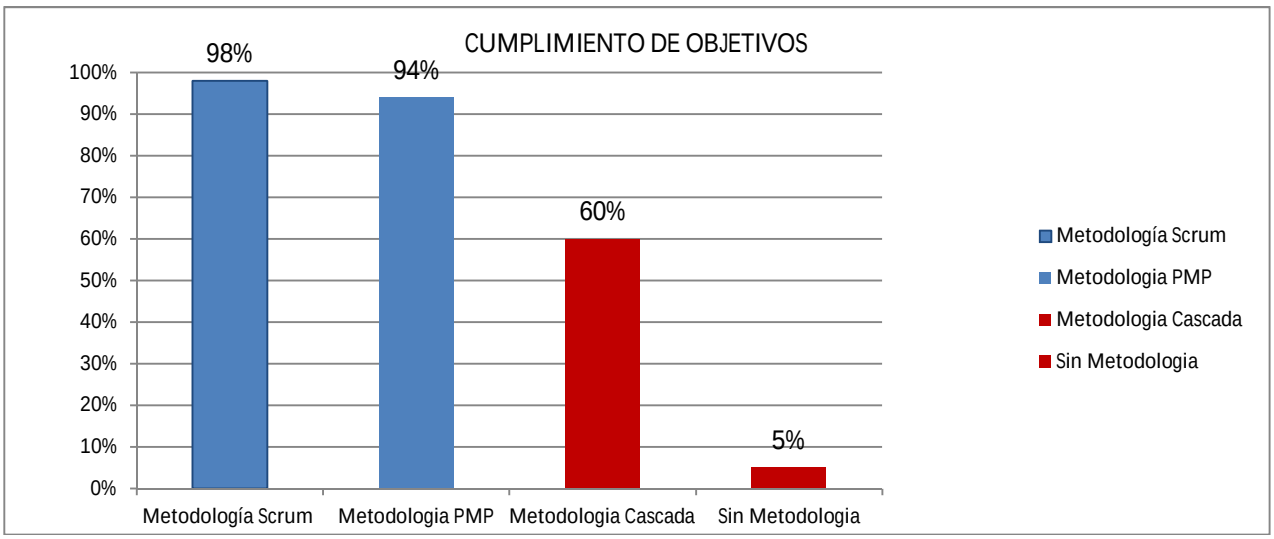


Figura 29.-Cumplimiento de objetivos
Elaborado por: **Autor**

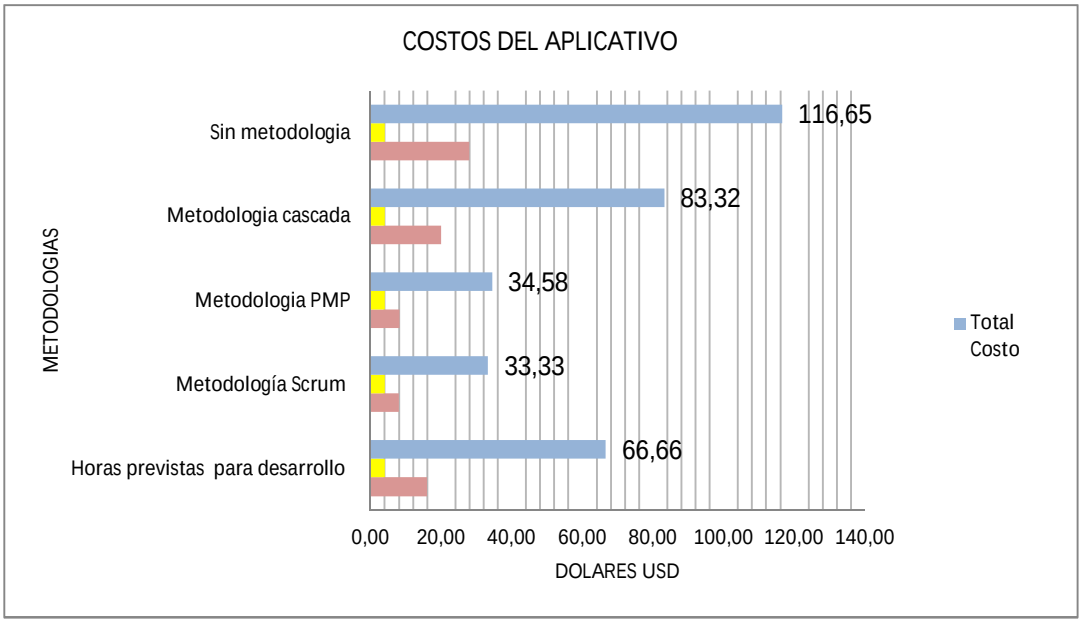


Figura 30.-Costos del aplicativo por hora de programación
Elaborado por: **Autor**

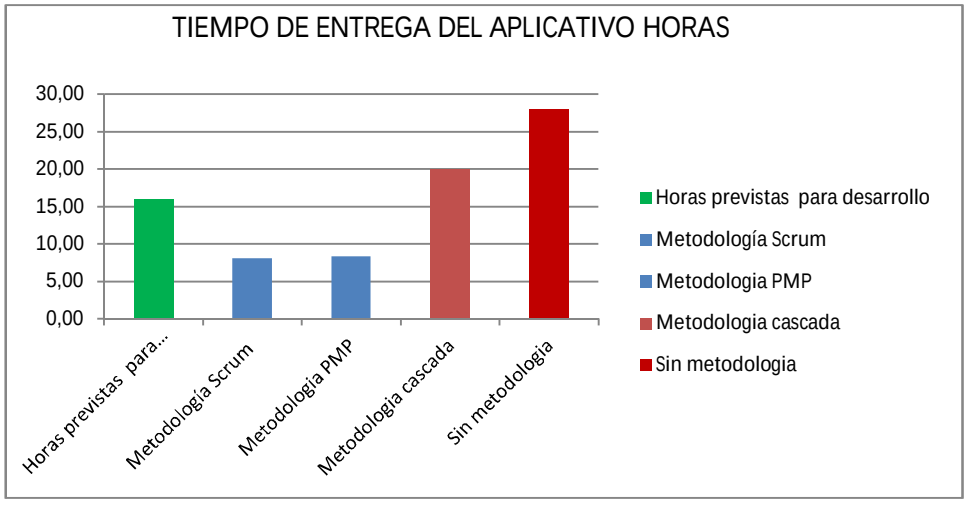


Figura 31.-Tiempo de entrega del aplicativo

Elaborado por: Autor

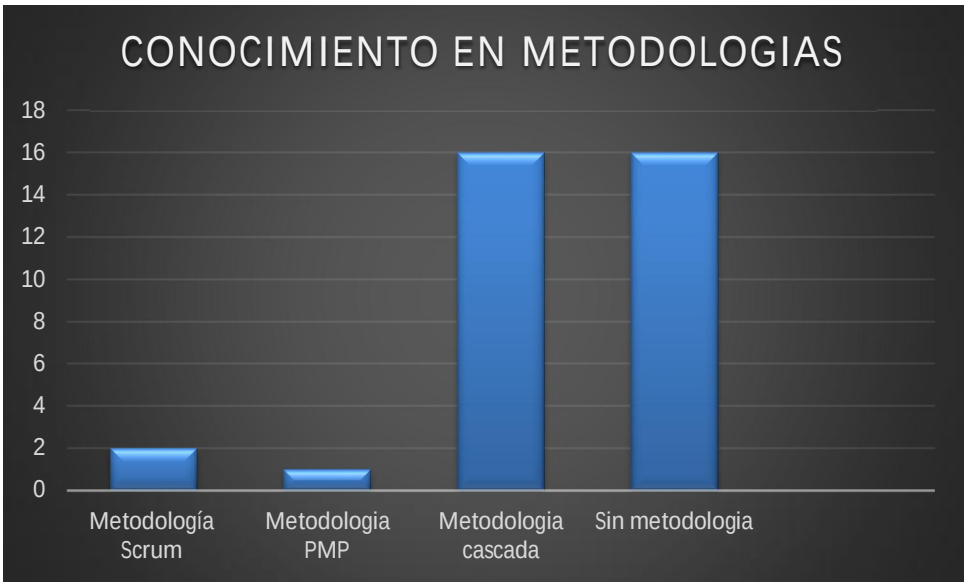


Figura 32.-Conocimiento de metodologías del equipo de desarrollo antes de realizar la prueba

Elaborado por: Autor

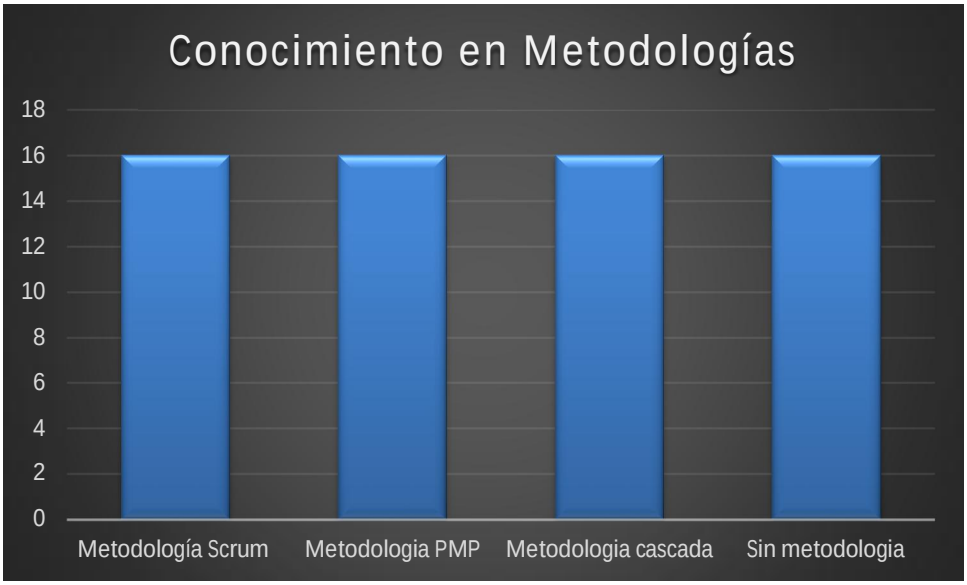


Figura 33.-Conocimiento de metodologías del equipo de desarrollo después de realizada la prueba

Elaborado por: Autor

CONCLUSIONES

- El uso de estas metodologías ágiles como Scrum y la metodología PMP al contar con directivas establecidas tanto en la ejecución del proyecto como en la gestión del mismo, permite que en cada hito o etapa pueda ser susceptible a cambio debido a la retroalimentación continua esta ventaja competitiva la hace valiosa frente al uso de otras metodologías tradicionales como la lineal o cascada donde las fases de requerimientos funcionales y de sistemas, análisis y diseño, pruebas y corrección de errores de un ciclo tradicional de proyecto son muy burocráticas.

- El uso de estas metodologías ágiles que se desarrollaron en los laboratorios de la Universidad Israel con los estudiantes de Ingeniería en Sistemas posibilitó un trabajo más eficiente y mejor organizado en el desarrollo del aplicativo de prueba, además permitió que el equipo de trabajo se encuentre involucrado con los requerimientos en el proceso de desarrollo de software.

- El uso de la metodología Scrum y PMP probo un cumplimiento de objetivos del 98% y 94% lo que significa que cumple con la totalidad de lo que nos hemos propuesto, respectivamente frente a la tradicional metodología Lineal, esto se debe a la gestión y flexibilidad que con lleva administrar un proyecto de software en lo que se refiere a los cambios y a la naturaleza del proyecto como las necesidades a veces imprevistas del cliente.

- Se documentó el proceso de desarrollo ágil mediante el caso de estudio realizado en los laboratorios de la Universidad Israel, el cual viene adjunto al trabajo de titulación pertinente

- La capacitación en los talleres de buenas prácticas de programación, java avanzado, desarrollo ágil Scrum y PMP permitieron fortalecer el talento

humano de los estudiantes de la Facultad de Sistemas, los cuales se encuentran capacitados para integrarse a equipos de trabajo donde se requiera este tipo de habilidades técnicas.

RECOMENDACIONES

- Se sugiere talleres especializados sobre metodologías ágiles de desarrollo de software en la Universidad Israel para que el estudiante pueda ser capacitado a fin de que pueda desenvolverse profesionalmente en su entorno estudiantil como laboral.

- Es aconsejable que la Universidad instrumente una certificación internacional en este tipo de metodologías en su malla curricular, de esta manera el estudiante recién graduado podría contar con una ventaja competitiva en el mercado laboral.

- Los desarrollos de software futuros por los estudiantes con metodología ágil deben ser realizados in situ donde se pueda apreciar la gestión de equipos de programación a más de integrar las normas sobre buenas prácticas de programación de esta manera el Docente podrá reforzar aspectos donde el estudiante tenga falencias al desarrollar software.

BIBLIOGRAFÍA

1. AESOFT. (2012). Estudio de mercado de hardware y software en Ecuador.

2. Agiles, P. (2014). *¿Qué es Scrum?* Obtenido de http://www.proyectosagiles.org/que-es-scrum;

3. Augusto, C. (2008). *INTRODUCCIÓN A LA PROGRAMACIÓN EN JAVA.* Armenia-Colombia: Edit. Elizcom.

4. Beck, K. (2004). *Extreme Programming Explained: Embrace Change.* Addison-Wesley .

5. DATAMONITOR. (2014). *http://s3.amazonaws.com/zanran_storage/www.nif.kz/ContentPages/2486969159.pdf*.

6. Group, T. S. (2012). *CHAOSManifesto*. Obtenido de https://cs.calvin.edu/courses/cs/262/kvlinden/resources/CHAOSManifesto2012.pdf

7. PMBOK. (2004). *Guía de los Fundamentos de la Dirección de Proyectos .* Tercera Edicion.

8. Pymes. (Octubre de 2013). Desarrollo Software. (C. Rey, Entrevistador)

9. Sánchez, C. (2011). *METODOLOGÍA DE LA INVESTIGACIÓN CIENTÍFICA Y TECNOLÓGICA.* Madrid: Edit. Díaz de Santos S.A.

10.Schwaber. (1995). *SCRUM Development Process.*

11. Scrum. (2014). *Gestión de proyectos Scrum Manager.* Scrum Manager.

13. Services, T. C. (2010). Proyecto de plan de mejora competitiva del sector de software en Ecuador.

14. Sutherland et at. (2001). *Manifiesto por el Desarrollo Ágil de Software*. Obtenido de http://agilemanifesto.org/iso/es/manifesto.html

ANEXOS

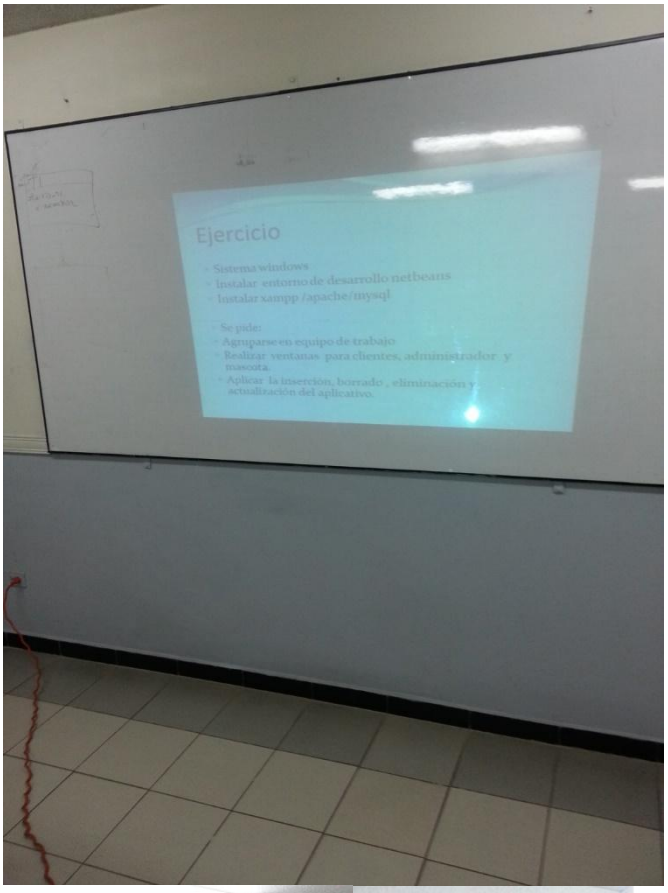
1. Encuesta

Tiene algún conocimiento teórico sobre metodologías de desarrollo de software	SI	explique cual	NO
Tiene algún conocimiento práctico sobre metodologías de desarrollo de software	SI	explique cual	NO
Ha trabajado antes en una empresa de desarrollo de software	SI	explique cual	NO
Si ha trabajado la empresa utilizaba metodologías de desarrollo de software	SI	explique cual	NO
Al programar en una aplicación realiza comentarios de la funcionalidad en una aplicación	SI	explique cual	NO
En una aplicación cuando el código donde programa es obsoleto (no sirve) . Usted	Borra	Inhabilita	Modifica
Ha trabajado Ud. Desarrollando software en equipo	SI	explique cual	NO
Tiene conocimientos de java	Básico	Medio	Avanzado
Tiene conocimiento de base de datos	Básico	Medio	Avanzado
Al programar en una aplicación utiliza buenas prácticas de programación	SI	explique cual	NO

Cuando Ud. Programa. la aplicación está orientada a satisfacer las necesidades	Aplicación	Proyecto	Cliente
Cuando programa cuales son los problemas al desarrollar software			
Que tipos de pruebas utiliza Ud. Para demostrar que su aplicativo sirve?			
En qué tiempo estimado (horas laborables) entregaría un aplicativo de escritorio que permita insertar, borrar, eliminar y actualizar; las pantallas son Administración, clientes, servicios, proveedores			
¿El desarrollo de software es en sí, es un proceso complejo?	SI	explique	NO
El desarrollo de una metodología de software elimina los riesgos	Si	Reduce	No
Usted ha sido designado como ingeniero en jefe de desarrollo, necesita realizar un aplicativo de escritorio y web sobre administración de clientes para una empresa, la empresa tiene un plazo de tres meses para entregar el producto. Qué tipo	1.Modelo lineal secuencial o cascada 2.El modelo de desarrollo rápido de aplicaciones 3.El modelo incremental en espiral		

de metodología utilizaría	4,El modelo de construcción de prototipos
Las métricas de un proyecto-software se utilizan por un lado para minimizar la planificación de desarrollo y por otro, para evaluar la calidad de los productos. Las métricas, son medidas o datos que se levantan durante el proceso y sirven para tomar decisiones en torno al mismo. Las métricas del proyecto de software sugieren que los proyectos deben medir varios aspectos. De las siguientes opciones seleccione dos que indican aspectos que **No son pertinentes** medir dentro de un proyecto de software:	1 Los recursos necesarios para el desarrollo de software 2. La capacidad de endeudamiento del cliente 3. La efectividad de las entregas y compromisos con el cliente 4. La oportunidad de generar más proyectos para el cliente

2. Fotografías del curso de desarrollo ágil


Ejercicio
- Sistema windows
- Instalar entorno de desarrollo netbeans
- Instalar xampp /apache/mysql
- Se pide:
- Agruparse en equipo de trabajo
- Realizar ventanas para clientes, administrador y mascota.
- Aplicar la inserción, borrado , eliminación y actualización del aplicativo.

Diapositivas de capacitación y talleres se adjuntan en el cd respectivo del trabajo de titulación respectivo.