UNIVERSIDAD TÉCNICA DEL NORTE

Facultad de Ingeniería en Ciencias Aplicadas

Carrera de Software

Comparación de la eficacia entre tres metodologías ágiles en el desarrollo de software basada en el estándar de la ISO/IEC 25022.

Trabajo de grado previo a la obtención del título de Ingeniero de Software presentado ante la ilustre Universidad Técnica del Norte.

Autor:

Sr. Juan Diego Iza Fuentes

Director:

PhD. José Antonio Quiña Mera.

Ibarra – Ecuador

2024

UNIVERSIDAD TÉCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	1003737978		
APELLIDOS Y NOMBRES:	IZA FUENTES JUAN DIEGO		
DIRECCIÓN:	IBARRA, AV. PÉREZ GUERRERO Y SÁNCHEZ Y CIFUENTES		
EMAIL:	jdizaf@utn.edu.ec jizafuentes@gmail.com		
TELÉFONO FIJO:	NINGUNO	**TELÉFONO MÓVIL:**	0959617565

DATOS DE LA OBRA	
TÍTULO:	COMPARACIÓN DE LA EFICACIA ENTRE TRES METODOLOGÍAS ÁGILES EN EL DESARROLLO DE SOFTWARE BASADA EN EL ESTÁNDAR DE LA ISO/IEC 25022.
AUTOR(ES):	IZA FUENTES JUAN DIEGO
FECHA:	29/07/2024
PROGRAMA:	PREGRADO
TÍTULO POR EL QUE OPTA:	INGENIERO DE SOFTWARE
DIRECTOR:	PhD. ANTONIO QUIÑA.
ASESOR 1:	ING. MAURICIO REA, MSC.

2. CONSTANCIAS

2. CONSTANCIAS

El autor manifiesta que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es el titular de los derechos patrimoniales, por lo que asume la responsabilidad sobre el contenido de la misma y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 29 días del mes de julio de 2024

EL AUTOR:

ESTUDIANTE

IZA FUENTES JUAN DIEGO

C.I 1003737978

CERTIFICACIÓN DIRECTOR

Ibarra 29 de julio del 2024

CERTIFICACIÓN DIRECTOR DEL TRABAJO DE TITULACIÓN

Por medio del presente yo PhD. Antonio Quiña, certifico que el Sr. Juan Diego Iza Fuentes portador de la cedula de ciudadanía número 1003737978, ha trabajado en el desarrollo del proyecto de grado "Comparación de la eficacia entre tres metodologías ágiles en el desarrollo de software basada en el estándar de la ISO/IEC 25022.", previo a la obtención del Título de Ingeniero en Software realizado con interés profesional y responsabilidad que certifico con honor de verdad.

Es todo en cuanto puedo certificar a la verdad

Atentamente

PhD. José Antonio Quiña Mera

DIRECTOR DE TRABAJO DE GRADO

DEDICATORIA

Dedico el presente proyecto de titulación a mi madre (Lorena) y a mi padre (Juan) por su gran sacrificio que han realizado en el trayecto de mi vida universitaria, a mi hermano (Juan Carlos) por ser un pilar fundamental en mi vida y a todos mis familiares que me han apoyado y han hecho los días más amenos y divertidos, considero que disfrutar los momentos en familia son el combustible para mantener mi calma y fortaleza y no sentirme solo, además de perseverar por mis sueños hasta alcanzarlos. Gracias a todos los que me rodean por apreciarme como soy.

Juan Diego Iza

AGRADECIMIENTO

Agradezco principalmente a mis padres y hermano, por su apoyo incondicional, creer en mí y alentarme a seguir adelante cada día.

Agradezco a mis compañeros, el transcurso de la carrera fue muy placentera y enriquecedora gracias a las largas horas de estudio y ocio que compartimos en este tramo y te ayudan a mejorar como persona y profesional.

Agradezco a mi director, PhD. Antonio Quiña, por el apoyo y seguimiento para culminar y poder presentar el trabajo de titulación.

A mi asesor, MSc. Mauricio Rea, por su tiempo, enseñanzas y observaciones que fueron muy valiosas para la corrección y excelencia del trabajo de titulación.

Juan Diego Iza

TABLA DE CONTENIDOS

DEDICATORIA 5
AGRADECIMIENTO 6
ÍNDICE DE FIGURAS 11
ÍNDICE DE TABLAS 13
ÍNDICE DE ANEXOS 15
RESUMEN 16
ABSTRACT 17
INTRODUCCIÓN 18
Tema 18
Antecedentes 18
Situación Actual 18
Prospectiva 19
Planteamiento del problema 20
Objetivos 22
Objetivo General 22
Objetivos Específicos 22
Alcance 22
Metodología 23
Elaboración: Propia. 25
Justificación 25
Justificación Tecnológica 25
Justificación Empresarial 26
Justificación Institucional 26
Justificación Educativa 26
CAPÍTULO 1 27
Marco Teórico 27

1.1 Revisión de literatura 27
1.1.1 Unidad de análisis 27
1.1.2 Preguntas de investigación.....28
1.1.3 Búsqueda de documentos 28
1.1.4 Selección de artículos 29
1.1.5 Extracción de la información.....32
1.1.6 Resultados de la revisión de la literatura 33
1.2 Metodologías de desarrollo de software 35
1.2.1 Evolución de las metodologías y modelos utilizados en el desarrollo de software 36
1.2.2 Estructura de las metodologías de desarrollo de software.....37
1.2.3 Comparativa de metodologías ágiles y tradicionales 37
1.2.4 Tipo de metodologías de desarrollo de software 38
1.3 Metodología de desarrollo ágil 39
1.3.1 Definición 40
1.3.2 Principios y valores del manifiesto ágil 40
1.3.3 Ventajas y desventajas de las metodologías ágiles.....42
1.4 Marco de trabajo SCRUM 42
1.4.1 Ventajas y desventajas de Scrum.....42
1.4.2 Características de SCRUM 43
1.4.3 Impacto de Scrum en el desarrollo de software.....46
1.5 Marco de trabajo Extreme Programming.....48
1.5.1 Ventajas y desventajas.....48
1.5.2 Características 49
1.5.3 Impacto de XP en el desarrollo de software 53
1.6 Marco de trabajo KANBAN.....54
1.6.1 Ventajas y desventajas.....55
1.6.2 Características 56
1.6.3 Impacto de Kanban en el desarrollo de software 58

1.7 Modelo de calidad interna de la ISO/IEC 25000 60
1.7.1 División modelo SQuaRE 60
1.7.2 Calidad del software 61
1.7.3 Normas ISO/IEC para la calidad del uso 61
1.7.4 Norma ISO/IEC 25022 62
1.8 Trabajos relacionados 65
CAPÍTULO 2 70
Desarrollo del proyecto 70
2.1 Introducción 70
2.2 Definición del Modelo de Calidad en Uso 70
2.3 Diseño del experimento controlado para comparar la eficacia entre las metodologías ágiles SCRUM, XP y Kanban 73
2.3.1 Objetivo 73
2.3.2 Variables 74
2.3.3 Población 76
2.3.4 Diseño 76
2.3.5 Tareas 77
2.3.6 Instrumentos 93
2.4 Operacionalización del experimento 94
2.4.1 Muestra 94
2.4.2 Preparación 94
2.5 Recolección de resultados 96
CAPÍTULO 3 97
Validación de resultados 97
3.1 Medición del Modelo de Calidad en Uso 97
3.2 Análisis de resultados 98
3.2.1 Encuesta SUS 98
3.2.2 Análisis de resultados Scrum 100
3.2.3 Análisis de resultados Extreme Programming 104

3.2.4 Análisis de resultados Kanban 109
3.3 Evaluación del modelo de calidad en uso 114
3.3.1 Evaluación de la característica Eficacia.....114
3.3.2 Evaluación de la característica Satisfacción 121
3.4 Resultados del modelo de calidad en uso 129
3.4.1 Resultados Scrum 129
3.4.2 Resultados Kanban 130
3.4.3 Resultados Extreme Programming 130
3.4.4 Análisis de resultados del modelo de calidad en uso.....131
CONCLUSIONES.....133
RECOMENDACIONES.....134
BIBLIOGRAFÍA 135
ANEXOS 143

ÍNDICE DE FIGURAS

Figura 1 Árbol de problemas.....21

Figura 2 Alcance del proyecto.....23

Figura 3 Metodología del proyecto.....24

Figura 4 Ciclo de desarrollo de software.....36

Figura 5 El 3-5-3 de Scrum.....45

Figura 6 Prácticas de Extreme Programming.....50

Figura 7 Comparación de metodología XP 53

Figura 8 El método Kanban 58

Figura 9 ISO/IEC 25000.....60

Figura 10 Escala de evaluación.....65

Figura 11 Estructura Capítulo II 70

Figura 12 Estructura primera fase.....78

Figura 13 Estructura segunda fase 79

Figura 14 Actividad SIIU de la segunda fase 80

Figura 15 Scrum Team 81

Figura 16 Planificación de Sprints.....81

Figura 17 Historia de usuario 1 82

Figura 18 Historia de usuario 4 82

Figura 19 Sprint Backlog.....83

Figura 20 Product Backlog.....83

Figura 21 Roles de Extreme Programming 84

Figura 22 Historia de usuario 006 85

Figura 23 Historia de usuario 009 86

Figura 24 Release Planning.....86

Figura 25 Planificación horas de trabajo 87

Figura 26 Designación de responsabilidades Kanban 88

Figura 27 Creación de Tarjetas Kanban 89

Figura 28 Levantamiento de Tablero Kanban 89

Figura 29 Estructura de la tercera fase 90

Figura 30 Pair Programming 91

Figura 31 Sprint Retrospective 91

Figura 32 Limitar el trabajo 92

Figura 33 Estructura de la cuarta fase 93

Figura 34 Estructura del tercer capítulo 97

Figura 35 Resultados en la escala de medición 131

ÍNDICE DE TABLAS

Tabla 1 Preguntas de investigación de la revisión literaria. 28

Tabla 2 Selección de artículos para la revisión literaria. 29

Tabla 3 Artículos seleccionados para la revisión literaria. 30

Tabla 4 Matriz de la revisión literaria. 32

Tabla 5 Comparativa de metodologías ágiles y tradicionales 38

Tabla 6 Modelo de calidad en uso 62

Tabla 7 Definición Modelo de Calidad en Uso 71

Tabla 8 Medidas de eficacia y satisfacción 71

Tabla 9 Variable dependiente e independiente 74

Tabla 10 Diseño factorial cruzado 77

Tabla 11 Ejecución del experimento 95

Tabla 12 Resultados Scrum 100

Tabla 13 Porcentajes de actividad - Scrum 101

Tabla 14 Resultados Extreme Programming 104

Tabla 15 Porcentajes de actividad - Extreme Programming 105

Tabla 16 Resultados Kanban 109

Tabla 17 Porcentajes de actividades - Extreme Programming 110

Tabla 18 Métrica medir el rendimiento del usuario – Scrum 114

Tabla 19 Métrica medir el rendimiento del usuario – Extreme Programming 116

Tabla 20 Métrica medir el rendimiento del usuario – Kanban 117

Tabla 21 Métrica de rendimiento del usuario - Scrum 119

Tabla 22 Métrica de rendimiento del usuario - Kanban 120

Tabla 23 Métrica de rendimiento del usuario - XP 121

Tabla 24 Peso respuestas 121

Tabla 25 Resultados SUS Utilidad - Scrum 122

Tabla 26 Resultados SUS Utilidad - Kanban 123

Tabla 27 Resultados SUS Utilidad - XP 124

Tabla 28 Resultado SUS Comodidad – Scrum.....124

Tabla 29 Resultado SUS Comodidad – Kanban.....126

Tabla 30 Resultado SUS Comodidad – XP.....127

Tabla 31 Resultado evaluación eficacia y satisfacción – Scrum.....129

Tabla 32 Resultado evaluación eficacia y satisfacción – Kanban.....130

Tabla 33 Resultado evaluación eficacia y satisfacción – XP.....131

Tabla 34 Resultados Scrum, Kanban y XP.....132

ÍNDICE DE ANEXOS

Anexo 1 Requerimientos del Proyecto de Aula 143

Anexo 2 Búsqueda de artículos 143

Anexo 3 Capacitación a estudiantes de la asignatura de Construcción de Software.....144

Anexo 4 Revisión de avances.....144

Anexo 5 Encuesta de Usabilidad I 145

Anexo 6 Encuesta de Usabilidad II 145

Anexo 7 Encuesta de Usabilidad III 146

Anexo 8 Encuesta de Usabilidad IV.....146

Anexo 9 Encuesta de Usabilidad V.....147

Anexo 10 Identificación de reporte de similitud 148

RESUMEN

El desarrollo de una aplicación informática abarca actividades como la programación, levantamiento de requerimientos, implementación de requisitos, entre otros. Para una correcta construcción de software se utilizan las metodologías las cuales se presentan como una solución para mejorar la calidad del software, esto implica el uso de artefactos, utilizar técnicas, aplicar prácticas y reglas utilizadas para guiar el proceso de desarrollo de software desde su concepción hasta su entrega.

El presente trabajo, titulado “Comparación de la eficacia entre tres metodologías ágiles en el desarrollo de software basada en el estándar de la ISO/IEC 25022” compara las metodologías Scrum, Kanban y Extreme Pramming (XP) con respecto a la eficacia y la satisfacción.

El primer capítulo se centra en el desarrollo de un marco teórico para el conocimiento de los conceptos de cada metodología y tener más comprensión con el experimento controlado y su ejecución.

El segundo capítulo se enfoca en el diseño de un modelo de calidad de uso basado en la ISO/IEC 25022, se utilizó un diseño factorial cruzado para realizar el experimento controlado.

El tercer capítulo realiza un análisis detallado de los resultados obtenidos, proporcionando una interpretación rigurosa de estos. Se inspecciona minuciosamente cómo el uso de cada metodología proporciona un impacto en el desarrollo del experimento controlado.

La investigación finaliza con una recapitulación de los principales hallazgos, recomendaciones para futuras investigaciones y reflexiona sobre la importancia del uso de metodologías en el ámbito empresarial y educativo.

Palabras Clave: Metodología, Scrum, XP, Kanban, Experimento Controlado.

ABSTRACT

The development of a software application includes activities such as programming, the gathering of requirements, implementation, integration of different components, among others. For a correct software construction, methodologies are used, which are presented as a solution to improve software quality. This implies the use of artifacts, techniques, practices and rules used to guide the software development process from its conception to its delivery.

This paper, entitled “Comparison of the effectiveness between three agile methodologies in software development based on the ISO/IEC 25022 standard” compares Scrum, Kanban and XP methodologies with respect to effectiveness and satisfaction.

The first chapter focuses on the development of a theoretical framework for the knowledge of the concepts of each methodology and to have more understanding with the controlled experiment and its execution.

The second chapter focuses on the design of a quality of use model based on ISO/IEC 25022, a crossed factorial design was used to perform the controlled experiment.

The third chapter performs a detailed analysis of the results obtained, providing a rigorous interpretation of these. It thoroughly inspects how the use of each methodology provides an impact on the development of the controlled experiment.

The research ends with a recapitulation of the main findings, recommendations for future research and reflects on the importance of the use of methodologies in business and education.

Keywords: Methodology, Scrum, Extreme Programming, XP, Kanban, controlled experiment.

INTRODUCCIÓN

Tema

Comparación de la eficacia entre tres metodologías ágiles en el desarrollo de software basada en el estándar de la ISO/IEC 25022.

Antecedentes

Actualmente, el software se encuentra en la mayor parte de las actividades humanas: industria, el comercio, educación, finanzas, gobierno e incluso en la salud. La importancia de realizar software con calidad es de suma importancia en el ámbito industrial y el desarrollo de la sociedad, sin embargo, desde la perspectiva de normativas y estándares hacia el producto, la sobreabundancia de información, el alto costo y el acceso limitado a esta información, impiden un acercamiento de estos a los ingenieros de software en pro-calidad del producto(Vaca & Jácome, 2018).

En la Universidad Técnica del Norte, específicamente en la carrera de software cada semestre se desarrollan proyectos de aula que involucran la auto organización, auto planificación y la auto suficiencia de cada grupo involucrado en cada asignatura, en base a esto los sujetos se enfrentan a desafíos de cómo construir un sistema informático, sin aprovechar estrategias del uso de una metodología que permita satisfacer las necesidades de tener una guía de cómo desarrollar software desde su inicio hasta su entrega, además de ayudar a gestionar las actividades que cada integrante debe desarrollar, por lo cual es importante adoptar una metodología que se acople al proyecto de software que se planea desarrollar.

Situación Actual

En el presente existen empresas que tienen falencias en la adecuada producción de software, debido a que no cuentan con un modelo que permita asegurar la calidad del mismo (Roa et al., 2015). La estructura de los modelos de calidad se compone de diversos criterios

que son evaluados por métricas, con el propósito de reducir la subjetividad en la asignación de un valor, ya sea cuantitativo o cualitativo (Alarcón-Aldana et al., 2014), una de estas métricas es la calidad en uso. Además, tenemos que el buscar un valor agregado al producto de software es lo que impulsa a la aplicación de normas de calidad, y a la vez brindar productos con altos estándares de calidad y con una disminución de fallos (Mera Paz, 2016). Aquí nace la importancia de usar un modelo de calidad en uso para comparar metodologías como Scrum, Extreme Programming y Kanban.

Scrum , Kanban y XP se ofrecen como una alternativa a otras metodologías utilizadas en el ámbito empresarial tanto tradicionales como ágiles, y para comprender las diferencias de eficacia y satisfacción, es que se necesita estudios que contrasten el esfuerzo de perspectivas sobre el desarrollo e implementación de estas metodologías en ámbitos educativos, en este sentido se ha revisado bases de datos bibliográficas que verifiquen si existe un estudio similar al nuestro y dando como resultado que no existe registro, en base a esto se implementará un modelo de calidad en uso para de manera objetiva comparar entre estas metodologías y ver cuál es más eficaz.

Prospectiva

El proyecto de investigación plantea desarrollar un experimento controlado como estrategia pedagógica para potenciar la autogestión de los estudiantes en el desarrollo de software para tanto como dentro de las aulas como fuera de ellas.

Se trata de un experimento controlado basado en un diseño factorial cruzado dónde los sujetos de prueba son sometidos a tratamientos en los que, cada tratamiento es una condición experimental única que se aplica a los sujetos en un orden determinado para evaluar sus efectos individuales y sus posibles interacciones(Vegas et al., 2016).

Este experimento no solo busca mejorar el proceso de desarrollo de software, sino que también simula prácticas de cómo manejar proyectos desde cero, permitiendo a los estudiantes desarrollar la autogestión, autosuficiencia y autoorganización.

En consecuencia, esta propuesta aspira a mejorar el proceso de desarrollo de aplicaciones informáticas en la Carrera de Software, con esto se obtiene una caracterización y comparación de metodologías en el ámbito educativo que permita una manera más eficaz de gestionar estos proyectos.

Planteamiento del problema

Las metodologías ágiles han marcado en los últimos años una tendencia en su adopción en las organizaciones dedicadas al desarrollo de software. Estudios recientes realizados por el (WQR, 2022) indica que la adopción de métodos ágiles para el desarrollo ha incrementado, siendo que el 83% de las empresas usan metodologías ágiles para la creación de sus aplicaciones y que éstas les permiten adaptarse mejor a los cambios del mercado, además reveló que el 64% de los encuestados citó la entrega a tiempo como la mayor mejora. La reducción del coste de la calidad fue otra mejora clave (62%), seguida de cerca por la mejora de la experiencia del cliente (61%).

Según (E. G. Maida & Pacienzia, 2015) el desarrollo de software presenta algunos problemas que pueden involucrar la falta de planificación de tareas o actividades, determinar con precisión la cantidad de tiempo y dinero, organizar una interacción con el equipo de trabajo, irregularidades en cuanto a la documentación o revisión de avance. Según (Villarreal, 2013) menciona que aparecen las metodologías ágiles que aportan en el aumento de la productividad, alcanzar resultados más eficaces y la adaptabilidad, son algunos de los beneficios que contribuyen los marcos de trabajo ágiles.

En base a la encuesta realizada a los estudiantes de la asignatura de Construcción de Software del período ABRIL – AGOSTO 2023, en el entorno de desarrollo de aplicaciones, la carrera de Software de la Universidad Técnica del Norte, se encuentran problemas como un bajo nivel de adopción de metodologías como XP (35,45%) y Kanban (14,35%) en los proyectos de aula, índice de estudiantes sin conocimientos de los valores y principios del manifiesto ágil (21,4%), índice regular acerca del conocimiento de la metodología XP(78%) y

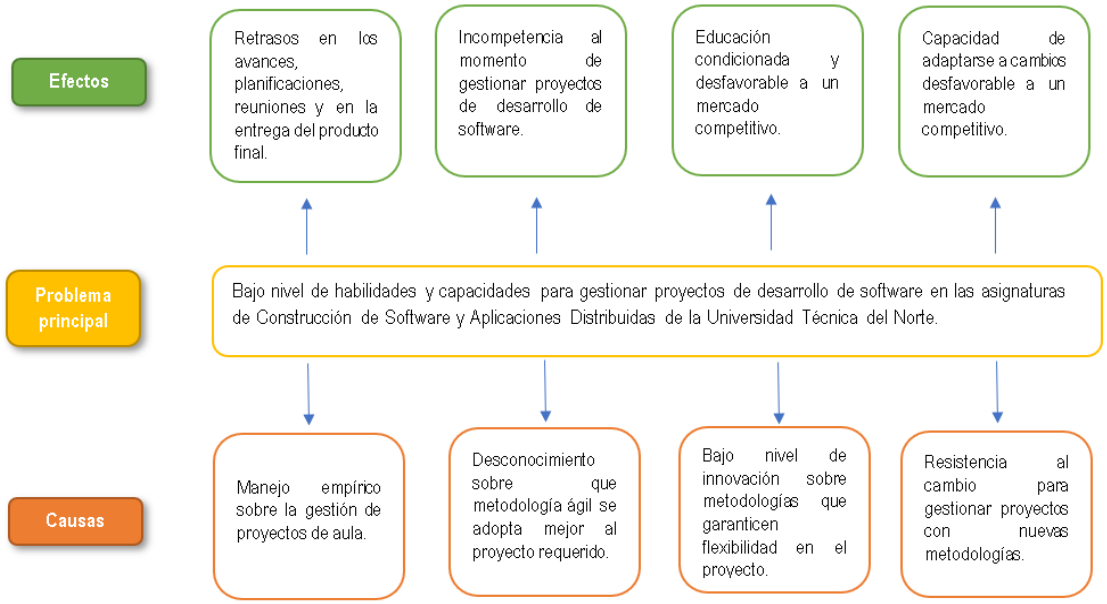
Kanban (92%), falencias en la adaptabilidad de nuevas metodologías (50%) y poca versatilidad al momento de seleccionar marcos de trabajo ágiles (50%) (CSOFT - UTN, 2023).

Asimismo, los estudiantes de la CSOFT – Universidad Técnica del Norte (UTN) presentan estos inconvenientes al gestionar sus proyectos de desarrollo de software y deciden optar por Scrum como su marco de trabajo ágil sin saber qué tipo de metodología se puede acoplar al contexto que lo requieran. Para desarrollar software es de suma importancia la elección de una metodología en base a esto se realizará experimento controlado nos permitirá clarificar el panorama de eficacia de las metodologías ágiles como Scrum, XP y Kanban de manera objetiva en la construcción de un software, teniendo en cuenta que medir la calidad (interna) de un producto de software involucra evaluar el modelo para alcanzar a este (Roa et al., 2015).

A continuación, en la Figura 1 se detalla el árbol de problemas del proyecto:

Figura 1

Árbol de problemas



Objetivos

Objetivo General.

Comparar la eficacia de las metodologías ágiles en el proceso de desarrollo de software basada en la ISO/IEC 25022.

Objetivos Específicos

- Caracterizar las metodologías ágiles a utilizar en los proyectos de aula.
- Realizar un experimento controlado para comparar la eficacia de las metodologías ágiles en el proceso de desarrollo de software basada en las ISO/IEC 25022.
- Analizar los resultados del experimento controlado.

Alcance

La investigación propuesta va encaminada a realizar una comparación entre las metodologías ágiles: Scrum, Extreme Programming (XP) y Kanban para determinar una mejor adopción de marcos de trabajo ágiles eficaces para el desarrollo de proyectos de la carrera de Software de la Universidad Técnica del Norte. La medición y evaluación del proyecto, estará basado en un modelo de calidad en uso de productos de software de la serie de normas ISO/IEC 25022, específicamente en la subcaracterística de la eficacia.

El experimento controlado se ejecutará a base de una prueba de concepto implementado en los proyectos de aula de la asignatura de Construcción de Software. En donde se empleará el uso de las tres metodologías ágiles definidas para los proyectos de desarrollo de software. Para lo cual (Patilla et al., 2021) mencionan que se debe seleccionar una metodología ágil para gestionar el proceso de desarrollo de software y obtener un flujo óptimo de trabajo dentro del marco de trabajo propuesto.

La comparación entre las metodologías se efectuará mediante la ejecución del experimento controlado enfocado en las asignaturas seleccionadas, el cual constará de las siguientes fases:

1) Diseño del experimento, en dónde se caracterizará y se definirá la manera en la que se realizará el experimento.
2) Operación del experimento, se ejecutará el diseño del experimento en donde se enseñará las metodologías Scrum, XP y Kanban a los sujetos de acuerdo con el diseño del experimento.
3) Análisis de resultados, se examinará los resultados obtenidos en la operación del experimento para evaluar, comparar y concluir la información del estudio.

A continuación, en la Figura 2 se evidencia el proceso para efectuar el alcance:

Figura 2

Alcance del proyecto

Elaboración: Propia.

Metodología

El enfoque de esta indagación es cuantitativo, ya que al medir podemos tener un mejor rendimiento y control de nuestro proyecto de software en base a características normalizadas que permitan validar y garantizar un trabajo de calidad (Chauhan, 2014).

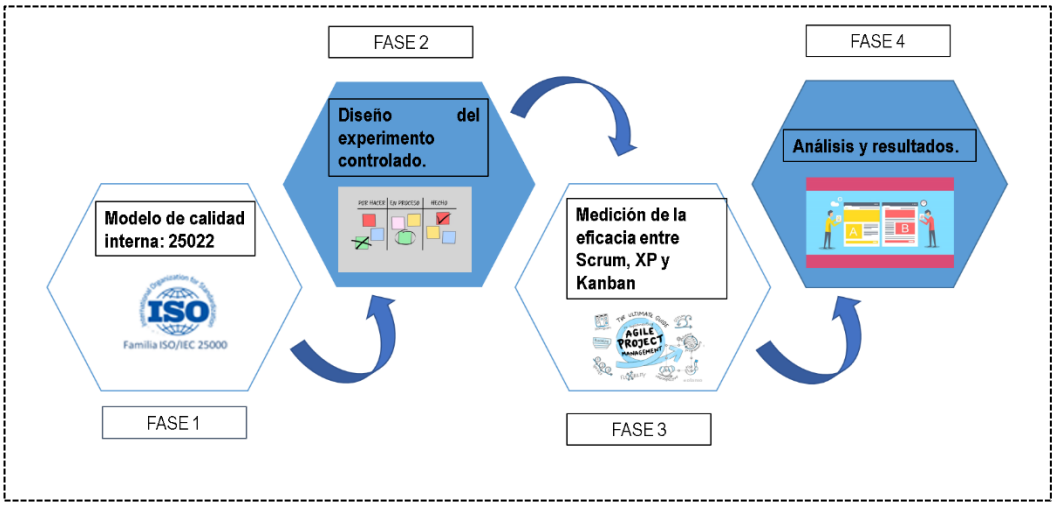
Para cumplir con el primer objetivo que corresponde a la caracterización de las metodologías seleccionadas, se hará uso de artículos, estudios, recursos y capacitaciones que brinden información, tareas y detallen claramente el proceso que implica cada marco de trabajo ágil definido, evidenciando las características que diferencia una metodología de otra. Además, se precisará la adopción del manifiesto ágil en este proyecto ya que según Cadavid et al. (2013) estas metodologías son flexibles, permiten una comunicación iterativa con el cliente, retroalimentación, entregables constantes y es más aceptada en la industria de desarrollo de software.

Para cumplir con el segundo objetivo el marco teórico tendrá un eje investigativo de manera cualitativa y analítica en cuanto a las métricas de eficacia de desempeño que está en la ISO/IEC 25022, la cual nos brinda el cómo evaluar los respectivos parámetros. Para poder evidenciar las diferencias de la eficacia de las metodologías Scrum, XP y Kanban, se realizará un experimento controlado el cual según Wohlin et al. (2003) constará de un diseño, operación y análisis de resultados.

Finalmente, para cumplir con el tercer objetivo, el experimento desarrollado será evaluado bajo el estándar de la norma ISO/IEC 25022 mediante las métricas seleccionadas de la subcaracterística de eficacia. Además, por medio del uso de documentos y recopilación de información se analizará los datos y la calidad de software, ya que Vega et al (2018) menciona que la medición del éxito o eficacia de los productos software es una actividad importante que nos ayudan a entender el valor que otorgan las inversiones que se realizan en éstos.

A continuación, la Figura 3, evidenciará el proceso que se llevará para alcanzar los objetivos propuestos:

Figura 3
Metodología del proyecto



Elaboración: Propia.

Justificación

El presente proyecto busca cumplir con el Objetivo de Desarrollo Sostenible (ODS) N°9, “Industria, Innovación e Infraestructura”, específicamente con la meta 9.5 el cual menciona sobre fomentar la investigación científica para mejorar la capacidad tecnológica de los sectores industriales, ya sea innovando o aumentando el número de personas en investigación científica (Naciones Unidas, 2016), a partir de esto, el presente proyecto busca innovar el uso de nuevos marcos de trabajo ágiles para poder generar más eficacia en cuanto al desarrollo del software para buscar ventajas competitivas con respecto a otras industrias (De La Cruz, 2022).

En lo que respecta al proyecto, se rige en lo establecido en la norma ISO 25022 acerca de la evaluación de la eficacia, lo que implica medir el grado de datos que tienen atributos que pueden ser procesados y proporcionados con los niveles de rendimiento esperados mediante el uso de cantidades y tipos adecuados de recursos en un contexto de uso determinado (ISO, 2016).

Justificación Tecnológica

La gestión de proyectos de desarrollo de software es un tema ampliamente indagado debido a que se busca erradicar el fracaso en proyectos tecnológicos. En el siguiente trabajo

se buscará profundizar el uso de metodologías ágiles que han sido probadas en empresas de desarrollo como Google.inc y su impacto de la eficacia en el éxito de este. En base a esto se investiga tres metodologías Extreme Programming, Scrum y Kanban siendo estas las que registran más uso (Bowes, 2015), por ejemplo, en el caso de Kanban que es usado por Toyota (Renata & Centeno, 2015), en el caso de Scrum que es empleado por IBM (Randall, 2014) y XP que es implementado por Microsoft (Cusumano, 2007).

Justificación Empresarial

La investigación presente busca tener un impacto en el estudiante al impulsar los valores y principios del manifiesto ágil, de manera que la persona desarrolle capacidades para manejar variables empresariales tales como planificación, trazabilidad del proyecto, flexibilidad, adaptación a los cambios, reducción de riesgos y gestión de proyectos de desarrollo de software adoptando la metodología más eficaz.

Justificación Institucional

El trabajo busca apoyar a la Carrera de Software de la Universidad Técnica del Norte en su visión de "(…) formar profesionales de excelencia críticos, líderes y emprendedores; que generen, fomenten y ejecuten procesos de conocimientos científicos, tecnológicos y de innovación (…) " (Carrera de Software - UTN, 2023).

Justificación Educativa

La investigación presente busca tener un impacto en la educación de los estudiantes de la Carrera de Software de la UTN de modo que mejoren sus capacidades y destrezas de cómo gestionar proyectos de aula adoptando metodologías eficaces que se adecúen al tipo de trabajo requerido.

CAPÍTULO 1

Marco Teórico

En este capítulo, se ha establecido un marco conceptual con el propósito de fundamentar teóricamente los elementos necesarios para llevar a cabo la comparación de la eficacia en entre las metodologías ágiles Scrum, XP y Kanban. Este marco se basa en un modelo de calidad en uso que hace uso de la familia de estándares de la ISO/IEC 25000 específicamente la norma ISO/IEC 25022.

1.1 Revisión de literatura

Una revisión bibliográfica (RL) es un proceso metódicamente riguroso de recopilación, evaluación y síntesis de información sobre un tema de investigación a partir de la literatura académica sobre ese tema (Kitchenham et al., 2009). Se sugieren los siguientes pasos para realizar una revisión bibliográfica (Miswar et al., 2018):

a) Preguntas de investigación.

b) Investigación de documentos.

c) Selección de artículos.

d) Extracción de datos pertinentes

e) Resultados de la revisión de literatura

1.1.1 Unidad de análisis

Las unidades analíticas son el foco principal de un estudio debido a su importancia para la pregunta o hipótesis de investigación. He aquí la cuestión: *Comparación de la eficacia entre tres metodologías ágiles en el desarrollo de software basada en el estándar de la ISO/IEC 25022.*

1.1.2 Preguntas de investigación

Con el fin de orientar la revisión de la literatura sobre el tema del proyecto de tesis, se formularon cuatro preguntas de investigación (PI), que se detallan en la Tabla 1:

Tabla 1
Preguntas de investigación de la revisión literaria.

No.	Preguntas de investigación	Motivación
PI1	¿Cuál es la importancia del uso de una metodología ágil en la gestión de un proyecto de desarrollo de software?	Comprender el impacto que genera una metodología ágil en la gestión de un proyecto de software.
PI2	¿Qué metodologías ágiles son las más usadas en proyectos de desarrollo de software?	Identificar que metodologías ágiles son las más populares en el mercado.
PI3	¿Cuáles son los beneficios y desafíos que conllevan utilizar la metodología ágil más apta para el proyecto?	Conocer que metodología ágil es más adecuada para el proyecto que se requiere.
PI4	¿Cómo se puede medir la eficacia o efectividad de una metodología ágil en un proyecto de desarrollo de software?	Determinar indicadores, herramientas y submétricas disponibles para medir la calidad de uso de software.

Elaboración: Propia.

1.1.3 Búsqueda de documentos

Para responder a las preguntas de investigación planteadas, es necesario recopilar la información necesaria durante la fase de búsqueda documental. Esta etapa se inicia con la selección de base de datos bibliográficas incluyendo Web of Science, IEEE Xplore, ScienceDirect, Google Scholar y repositorios digitales universitarios de Ecuador. Las cadenas de búsqueda posteriores son las siguientes:

(“software devolopment methodologies” OR “software development”) AND (“manifiesto ágil” OR “methodologies agile”) AND (“scrum” OR “XP” OR “extreme programming” OR “Kanban”) AND (“quality of software use “OR “calidad del uso de software”) AND (“metodologias ágiles” OR “agile development”)

1.1.4 Selección de artículos

Se tuvieron en cuenta tres fases principales a la hora de seleccionar los artículos y trabajos más aplicables para responder a las preguntas de la investigación.

En la primera fase del proceso de selección de artículos se utilizaron criterios de inclusión y exclusión. Los criterios de selección se centraron en elegir artículos científicos revisados por pares, de alta calidad, recientes (publicados en los últimos cinco años) así como obras tituladas que estuvieran estrechamente relacionadas con los campos de ingeniería del software, construcción del software y la tecnología. Los criterios de exclusión, por su parte, incluían aspectos como trabajos duplicados y estudios publicados en campos del conocimiento no relacionados.

En la segunda etapa, se revisaron los resúmenes y el cuerpo de cada artículo y trabajo para comprobar que eran pertinentes para las preguntas de investigación planteadas en la primera etapa.

Por último, en la tercera etapa, se utilizaron criterios de lectura de calidad y solo se eligieron publicaciones en revistas clasificadas como Journal ranking Q1 y Q2 según la evaluación de SJR Scimagojr. Los resultados del procedimiento de selección de artículos se muestran en la Tabla 2.

Tabla 2
Selección de artículos para la revisión literaria.

Base de datos	Fase 1	Fase 2	Fase 3
Web of Sience	4	2	0
IEEE Xplore	20	17	15

Google Scholar	23	18	13
ScienceDirect	1	0	0
Scopus	5	2	1
Total	**53**	**39**	**29**

Al finalizar las fases se seleccionaron 28 referencias que se muestran en la Tabla 3.

Tabla 3
Artículos seleccionados para la revisión literaria.

Código	Título	Autor	Año
A1	Work breakdown structures (wbs) in the scope management of software development projects.	Puentes R., Parada J., & Pabón L	2023
A2	La estructura de desglose del trabajo como mecanismo viable para la generación de proyectos exitosos.	Fonseca B., Matamoros L., Hernández A.	2019
A3	A survey on Adopting Agile Software Development: Issues & Its impact on Software Quality.	Ruk, Ali K., Muhammad & Khan.	2019
A4	Using Conceptual Models in Agile Software Development: A Possible Solution to Requirements Engineering Challenges in Agile Projects.	Gupta Abhimanyu, Poels Geert & And Palash	2022
A5	Effective distributed pair programming.	Rajpal M.	2018
A6	An Intelligent Recommender and Decision Support System (IRDSS) for Effective Management of Software Projects.	Hamid M., Zeshan F., Ahmad A., Khan Z., Munwar S., & Aljuaid H.	2020
A7	A catalogue of agile smells for agility assessment.	Telemaco U., Oliveira T., & Cowan D.	2020
A8	A Simulation Model of Kanban Software Process.	Gong H., Liu B., & Shao D	2018
A9	Study of effect of Agile software development Methodology on Software Development Process.	Chaudhari Ashvini & Joshi, Shashank D.	2021
A10	Integrating the extreme programing model with secure process for requirement selection.	Singh A.	2018

A11	Análisis y comparación de las metodologías de SCRUM y según PMI gestión de proyectos.	Jaramillo A.	2021
A12	Enfoque de aplicación ágil con Scrum, Lean y Kanban Agile application approach with Scrum, Lean and Kanban.	Gaete J., Villarroel R., Figueroa I., Reyes H., & Muñoz R	2021
A13	A Comparative Study of Implementing Agile Methodology and Scrum Framework for Software Development.	Gupta N., Sharma H., Kumar S., Kumar A. & Kumar R	2022
A14	Review On Extreme Programming-XP.	Yasvi Maleeha & Yadav Sahendrasingh	2019
A15	Metodologías ágiles: un análisis de los desafíos organizacionales para su implementación.	Flores F., Sanhueza V., Valdés H., & Reyes L.	2021
A16	Evolution of the Methodologies and Models used in Software Development.	Zumba J & León C	2018
A17	Back to the future: origins and directions of the “Agile Manifesto” – views of the originators.	Hohl Philipp, Klünder Jil, van Bennekum Arie, Lockard Ryan, Gifford James & Münch Jürgen,	2018
A18	Agile Software Methodology with Scrum for Developing Quality Assurance System.	Legowo M., Budi I. & Deden P.	2019
A19	Scrum Master (3rd ed.).	Palacio M.	2022
A20	Usability Testing in Kanban Agile Process for Club Management System.	Rahmat A. & Hanifiah N.	2020
A21	Calidad de software del módulo de talento humano del sistema informático de la Universidad Técnica del Norte bajo la norma ISO/IEC 25000.	Vaca T., & Jácome A.	2018
A22	Impact of Agile Scrum Methodology on Time to Market and Code Quality - A Case Study.	Kaur K., Khurana M. & Manisha	2021
A23	Presented to the Interdisciplinary Studies Program: Implementing Agile Methodology: Challenges and	Verret Jeffrey	2018

	Best Practices.		
A24	La Guía Kanban para Scrum Teams.	Scrum.org, Vacanti D. & Yeret Y.	2021
A25	Scrum2kanban: Integrating kanban and scrum in a university software engineering capstone course.	Matthies C.	2018
A26	Optimization of the Production Management of an Upholstery Manufacturing Process Using Lean Tools: A Case Study.	Santos E., Lima T. & Gaspar P.	2023
A27	The Influence of Agile Methodology (Scrum) on Software Project Management.	Hayat F., Rehman A., Sarmad K., Wahab K. & Abbas M.	2019
A28	The standard for project management and a guide to the project management body of knowledge (PMBOK guide).	Project Management Institute	2021
A29	Quality in Use of Digital Wallet based on ISO/IEC 25022.	Pradanita W. & Rochimah S.	2020

1.1.5 Extracción de la información

De acuerdo con la unidad analítica propuesta, se elaboró una matriz para ayudarnos a seleccionar las ideas más importantes de la investigación, como se observa en la Tabla 4.

Tabla 4
Matriz de la revisión literaria.

Artículos	***Conceptos***							
Código	**Construcción de software**	**Diseño de software**	**Metodologías ágiles**	**Scrum**	**Extreme Programming**	**Kanban**	**Calidad del software**	**Norma ISO/IEC 25022**
A1	X	X	X					
A2	X	X	X					
A3	X		X					
A4				X				
A5					X			
A6			X					
A7			X					
A8						X		

A9					X				
A10						X			
A11					X				
A12							X		
A13					X				
A14						X			
A15				X					
A16				X					
A17				X					
A18					X				
A19					X				
A20							X		
A21								X	
A22					X				
A23				X					
A24							X		
A25							X		
A26							X		
A27					X				
A28	X		X						
A29									X

1.1.6 Resultados de la revisión de la literatura

De los 53 artículos o estudios relevantes escogidos en la Tabla 2, al final se seleccionaron 29 que cumplen los conceptos de estudio de la unidad de análisis los cuales corresponden a la Tabla 4, en base a esto se analizaron los trabajos seleccionados que respondan a las preguntas planteadas en la Tabla 1.

Las metodologías ágiles están ganando cada vez más terreno en el desarrollo de software debido a su capacidad para garantizar un desarrollo temprano de productos de alta calidad. Además, ofrecen la flexibilidad necesaria para adaptarse a los cambios en los requisitos del usuario, permitiendo una rápida incorporación durante el proceso de desarrollo.

Debido a sus múltiples ventajas en comparación con enfoques tradicionales más secuenciales, como el modelo en cascada, las metodologías ágiles, como Scrum, Kanban o XP, están ganando popularidad en el desarrollo de software moderno. Los marcos de trabajos ágiles están ganando terreno, principalmente porque permiten la entrega temprana y continua de software de alta calidad. Los equipos pueden obtener retroalimentación útil de los usuarios de manera regular y continuar construyendo sobre una base sólida al centrarse en desarrollar

características pequeñas pero completas de manera iterativa. Esto aumenta la probabilidad de satisfacer las necesidades reales de los usuarios.

Otra característica principal del enfoque ágil es la adaptabilidad. En proyectos largos con requisitos cambiantes a lo largo del tiempo, un enfoque secuencial puede volverse problemático. Las metodologías ágiles pueden evaluar y ajustar sus prioridades de manera continua, lo que les permite adaptarse rápidamente a los nuevos requisitos.

Como se puede apreciar en los artículos seleccionados se hace mayor énfasis al uso de las metodologías ágiles más usadas en el mercado, algunos autores como (Fernández González, 2013; Mohammed & Abushama, 2013) mencionan que las metodologías más populares en el mercado corresponden a Extreme Programming (XP), Scrum, Crystal, DSDM (Dynamic System Development Method), Feature Driven Development (FDD) y Test Driven Development (TDD). Sin embargo, algunos investigadores como (Ballesteros & Silva, 2008; Bowes, 2015; Gaete et al., 2021; Gong et al., 2018a; Matthies, 2018; Mohammed & Abushama, 2013; Patilla et al., 2021; Rahmat & Hanifiah, 2020; Renata & Centeno, 2015; Salvay, 2017; Scrum.org et al., 2021; Shamshurin & Saltz, 2019; Universidad de Ingeniería y Tecnología, 2022) también consideran a Kanban como una de las metodologías que más ha tomado fuerza en el mercado.

De acuerdo con los autores (Anand & Dinakaran, 2016) afirma que, debido a su enfoque en la mejora continua y la transparencia en el proceso de desarrollo de software, Kanban está ganando popularidad en el mercado. Su énfasis en la visualización del proceso y el flujo de trabajo asiste a los equipos en identificar desafíos y potenciar la eficiencia del proceso. Además, habilita a los equipos para abordar rápidamente las solicitudes del cliente y dar prioridad a las tareas de manera correspondiente, lo que le convierte en una elección apropiada para equipos que se dedican a proyectos de mantenimiento o soporte.

Una vez determinado las metodologías más populares, se procede a definir como medir la efectividad o eficacia de una metodología ágil en proyectos de desarrollo de software,

en este caso, la norma ISO/IEC 25022 que forma parte de la serie de estándares SQuaRE(Systems and software Quality Requirements and Evaluation) desarrollados por ISO/IEC (International Organization for Standardization/ International Electrotechnical Commision), ofrece pautas para evaluar la calidad en uso, es decir, cómo funciona un sistema o producto de software cuando los usuarios finales lo utilizan en situaciones reales. La efectividad, la eficiencia, la satisfacción y la libertad de riesgo son las cuatro características principales para medir la calidad en uso, según ISO/IEC 25022. La efectividad se refiere específicamente a la precisión y exhaustividad con la que los usuarios logran objetivos específicos mediante el uso del sistema o software.

En este caso centrándonos en la eficacia, la norma recomienda utilizar tareas completadas, objetivos logrados, los errores en una tarea, tareas con errores e intensidad de error de tareas. Una alta efectividad significa que el sistema o software permite a los usuarios completar sus tareas y objetivos de manera precisa y completa. Como resultado, el usuario final lo ve como un indicador crucial de calidad.

La norma ISO/IEC 25022 proporciona una base estandarizada para medir la calidad en uso y la efectividad, lo que permite comparar varios sistemas y productos de software. Con su implementación, se puede realizar una evaluación objetiva de calidad utilizando métricas normalizadas.

1.2 Metodologías de desarrollo de software

Hoy en día, las metodologías de ingeniería de software se consideran una base esencial para la ejecución exitosa de proyectos de desarrollo de software serios. Estas metodologías son necesarias para garantizar que un proyecto no dependa únicamente de la experiencia y habilidades de su equipo de programadores.

Tal como indican (G. Maida & Pacienzia, 2015) estas metodologías son esenciales para la exitosa conclusión de cualquier proyecto profesional, ya que proporcionan documentación detallada y permiten una comunicación de los resultados.

1.2.1 Evolución de las metodologías y modelos utilizados en el desarrollo de software

De acuerdo con (J. Zumba & León, 2018) alude que las metodologías de desarrollo de software han experimentado un proceso histórico y evolutivo que comenzó en los años 40 con la introducción de los primeros ordenadores. Aún no se habían establecido parámetros ni normas, por lo que el desarrollo de software dependía en gran medida de enfoques de ensayo y error. Esto dio lugar a lo que se conoció como la “crisis del software”, ya que muchos proyectos no cumplían las expectativas de los usuarios, sufrían retrasos en la entrega y se salían del presupuesto.

A partir de esto (Flores et al., 2021) menciona que los métodos convencionales o tradicionales surgieron en la década de los setenta. Estas metodologías se centran en la obtención de entregables del proyecto, como informes detallados, documentos y diagramas, a través de la previsión de todos los requisitos del sistema y planificación anticipada de su funcionamiento.

Desde la perspectiva de (Hohl et al., 2018) menciona que, un grupo de 17 expertos en procesos interesados en promover un desarrollo iterativo e incremental (IID) contemporáneo y sencillo se reunió en Utah en febrero de 2001 para buscar puntos en común y crear el manifiesto ágil, a partir de los conceptos de ciclos “planificar-hacer-estudiar-actuar”.

De acuerdo con (J. P. Zumba, 2018) reconocen que el impacto de la evolución de las metodologías de desarrollo de software se ha convertido en un consenso formal para la construcción de sistemas, estos modelos o marcos guían el desarrollo de un producto desde su concepción hasta su finalización, incluyendo su mantenimiento. Los autores designan que las metodologías de desarrollo de software deben tomar pautas en cuenta el ciclo básico del desarrollo de software, el cual se representa en la Figura 4.

Figura 4
Ciclo de desarrollo de software

Nota. Adaptado de (J. P. Zumba, 2018).

1.2.2 Estructura de las metodologías de desarrollo de software

Para la implementación de las metodologías de desarrollo se utilizan las Estructuras de Desglosadas de Trabajo (EDT) o Work Breakdown Structure (WBS) que según (Fonseca et al., 2019) esta herramienta de planificación es crucial porque establece una organización jerárquica de las tareas y los proyectos al dividirlos en trozos más pequeños y manejables.

Otra definición que podemos apreciar es la de (Project Management Institute, 2021), que menciona a un EDT como un desglose jerárquico de la totalidad del alcance del trabajo que un equipo de proyecto debe completar para lograr los objetivos del proyecto y desarrollar los requisitos de este.

Para dimensionar un EDT se menciona que “La estructura de la EDT se puede representar en tres niveles, donde en el segundo nivel se establecen las etapas del ciclo de vida del proyecto y el tercer nivel los entregables o productos”(Puentes et al., 2023).

1.2.3 Comparativa de metodologías ágiles y tradicionales

A continuación, en la Tabla 5 se muestra una comparación entre las metodologías ágiles y metodologías tradicionales.

Tabla 5
Comparativa de metodologías ágiles y tradicionales

Metodologías Ágiles	Metodologías Tradicionales
Basadas en heurísticas provenientes de prácticas de producción de código.	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo.
Especialmente preparados para cambios durante el proyecto.	Cierta resistencia a los cambios
Impuestas internamente (por el equipo).	Impuestas externamente.
Proceso menos controlado, con pocos principios.	Proceso mucho más controlado, con numerosas políticas/normas.
No existe contrato tradicional o al menos es bastante flexible.	Existe un contrato prefijado.
El cliente es parte del equipo de desarrollo.	El cliente interactúa con el equipo de desarrollo mediante reuniones.
Grupos pequeños (<10 integrantes) y trabajando en el mismo sitio.	Grupos grandes y posiblemente distribuidos.
Pocos artefactos.	Más artefactos.
Pocos roles.	Más roles.
Menos énfasis en la arquitectura del software.	La arquitectura del software es esencial y se expresa mediante modelos.

Nota. Adaptado de (Salvay, 2017).

1.2.4 Tipo de metodologías de desarrollo de software

En el contexto del desarrollo de software, es fundamental contar con una metodología desde la perspectiva de la ingeniería de software que pueda satisfacer las necesidades de los

usuarios que buscan una solución informática. Existen dos enfoques principales en cuanto a las metodologías de desarrollo de software: el enfoque tradicional y el enfoque ágil.

Las metodologías tradicionales, como RUP, MSF, Win – Win Spiral Model e Iconix, entre otras, hacen hincapié en la planificación meticulosa y la gestión de procesos. Sin embargo, metodologías ágiles como SCRUM, Crystal Methodologies, Dynamic Systems Development Method (DSDM), Adaptative Software Development (ASD), Feature-Driven Development (FDD) y Lean Development (LD), entre otras, proponen procesos que se adaptan al cambio y se desarrollan de forma incremental con breves iteraciones (Navarrete & Cárdenas, 2016).

El desafío que enfrentan todas las empresas en el actual entorno empresarial en constante evolución es mantener e idealmente aumentar su competitividad en el mercado. De acuerdo con (Papadopoulos, 2015) sostiene que, los enfoques tradicionales de desarrollo de software son rígidos e incapaces de adaptarse a los exigentes requisitos de los clientes, mientras que las metodologías ágiles proporcionan prácticas que permiten adaptaciones rápidas para satisfacer las necesidades del desarrollo de productos contemporáneo.

1.3 Metodología de desarrollo ágil

Desde el punto de (Mazzanti, 2012) vista de menciona que los métodos ágiles surgieron en la década de los noventa para subsanar las deficiencias y los malos resultados de los enfoques convencionales del desarrollo del software.

Desde el punto de vista de (Hamid et al., 2020) cada iteración de un proyecto ágil como un sprint, generalmente tiene una duración de entre dos y cuatro semanas. Durante este período el equipo de desarrollo diseña, implementa, prueba y entrega un nuevo producto al cliente.

De acuerdo con (A. Quiña-Mera et al., 2023) menciona que puede existir una integración con estándares de calidad como la ISO/IEC 25022, que ofrece un marco para evaluar y mejorar las características de calidad del software, puede ser beneficiosa para las

metodologías ágiles, ya que fomentan la adaptabilidad y la colaboración continua. Asegurando que las prácticas ágiles no solo se enfoquen en la entrega rápida, sino también en la sostenibilidad y la calidad del software, esta alineación puede facilitar la identificación de áreas de mejora en el proceso de desarrollo.

1.3.1 Definición

Según lo afirman (Telemaco et al., 2020), las metodologías ágiles se definen como un conjunto de prácticas y valores que priorizan la entrega continua de software de alta calidad a través de un trabajo en equipo efectivo y una comunicación constante con los clientes.

Otra definición por parte de (Verret, 2018), afirma que, una metodología ágil se define como un método incremental e iterativo de gestión de proyectos de ingeniería y TI. Este método requiere trabajar en colaboración con los clientes y la dirección en periodos cortos e iterativos para lograr hitos incrementales hacia el objetivo de entrega.

Scrum, Extreme Programming (XP), Kanban y Lean son sólo algunas de las metodologías más conocidas. Aunque cada una de estas metodologías tiene sus propias características y enfoques, todas se adhieren a los valores y principios básicos del desarrollo ágil.

1.3.2 Principios y valores del manifiesto ágil

De acuerdo con una guía desarrollada por (A. Gupta et al., 2022) menciona que los valores del manifiesto ágil corresponden a los siguientes:

a) Priorizar a las personas y sus interacciones sobre los sistemas y las herramientas.

b) Valorar un programa que funcione por encima de una amplia documentación.

c) Promover el trabajo colaborativo con el cliente para la negociación de contratos.

d) Adaptable a cambios en lugar de ceñirse rígidamente a un plan.

Los siguientes principios, desarrollados a partir del trabajo de la Agile Alliance y compartidos por varias metodologías de desarrollo, son destacados por (Pérez, 2012):

1) Entregar a los clientes software de alta calidad de forma rápida y coherente es una de las principales prioridades.

2) Los requisitos cambiantes son bienvenidos, aunque lleguen tarde en la fase de desarrollo. El cambio convierte los procesos ágiles en una ventaja competitiva aún mayor para el cliente.

3) Proporcionar software operativo con regularidad, preferiblemente en plazos de entre unas semanas y unos meses.

4) El personal de la empresa y los desarrolladores del proyecto deben colabora a diario.

5) Construir proyectos en torno a personas motivadas, proporcionándoles los recursos que necesitan y la confianza que merecen para hacer el trabajo.

6) Comunicar información de un lado a otro dentro de un equipo de desarrollo mediante conversaciones cara a cara es el método más eficiente y eficaz.

7) El software operativo es el principal indicador del progreso.

8) Los procesos ágiles fomentan el desarrollo sostenible. Patrocinadores, desarrolladores y usuarios deben mantener un ritmo constante e indefinido.

9) La atención continua a la excelencia técnica pone el relieve de agilidad.

10) El arte de la simplicidad consiste en maximizar la cantidad de trabajo que no se hace.

11) Las arquitecturas, requisitos y diseños más eficaces son producidos por equipos autoorganizados

12) El grupo evalúa periódicamente su rendimiento y realiza ajustes en función de lo que aprende.

1.3.3 Ventajas y desventajas de las metodologías ágiles

De acuerdo con (Ruk et al., 2019) menciona algunas ventajas con respecto a las metodologías ágiles de desarrollo de software, algunas de ellas son:

Ventajas

a) Mayor interacción entre los miembros del equipo y los clientes.

b) Mayor flexibilidad para adaptarse a los cambios en los requisitos del proyecto.

c) Mayor enfoque en la entrega de software totalmente funcional en ciclos cortos.

d) Mayor énfasis en la calidad del producto

Desventajas

En este contexto (Mohammad et al., 2013) menciona que existen algunas desventajas que corresponden a:

a) Las metodologías ágiles dependen principalmente de la interacción entre el cliente o el usuario con el equipo de desarrolladores.

b) Las metodologías ágiles se basan en la comunicación verbal con los clientes lo que conducirá una documentación débil.

c) Difícil coordinación entre equipos en proyectos de gran envergadura.

d) Las metodologías ágiles carecen de planificación a largo plazo.

1.4 Marco de trabajo SCRUM

Scrum, una metodología de desarrollo ágil, actualmente está muy extendida en los proyectos de software actuales. Como afirma (Hamid et al., 2020) la metodología Scrum pone un fuerte énfasis en el desarrollo incremental de software a lo largo de vida del proyecto, lo que la hace altamente flexible para adaptarse a cambios en los requisitos.

1.4.1 Ventajas y desventajas de Scrum

1.4.1.1 Ventajas

Según lo argumentado por (Adnan & Afzal, 2017) se destacan varias ventajas al utilizar el marco de trabajo Scrum, entre las cuales se encuentran la simplicidad, la flexibilidad, la coordinación del equipo, la participación del cliente y un aumento en la productividad.

1.4.1.2 Desventajas

En referencia a menciona (Jaramillo, 2021) que, a pesar de sus numerosos beneficios, la metodología Scrum todavía presenta ciertas limitaciones en áreas como la gestión de riesgos, la presupuestación y la gestión de recursos humanos.

De hecho, esta metodología podría mejorarse al incluir procesos adicionales que permitan una gestión integral del proyecto y eviten la distribución desorganizada de sus esfuerzos por parte de los equipos del proyecto.

1.4.2 Características de SCRUM

Scrum es una metodología de desarrollo ágil definida por equipos autoorganizados y gestionados que intercambian y construyen sobre el conocimiento de los demás. Según una guía escrita por (Palacio, 2022), en lugar de realizar una planificación exhaustiva por adelantado, la estrategia de desarrollo se centra en pasos pequeños y graduales. La calidad del resultado se basa en el conocimiento tácito y la creatividad humana, en lugar de depender únicamente de la calidad de los procesos. Además, la guita adjunta establece en detalle los roles, artefactos y eventos que componen el ciclo estándar de Scrum.

1.4.2.1 Roles

Scrum, marco de trabajo ágil, consta de tres puestos de trabajo principales. Estos tres perfiles son:

a) Scrum Team: Este es un grupo de miembros del equipo que trabajan juntos

en un proyecto para mejorar la productividad. Su objetivo es simplificar las tareas y procedimientos rutinarios, lo que a menudo implica la colaboración estrecha y la autogestión.

b) Propietario del producto / *producto owner*: El propietario del producto representa al cliente o a los interesados y es el responsable de definir y priorizar los elementos del trabajo a realizar. Su objetivo es asegurarse de que el proyecto cumpla con los objetivos y que se entreguen las funcionalidades más valiosas primero.

c) Scrum Máster: El Scrum máster es el facilitador del proceso Scrum. Su función principal es garantizar que el equipo Scrum cumpla con los hitos y plazos del proyecto. Además, ayuda al equipo a encontrar soluciones eficaces para los problemas que surgen durante el proyecto y se encarga de mantener a todos informados durante cada sprint, fomentando la comunicación efectiva y eliminando obstáculos.

1.4.2.2 Artefactos

Los artefactos que corresponden a Scrum son los siguientes:

a) Pila de producto / *producto Backlog*: Un backlog es una lista que lleva el registro de las solicitudes de los clientes y les asigna prioridades. Estas solicitudes suelen denominarse “historias de usuario” y se desglosan en “tareas” más pequeñas.
b) Pila de sprint / *sprint backlog*: Un conjunto iterativo de tareas que deben completarse en un sprint para alcanzar un “incremento” planificado, considerando las necesidades desde el punto de vista de los programadores.
c) Incremento: Es el resultado acumulado de cada sprint.

1.4.2.3 Eventos

Scrum consta de eventos fundamentales para la organización y gestión de proyectos ágiles. A continuación, los eventos que conforman Scrum:

a) Sprints: Los Sprints son períodos de tiempo con una duración generalmente

entra una y cuatro semanas, durante los cuales el equipo se enfoca en completar las tareas necesarias para alcanzar los objetivos del sprint.

b) Reunión de planificación del sprint: En esta reunión, el equipo define y acuerda los objetivos específicos y las tareas que se deben realizar durante el sprint que se está por comenzar.

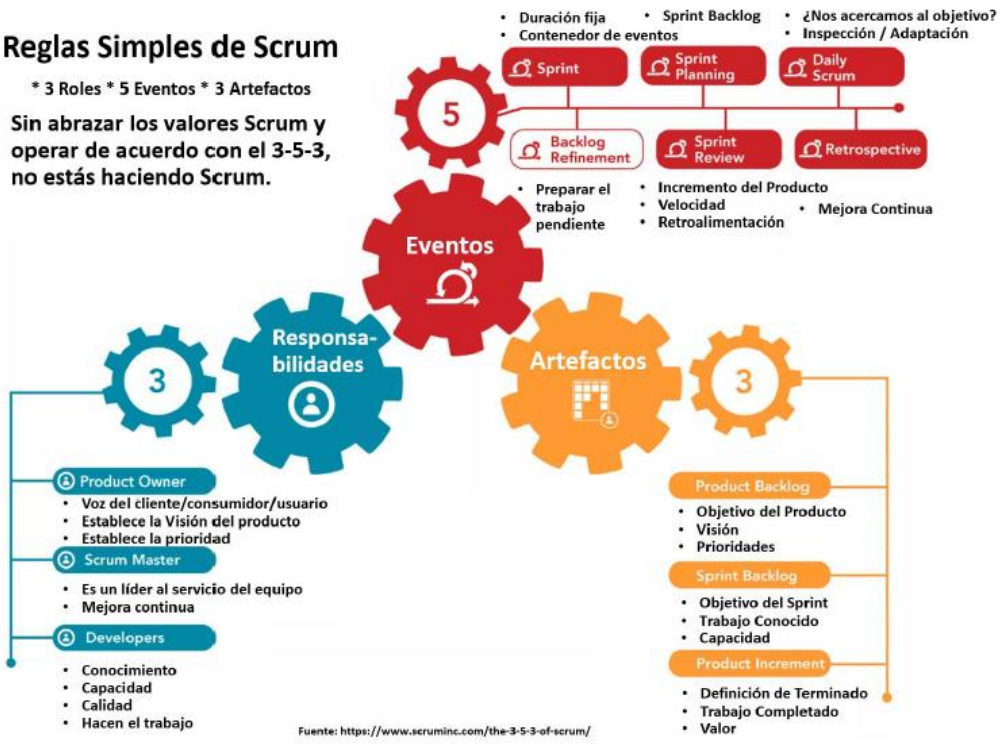
c) Scrum diario: También conocido como reunión Scrum, es una reunión diaria de 15 minutos en la que el equipo revisa los logros del día anterior, planifica las tareas del día actual y se actualiza mutuamente sobre el estado general del sprint.

d) Revisión del sprint: Cada sprint termina con una reunión de retroalimentación que suele durar dos horas. En esta reunión, el equipo discute el progreso alcanzado durante el sprint, evalúa si se han cumplido los objetivos establecidos.

e) Retrospectiva del sprint: La retrospectiva del sprint, también conocida como reunión de revisión posterior al sprint, es una reunión que se realiza al final de cada sprint. En esta reunión, el equipo analiza los problemas que surgieron durante el sprint y propone posibles soluciones para mejorar en futuras iteraciones el proceso.

A continuación, en la Figura 5 se observa las reglas simples que aplica Scrum en el uso de su marco de trabajo.

Figura 5
El 3-5-3 de Scrum



Nota. Adaptado de (Palacio, 2022).

1.4.3 Impacto de Scrum en el desarrollo de software

Como sostiene (Chaudhari & Joshi, 2021), las metodologías ágiles se centran en los requisitos cambiantes de los proyectos de software, mientras que las metodologías tradicionales tienden a ser menos flexibles. Los autores del trabajo investigaron el impacto de un enfoque ágil en el proceso de desarrollo de software en términos de calidad, valor comercial y marco arquitectónico.

Existe un amplio consenso en el que el enfoque más eficaz para resolver este problema de desarrollo de sistemas es emplear la metodología de Desarrollo Ágil de Software con Scrum. Este marco de investigación aplicada se basó en una metodología de investigación-acción, y el sistema se desarrolló utilizando la metodología de Desarrollo Ágil de Software con Scrum. Los resultados de este estudio explican y analizan la evolución del sistema de aseguramiento de la calidad mediante la adopción de la metodología Scrum. Se espera que el sistema de aseguramiento de la calidad ayude significativamente a las

instituciones de educación superior con certificación ISO a elevar sus estándares de acreditación (Legowo et al., 2019).

El ciclo de vida del desarrollo de software está experimentando una transformación radical: los engorrosos procesos tradicionales están dando paso a técnicas ligeras de desarrollo ágil de software (ASD). El ASD crea software excepcional de alta calidad en iteraciones cortas y rápidas para adaptarse a las cambiantes condiciones del mercado. Agile fomenta una respuesta rápida y adaptable al cambio. Con la ayuda de sistemas ágiles, una empresa puede aumentar la satisfacción de sus clientes, reducir la tasa de errores, acortar los tiempos de ciclo y reaccionar mejor a las necesidades cambiantes. Aunque Agile ayuda a la entrega temprana, puede comprometer la calidad del software entregado (Kaur et al., 2021).

Para cualquier proveedor de servicios de software que se aprecie, la satisfacción del cliente es su máxima prioridad. Un equipo de short-sprint se entra a varios retos a la hora de ofrecer resultados de desarrollo de software al cliente. Cuando se trata de conseguir un diseño que cumpla los estándares del sector, el cliente, por su parte, tiene mucho donde elegir. Los cambios en el desarrollo ágil de software se registran en historias de usuario para el sprint, se implementan por etapas y son revisados por los clientes a intervalos regulares. Los Scrum Masters supervisan a sus equipos mientras trabajan en sprints iterativos para completar los objetivos del sprint. Agile ayuda a entregar un producto acabado que cumple los requisitos del cliente, aunque el equipo puede experimentar estrés durante todo el proceso (Manisha et al., 2021).

Agile es una metodología de desarrollo de software que se está convirtiendo rápidamente en la norma de facto del sector. Todas las empresas buscan la agilidad porque la rapidez de entrega y la capacidad de adaptarse a los rápidos cambios de la demanda de los clientes son esenciales. Muchas empresas han tenido éxito utilizando la metodología ágil de gestión de proyectos, que les permite reaccionar con rapidez a las cambiantes condiciones del mercado. La metodología ágil fue desarrollada en la India por una empresa de desarrollo de software, pero puede aplicarse prácticamente en cualquier entorno. La satisfacción del

cliente y la productividad general pueden mejorar como resultado de la adopción de prácticas ágiles. Se puede utilizar ágil o scrum por separado, pero los dos juntos proporcionan una metodología poderosa (N. Gupta et al., 2022).

La gestión de proyectos de software desempeña un papel crucial en la industria del software. Abarca varios métodos y campos de especialización. Tiempo, dinero y alcance son las tres limitaciones de un proyecto de software directamente proporcionales a las necesidades del proyecto. El desarrollo ágil de software es un proceso iterativo que permite cambios frecuentes, entregas rápidas y mitigación de riesgos. La gestión de proyectos de software desempeña un papel decisivo en los proyectos de desarrollo ágil de software. La metodología ágil afecta a la gestión de proyectos de software en 10 dominios de conocimiento. Casi todas las empresas de software utilizan el desarrollo ágil de software (Scrum) y obtienen resultados positivos mientras gestionan proyectos de software, según una encuesta realizada para este estudio (Hayat et al., 2019).

1.5 Marco de trabajo Extreme Programming

De acuerdo con (Joskowicz, 2008), la estrategia de XP está diseñada para brindar al cliente el software que necesitan cuando lo necesiten. XP anima a los desarrolladores a adaptarse a los requisitos cambiantes de los clientes, incluso en las etapas más avanzadas del ciclo de vida del desarrollo.

De acuerdo con (Goto et al., 2014) menciona que la popular metodología de desarrollo ágil de software Extreme Pramming (XP) sigue un ciclo de desarrollo de cuatro etapas que incluye planificación, diseño codificación y pruebas.

1.5.1 Ventajas y desventajas

1.5.1.1 Ventajas

Desde la perspectiva de (Meléndez et al., 2016) , se mencionan algunas ventajas de implementar la metodología ágil Extreme Programming, que incluyen la satisfacción de los

clientes, una planificación horaria efectiva y un método de trabajo flexible que se adapta fácilmente a las condiciones cambiantes.

1.5.1.2 Desventajas

De acuerdo con (Yasvi & Yadav, 2019), se mencionan algunas desventajas de Extreme Programming, como la programación en parejas, que presenta inconvenientes como problemas de sincronización entre programadores, costos adicionales de gestión debido a la necesidad de dos personas para realizar un trabajo en lugar de una, y la posibilidad de que no sea necesario utilizar la programación en parejas si se utilizan estructuras de datos estándar.

Sin embargo, también se destaca que, a pesar de estos inconvenientes, XP sigue siendo más eficiente y proporciona una mejor comprensión del tema y del producto en comparación con los modelos convencionales de desarrollo de software.

1.5.2 Características

1.5.2.1 Roles

A criterio de (Letelier & Penadés, 2015) menciona que los siguientes roles son los siguientes:

a) Programador: Escribe las pruebas unitarias y genera el código del sistema. Requiere coordinación y comunicación con el equipo.

b) Cliente: Documenta las historias de usuario y las pruebas funcionales, prioriza las historias y selecciona aquellas que se implementarán en cada iteración para agregar valor al negocio. Puede representar a muchas personas afectadas por el sistema.

c) Encargado de pruebas: Garantiza que el programa funcione según lo previsto y cumple con los requisitos del cliente. Mantiene comunicación estrecha con los desarrolladores para identificar y corregir errores.

d) Entrenador / *coach:* Supervisa el proceso y guía al equipo en el uso de las prácticas XP, asegurando que se siga la metodología.

e) Consultor: Miembro externo del equipo con conocimientos expertos en un tema vital para el proyecto proporciona orientación en la resolución de problemas específicos.

f) Gestor / *big boss:* Actúa como enlace entre los clientes y los programadores, facilitando un entorno productivo y coordinando las actividades del equipo.

1.5.2.2 Artefactos

Como expresa (Pérez, 2011), los artefactos son esenciales para comprender el proceso de desarrollo de software, la construcción del sistema y el camino a seguir para agregar nuevas funcionalidades. En relación con XP, se destacan los siguientes elementos:

a) Tarjetas de historias de usuario / *story card*: Proporcionan una descripción en primera persona de una funcionalidad en específico desde la perspectiva del usuario.

b) Tarjetas de tareas: Resumen los pasos necesarios para implementar una historia del usuario.

c) Código: El código fuente del aplicativo.

d) Pruebas unitarias: Pruebas automatizadas que garantizan el correcto funcionamiento de una unidad de código.

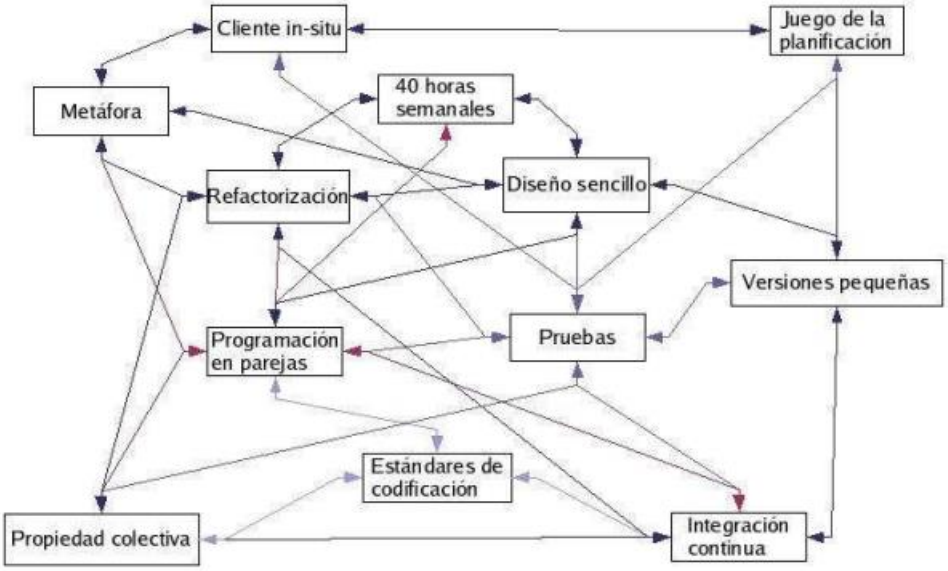
e) Pruebas de integración: Pruebas automatizadas que garantizan el correcto funcionamiento de varias unidades de código del programa.

f) Pruebas de aceptación: Pruebas manuales para asegurar que el software cumple con los requisitos del cliente.

1.5.2.3 Eventos o Prácticas

Para tener una mejor comprensión de cómo funciona y se manejan las prácticas Extreme Programming, se detalla en la Figura 6.

Figura 6
Prácticas de Extreme Programming



Nota. Adaptado de (Letelier et al., 2012)

De acuerdo con (A. Cadavid et al., 2013) se menciona los siguientes eventos que ocurren Extreme Programming:

a) Planificación del juego / *planning game*: En esta reunión, el equipo de desarrollo y el cliente negocian y seleccionan las historias de usuario que se abordarán en el próximo ciclo de desarrollo.

b) Entregas pequeñas / *small releases*: Promueve la distribución períodica de actualizaciones menores que agregas nuevas funciones y corrigen errores. En lugar de esperar a que el proyecto esté completo, se realizan entregas más pequeñas y frecuentes para obtener comentarios tempranos del cliente y verificar el proceso.

c) Diseño Simple / *simple design*: Se enfatiza en la simplicidad y la limpieza del diseño del software, evitando la complejidad innecesaria y fomentando la refactorización continua, lo que mejora la arquitectura del código.

d) Programación en pareja / *pair programming*: La programación en parejas implica que dos programadores colaboren estrechamente en un mismo fragmento de código, lo que fomenta el diálogo, el intercambio de información y las revisiones continuas del código.

e) Pruebas / *testing*: XP promueve tanto las pruebas automatizadas como las manuales con el objetivo de garantizar la calidad del programa, detectando errores tempranos y verificando su correcto funcionamiento.

f) Refactorización / *refactoring*: La refactorización es el proceso de mejora continua del diseño y la estructura del código sin alterar su comportamiento externo. El propósito es mantener una base del código limpia y mantenible, eliminando código redundante y facilitando su legibilidad.

g) Integración continua / *continous integration*: Se logra mediante la incorporación frecuente en el repositorio de código de los cambios realizados por múltiples desarrolladores. Esto permite detectar y resolver conflictos y errores a tiempo.

h) Propiedad común del código / *collective code ownership*: Todos los miembros del equipo son responsables de mantener y mejorar todo el código, sin divisiones estrictas del trabajo cuando se trata de cambios en codificación.

i) Paso sostenible: Se centra en la progresión sostenible, priorizando el mantenimiento de un ritmo de trabajo a largo plazo en lugar de sobrecarga de trabajo.

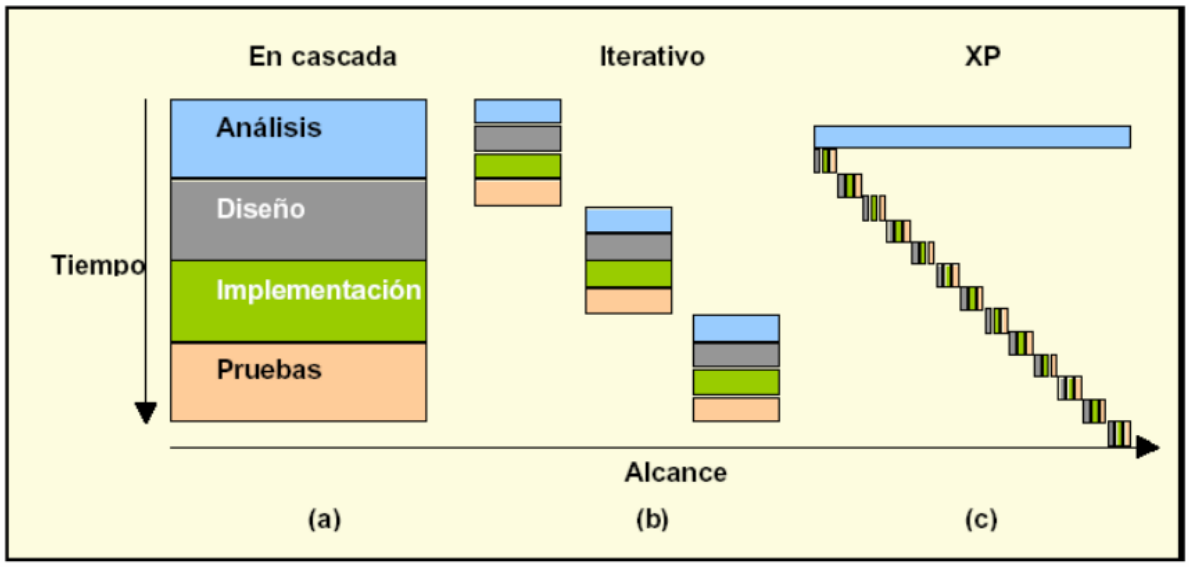
j) Cliente en sitio / *on-site customer:* Se basa en incluir un representante del cliente para resolver dudas con el equipo de desarrollo. Su prioridad es solventar problemas con respecto a la implementación de las historias de usuario.

k) Metáfora / *metaphor*: Es esencial que todos los participantes compartan una compresión común del sistema. Para lograrlo, es necesario crear abstracciones que establezcan un lenguaje compartido entre el cliente y los desarrolladores. La metáfora proporciona un marco general para avanzar de manera efectiva.

l) Estándares de código / *coding standards:* Son directrices que especifican cómo debe escribirse el código de la aplicación. Según la metodología, estas normas deben definirse de manera que el código pueda servir como documento independiente. Dado que tanto la programación en parejas como la propiedad del código son fundamentales, este tema reviste una gran importancia.

En la Figura 7 se observa cómo se desarrolla el proceso de la metodología XP en comparación con ciclos de desarrollo en cascada e iterativos tradicionales.

Figura 7
Comparación de metodología XP



Nota. Adaptado de (Joskowicz, 2008).

1.5.3 Impacto de XP en el desarrollo de software

Extreme Programming (XP) es una metodología ágil ligera diseñada para abordar requisitos cambiantes, con roles como programador, cliente y coach. Desde la posición de (Singh, 2018), realizó una investigación que propuso la integración de la seguridad en la metodología XP para abordar vulnerabilidades. XP, conocida por su flexibilidad para adaptarse a cambios en los requisitos, se mejoró al incluir autenticación de usuarios, análisis de riesgos y validación en etapas clave. Esto generó varios beneficios, como la autenticación de requerimientos y usuarios, la reducción de rechazos de solucione, el análisis de la propagación y la relación de funcionalidades con requisitos. Además, la integración de la seguridad en XP aborda vulnerabilidades en la elicitación de requisitos y la evaluación, lo que mejora la confiablidad y trazabilidad en el desarrollo incremental, y añade una perspectiva de seguridad importante a XP.

XP es una metodología que se basa en prácticas como la programación en parejas y las entregas incrementales. Como sostiene (Rajpal, 2018), en base a una investigación

realizada acerca de la programación en parejas distribuida en un proyecto de desarrollo de software con un equipo repartido en Canadá. En este proyecto, se utilizaron prácticas ágiles de Scrum y XP, y las parejas de programadores programaron en diferentes ciudades con una diferencia horaria de 3 horas. A pesar de algunos desafíos, lograron mantener una alta velocidad de desarrollo y producir un producto de alta calidad. Los desarrolladores destacaron ventajas como la efectividad, el disfrute de trabajo en parejas y la valentía para refactorizar el código. La programación en parejas distribuidas también facilitó la transferencia de conocimiento y la integración del equipo. Los resultados indicaron que la programación en parejas distribuida puede mejorar la productividad, la calidad y la colaboración en equipos distribuidos, lo que tiene implicaciones significativas para la gestión de proyectos ágiles en entornos similares.

1.6 Marco de trabajo KANBAN

Como sostiene (Gong et al., 2018b) alude que el Sistema de producción Toyota (TPS) implantó por primera vez el método Kanban como herramienta para lograr la producción Justo a Tiempo (JIT). Los objetivos del método Kanban incluyen la visualización del flujo de trabajo, la limitación del trabajo en curso (WIP) en cada etapa del flujo de trabajo y la medición del tiempo de ciclo. En resumen, Kanban es un método de gestión de procesos que se enfoca en la visualización y limitación del trabajo en progreso para mejorar la eficiencia y la calidad del proceso.

Citando a (Salvay, 2017) la palabra japonesa Kanban se traduce como “tabla” o “tarjeta gráfica”. En este contexto, el término “sistemas de tarjetas” se refiere a un método de representación de tareas y su progreso utilizando tarjetas físicas. Originalmente desarrollado en los años setenta para las cadenas de montaje de Toyota, esta técnica ha sido recientemente adoptada en el proceso de desarrollo de software. De acuerdo con (Scrum.org et al., 2021) , Kanban es una estrategia para optimizar el flujo de valor a través de un proceso que hace uso de un sistema visual con restricciones sobre el trabajo en curso *(work-in-progress)*.

1.6.1 Ventajas y desventajas

1.6.1.2 Ventajas

De acuerdo con (Gaete et al., 2021) menciona algunas ventajas como:

a) Es fácil de implementar y de integrar con otros enfoques ágiles.

b) Proporciona una herramienta para mejorar la visualización del flujo de trabajo dentro de los proyectos.

c) Ayuda a mantener un adecuado orden en las tareas y a evitar sobrecargas de trabajo.

d) Permite la mejora continua.

1.6.1.3 Desventajas

Algunas de las desventajas que se deben tener en cuenta según (Ballesteros & Silva, 2008) es que al implementar sistema Kanban en una empresa son las siguientes de acuerdo con:

a) Debido a su falta de poder predictivo, el sistema Kanban no se adapta bien a situaciones en las que la demanda es muy imprevisible.

b) Si se establecen plazos de entrega con mucha holgura, puede haber trabajadores con tiempo de ocio.

c) Estabilizar y racionalizar el proceso como preparación para la implementación del sistema Kanban puede requerir tiempo y esfuerzo.

d) Kanban no es una buena opción para las empresas que fabrican una gran variedad de productos, ya que podría ser difícil mantener un flujo constante de materiales.

1.6.2 Características

1.6.2.1 Roles

Kanban, según lo descrito por (Pérez, 2012), no especifica las funciones de los roles. La identidad de una persona esta moldeada por el papel que se le asigna y las responsabilidades asociadas. Por lo tanto, si se busca un cambio en la identidad de alguien, podría ser necesario que asuma un nuevo rol o acepte un nuevo trabajo. Esto plantea la posibilidad de que surja una resistencia significativa al cambio. Kanban aborda esta resistencia emocional reconociendo que la falta de roles dirigidos puede ser una ventaja para el equipo.

Sin embargo, de acuerdo con la (Universidad de Ingeniería y Tecnología, 2022), las prácticas ágiles de la empresa pueden mejorarse con la presencia de un Kanban Coach. Como resultado, aumentarán la eficiencia y la productividad del equipo, así como el nivel de motivación y compromiso de todos con su trabajo.

Adicionalmente, (Anderson & Carmichael, 2016) coinciden en que Kanban no hay roles obligatorios, no obstante, sugieren que puede ser más productivo usar roles como “gorros” que la gente se pone para realizar estas tareas:

- El Gestor de Peticiones de Servicio (*Service Request Manager*), es el encargado de comprender las necesidades y expectativas de los clientes y de facilitar, seleccionar y ordenar los elementos de trabajo en la Reunión de revisión de la cartera de trabajo. El Gestor de Producto, el Dueño de Producto y el Gestor de Servicio son otros nombres alternativos para el papel.
- El Gestor de Prestación de Servicio *(Service Delivery Manager)*, es responsable del flujo de trabajo entregando los elementos seleccionados a los clientes y facilitando la reunión de Kanban y la planificación de entrega. El Gestor del flujo de trabajo, el Gestor de la entrega y el Maestro del flujo son otros nombres para este puesto.

1.6.2.2 Artefactos

A criterio de (Santos et al., 2023), se pueden distinguir artefactos físicos y digitales en Kanban. Los artefactos físicos toman la forma de tarjetas que indican a una línea de producción que fabrique un número específico de un artículo determinado. Estas tarjetas se colocan en un tablero Kanban, una herramienta visual para supervisar el flujo de trabajo y asignar los recursos de manera eficiente. Cada etapa del proceso de producción se representa en el tablero Kanban mediante una columna, a la siguiente medida que se completan.

Por otro lado, los elementos electrónicos, a menudo denominados E-Kanban, están reemplazando cada vez más las tarjetas físicas con notificaciones electrónicas. Estas notificaciones pueden generarse mediante un ordenador u otro dispositivo inteligente, como un escáner de códigos de barras. Al eliminar la necesidad de tarjetas físicas y reducir el riesgo de error humano, el E-Kanban se presenta como un método más eficaz y preciso para gestionar el flujo de trabajo.

1.6.2.3 Eventos

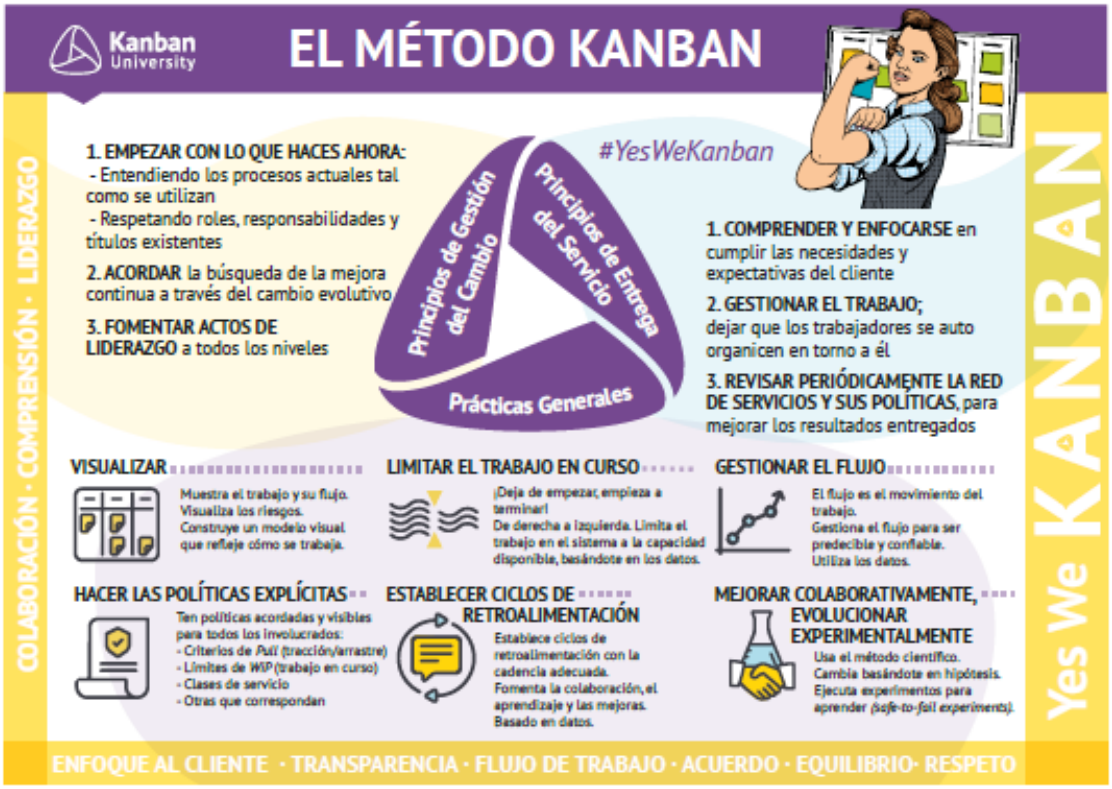
De acuerdo con (Scrum.org et al., 2021) existen las siguientes prácticas en Kanban:

1. Visualización del Workflow: Es la creación de un tablero Kanban que muestra el proceso de trabajo del equipo, desde la asignación inicial de tareas hasta su finalización. Esta visualización permite a los miembros del equipo tener una visión clara del estado actual del trabajo y detectar posibles cuellos de botellas.
2. Limitar el Work in Progress (WIP): Implica establecer un número máximo de elementos de trabajo que pueden estar en progreso al mismo tiempo. Esta práctica ayuda a evitar la sobrecarga del equipo y a mantener un flujo de trabajo constante.
3. Gestión activa de los ítems de trabajo en progreso: Se centra en finalizar las tareas en curso antes de iniciar nuevas. Esta gestión activa garantiza un flujo de trabajo continuo y evita retrasos.

4. Inspeccionar y adaptar la definición de Workflow del equipo: Comprende las políticas y prácticas que rigen el flujo de trabajo del equipo. La inspección y la adaptación son fundamentales para la mejora continua del proceso y el mantenimiento de un flujo de trabajo contante.

A continuación, en la Figura 8 se resume como se desarrolla la metodología Kanban con sus respectivos valores, principios, prácticas y artefactos.

Figura 8
El método Kanban



Nota. Adaptado de (Anderson & Carmichael, 2016).

1.6.3 Impacto de Kanban en el desarrollo de software

En este estudio de desarrollo de un sistema de gestión universitaria, se eligió la metodología ágil Kanban debido a su flexibilidad en comparación con Scrum. La investigación se centró en Kanban debido a sus beneficios en la mejora de procesos, personas y organizaciones. A pesar de los desafíos que implica la incorporación de la experiencia del usuario en un proceso ágil, Kanban se implementó con éxito para gestionar el ciclo de vida

del software. El proceso de investigación comenzó con la identificación y documentación de los requisitos de usuario, utilizando una Especificación de Requisitos de Software y diagramas de casos de uso y entidad-relación. Los resultados evidenciaron la efectividad de Kanban en la producción de software utilizable, ya que ayudó en la gestión de flujo de trabajo, la priorización de tareas y el enfoque en entregables claves de acuerdo con los requisitos del usuario (Rahmat & Hanifiah, 2020).

Hoy en día la metodología Scrum es una de las más empleadas para el desarrollo de software, sin embargo, la idea de unificar ideas de diferentes metodologías ágiles ya es un hecho que va tomando fuerza. Una investigación realizada por (Matthies, 2018), menciona que introdujo Kanban en un curso universitario de ingeniería de software que ya utilizaba Scrum. Tras trabajar en 4 sprints de Scrum, se implementó Kanban en la fase final del proyecto centrada en correcciones y mejoras. Los resultados mostraron que los estudiantes tenían una actitud positiva hacia Kanban debido a su eficiencia y autonomía. Además, se observó que adoptaron su proceso con historias de usuario más cortas y un enfoque en la resolución de errores.

"La elección de la metodología Kanban implica la presencia de un Kanban Coach (KC), cuya función es promover la práctica ágil. Una investigación realizada por (Shamshurin & Saltz, 2019), evaluó el impacto de un KC en equipos que implementaron Kanban. El experimento se llevó a cabo en un curso de ciencia de datos, donde se asignaron equipos con y sin KC. Los resultados revelaron mejoras significativas en la calidad del trabajo, el progreso y el rendimiento final en los equipos que contaron con la presencia de un KC. Los estudiantes valoraron al KC por su apoyo en el seguimiento y la eficiencia de Kanban. La explicación de cómo el KC mejoró la coordinación y el conocimiento compartido se basa en teorías de coordinación y modelos mentales compartidos. En resumen, la incorporación de un KC resultó eficaz para potenciar el rendimiento de los equipos que utilizan Kanban, lo cual tiene implicaciones tanto teóricas como prácticas.

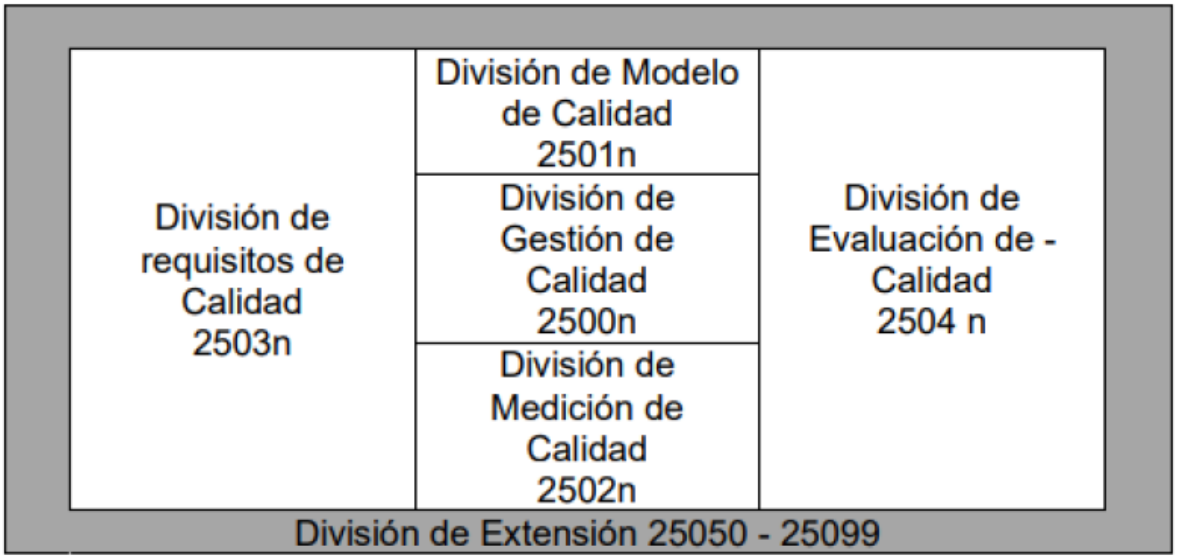
1.7 Modelo de calidad interna de la ISO/IEC 25000

La norma ISO/IEC 25000, también conocida como SQuaRE (Requisitos y evaluación de la calidad de productos del software) se componen de las normas ISO/IEC 9126 e ISO/IEC 14598, y se utiliza como referencia para desarrollar modelos, métricas, procesos y herramientas de evaluación de calidad del software (Roa et al., 2015).

1.7.1 División modelo SQuaRE

Como puede verse en la Figura 9, la (ISO/IEC 25000, 2015)se divide en cinco categorías:

Figura 9
ISO/IEC 25000



Nota. Adaptado de (ISO/IEC 25000, 2015)

- **ISO/IEC 2500n – División de Gestión de Calidad.** Los modelos, términos y definiciones comunes a otras normas SQuaRE se definen en las normas que componen esta división.
- **ISO/IEC 2501n – División de Modelo de Calidad.** Las normas que componen esta sección incluyen modelos de calidad exhaustivos para sistemas informáticos y productos de software, así como requisitos de usabilidad e integridad de datos.

- **ISO/IEC 2502n – División de Medición de Calidad.** Las normas que componen esta sección incluyen un modelo de referencia de medición de la calidad del software, definiciones matemáticas de métricas de calidad y una guía de aplicación práctica.
- **ISO/IEC 2503n – División de Requisitos de Calidad.** Los requisitos de calidad pueden definirse más claramente con ayuda de las normas que componen esta sección, basadas en modelos de calidad e indicadores de calidad.
- **ISO/IEC 2504n – División de Evaluación de Calidad.** Los evaluadores, compradores y desarrolladores pueden beneficiarse de las normas de esta sección, que establecen lo que se espera de ellos a la hora de evaluar un producto de software.
- **ISO/IEC 25050 – 25099 División de Extensión.** Las directrices que componen esta sección cubren aspectos como la calidad mínima del software empaquetado antes de que pueda venderse y los formatos estándar utilizados por el mundo empresarial para los informes de usabilidad.

1.7.2 Calidad del software

A juicio de (Tulia & Jácome, 2018) , “el conjunto de atributos deseables que posee un producto software, que son medibles (cuantitativa o cualitativamente), permitiendo realizar comparaciones para saber si satisface o no las expectativas del cliente” es como definen la calidad del software diversos autores. Si la calidad del software se entiende como el cumplimiento de requisitos mensurables con el objetivo final de satisfacer al cliente, entonces un software superior es aquel que mejora el producto de forma mensurable que aumenta su valor final.

1.7.3 Normas ISO/IEC para la calidad del uso

De acuerdo con (Computer Society & Engineering Standards Committee, 2017) menciona que el sistema mundial de normalización especializada está formado por la Organización Internacional (ISO) la Comisión Internacional Electrotécnica (IEC). Con el fin de

colaborar en el desarrollo de normas internacionales, las organizaciones miembros de la ISO e IEC forman comités técnicos para abordar áreas específicas de trabajo técnico.

1.7.4 Norma ISO/IEC 25022

En la opinión de (Nakai et al., 2016) la norma ISO/IEC 25022 se utiliza como base para el marco de evaluación de calidad del software. Esta norma define un conjunto de características y subcaracterísticas de calidad que se pueden utilizar para evaluar el software en términos de su funcionalidad, confiabilidad, usabilidad, eficacia, eficiencia, mantenibilidad y portabilidad.

Desde el punto de vista de (Pradanita & Rochimah, 2020) menciona que la norma ISO/IEC 25022 define cuatro criterios para evaluar la calidad de uso: efectividad, eficiencia, satisfacción y libertad de riesgo.

De igual manera (J. A. Quiña-Mera et al., 2019) menciona que, para reducir los cambios no deseados durante el ciclo de vida del desarrollo de software, la gestión adecuada de requisitos, respaldada por estándares como ISO/IEC 25022, es esencial; esto mejora la calidad del producto final.

1.7.4.1 Modelo de calidad en uso

En la Tabla 6 se puede observar en completitud el modelo de calidad con sus respectivas subcaracterísticas y métricas propuestas por la ISO/IEC 25022.

Tabla 6
Modelo de calidad en uso

MODELO DE CALIDAD EN USO		
Características	**Subcaracterística**	**Métricas**
Eficacia	Tareas completas	X = A / B A = Número de tareas únicas completado B = Número total de tareas únicas intentadas
	Objetivos logrados	{X = 1 – Σ Ai | X≥0}

		Ai = valor proporcional de cada objetivo que falta u objetivo incorrecto en la salida de la tarea (valor máximo = 1)
	Los errores en una tarea	X = A A = número de errores cometidos por el usuario durante una tarea
	Tareas con errores	X= A / B A = Número de tareas con errores B = Número total de tareas
	Intensidad de errores de tareas	X = A / B A = Número de usuarios de cometer un error B = Número total de usuarios realizar la tarea
Eficiencia	Tiempo de tareas	X = T T = Tiempo de Tarea
	Eficiencia del tiempo	X = A / T A = Número de objetivos alcanzados T = Tiempo
	La rentabilidad	X = A / B A = Coste total de la realización de la tarea B = Número de objetivos alcanzados
	Productividad relación del tiempo	X = Ta / Tb Ta = tiempo productivo = tiempo tomado para completar la tarea - el tiempo pasado obtener ayuda o asistencia - tiempo necesario recuperarse de

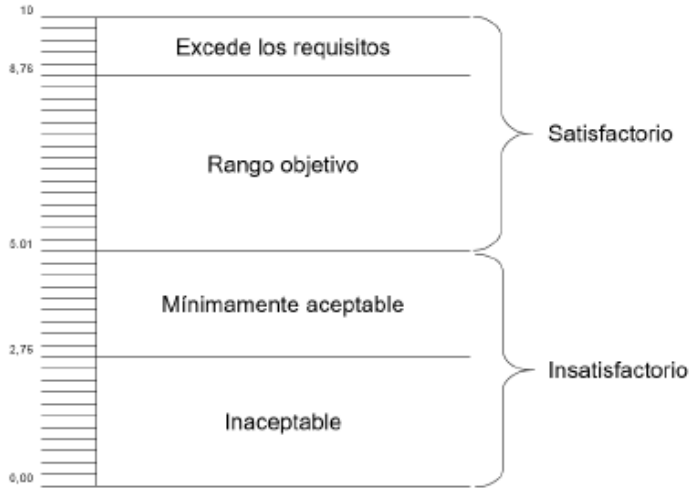
		errores - tiempo tomado buscar inútilmente Tb = tiempo de tareas
	Comportamiento innecesario	X = A / B A = Número de acciones que en realidad no fueron necesarias para lograr la tarea B = Número de acciones realizadas por el usuario
	Consecuencias de la fatiga	X = 1 - A / B A = El rendimiento actual B = rendimiento inicial.
Satisfacción	Utilidad	X = S/E S = Número de usuarios satisfechos E = Número de usuarios encuestados
	Confianza	X = A / T, C=1-X X = % reclamos, % Confianza A = Número de quejas presentadas T = Total de encuestados
	Comodidad	X = (A+B+C+D+F) /E A = Muy de acuerdo; B=Algo de acuerdo; C=Ni de acuerdo ni en desacuerdo; D= Algo en desacuerdo; F=Muy en desacuerdo. E = Número de usuarios encuestados

Fuente: (ISO/IEC 25022, 2016)

1.7.4.2 Escala de evaluación

Para evaluar de manera cualitativa los resultados de la medición del modelo de calidad, se utilizó la escala de evaluación sugerida por la ISO/IEC 25000, la cual se observa en Figura 10. La escala se divide en dos categorías principales y cuatro categorías secundarias.

Figura 10
Escala de evaluación



Fuente: Basada en la (ISO/IEC 14598-1,1999)

1.8 Trabajos relacionados

A continuación, se muestran estudios y trabajos relacionados a la unidad de análisis de estudio:

a) Comparación de la facilidad de aprendizaje entre GraphQL y REST mediante un modelo de calidad interna basada en las ISO/IEC 25000.

   En el trabajo de investigación de (Chulde, 2022) se definió un modelo de calidad tomando la característica de usabilidad y la sub-característica de facilidad de aprendizaje de la norma ISO/IEC 25010. Como métrica para medir la facilidad de aprendizaje se utilizó la completitud de la guía de usuario definida en la ISO/IEC 25023. A partir de eso se diseñó un experimento controlado en el que se enseñó GraphQL y REST a un grupo de estudiantes, se les asignó tareas y se midió su

desempeño. Los resultados indicaron que con GraphQL los estudiantes completaron correctamente el 53% de las tareas, mientras que con REST completaron el 46%. Según la escala de medición de la ISO/IEC 14589, GraphQL obtuvo un resultado “satisfactorio”, mientras que REST obtuvo un resultado “insatisfactorio”. Esto permitió concluir, que, según el modelo de calidad definido, GraphQL tiene un mayor de facilidad de aprendizaje en comparación a REST.

b) Creación de una aplicación backend del módulo de transferencias (cuentas y montos), con microservicios GraphQL y REST en contenedores Docker, para fomentar el levantamiento de una arquitectura DevOps.

   En base a la investigación realizada por (De La Cruz, 2022), el proyecto se centró en desarrollar una aplicación de microservicios con NestJS y comparar su eficiencia con Docker y entorno local. Dividido en tres fases bajo la metodología Scrum, se diseñó la arquitectura, se implementaron los microservicios y se realizaron pruebas en ambos entornos. Los resultados demostraron que Docker superó al entorno local, especialmente en el despliegue de GraphQL a comparación de REST, ofreciendo un mejor rendimiento en términos de tiempo de respuesta promedio para los casos de uso.

c) Desarrollo de un software educativo web gamificado para la enseñanza de informática básica.

   De acuerdo con el trabajo realizado por (Chasqui, 2020) menciona que desarrolló una aplicación web gamificada para enseñar informática básica en la Escuela Politécnica Nacional. La metodología ágil Extreme Programming (XP) se utilizó para el desarrollo y la metodología iPlus para la obtención de requerimientos. XP permitió un diseño simplificado, una reevaluación del tiempo del proyecto, entregas frecuentes, adaptación a cambios y un producto de alta calidad en iteraciones cortas. En conclusión, XP, permitió un desarrollo ágil y eficiente que se adaptó a los requisitos y proporcionó un producto funcional de alta calidad progresivamente.

d) Desarrollo de un sistema web turístico para la parroquia de Nono.

En base a un estudio realizado por (Donoso, 2023), desarrolló un sistema web turístico en Nono, Ecuador, se llevó a cabo con la metodología Kanban. Kanban permitió una organización eficiente al visualizar el flujo de trabajo a través de tarjetas en un tablero. Los resultados derivaron en una definición clara de tareas, diseño eficiente, implementación organizada y detección temprana de errores. Se lograron principalmente entregas incrementales y adaptación ágil. En conclusión, Kanban tuvo un impacto positivo al facilitar entregar continuas, mejorar la productividad y la calidad del sistema web desarrollado.

e) Desarrollo de un prototipo de sistema web informativo para cálculo de matrícula y aranceles.

   Teniendo en cuenta a un trabajo de pregrado desarrollado por (Cargua, 2023), indica que Kanban es una metodología ágil que se basa en la gestión visual de trabajo, que fue aplicada en el desarrollo de un prototipo de sistema web para cálculo de matrícula y aranceles. Esto resultó en la definición y priorización efectiva de los requerimientos, la planificación y seguimiento de tareas, la evitación de sobrecarga de trabajo, la identificación de cuellos de botella y la entrega incremental de funcionalidades. Los resultados indicaron Kanban tuvo un impacto altamente positivo al mejorar la gestión, la eficiencia del equipo y la entrega de un prototipo altamente positivo al mejorar la gestión, la eficiencia del equipo y la entrega de un prototipo funcional que cumplió con los requerimientos del usuario en el proyecto de desarrollo de software.

f) Diseño e implementación de una arquitectura core para automatización y mejora de servicios en los laboratorios de la facultad de ingeniería en sistemas utilizando la metodología Scrum.

   De acuerdo con un proyecto desarrollado por (Narváez & Vaca, 2022), indica que el propósito de esta investigación fue diseñar e implementar una arquitectura core para la automatización y mejora de servicios en los laboratorios de la Facultad de Ingeniería en Sistemas utilizando Scrum. La metodología Scrum permitió a los

autores del proyecto trabajar de manera ágil y eficiente, lo que les permitió obtener avances constantes y reales del producto que deseaban. La implementación de esta solución tuvo un impacto positivo en cuanto a la reducción de costos y mejorar la calidad de los servicios, generando confianza en un entorno complejo. En conclusión, Scrum resultó en un mejor retorno de inversión y clientes más satisfechos debido a los avances reales y constantes del producto deseado.

g) Aplicación de un framework para la evaluación de la calidad en uso del SII-académico de la escuela politécnica nacional.

   Como expresa (Maldonado, 2020), en su proyecto de tesis de maestría, evalúa la calidad en uso de los módulos del Sistema Integrado de Información Académica (SII-Académico) de la EPN, utilizando la norma ISO 25022. Esta norma proporciona métricas estandarizadas para medir al efectividad, eficiencia y satisfacción del usuario. El uso de la norma ISO 25022 permite una evaluación objetiva y la identificación de deficiencias, lo que facilita la formulación de recomendaciones para mejorar la calidad desde la perspectiva del usuario. Los resultados de la evaluación de la calidad en el uso del SII-Académico revelaron que la efectividad obtuvo un valor de 2.87/3, la eficiencia un 2.32/3, satisfacción un valor de 3.27/4 y la calidad de uso 8.46/10. Las derivaciones revelan problemas en eficiencia y satisfacción, como tiempos de tareas prolongados y quejas sobre la complejidad del sistema.

h) Quality in Use of Digital Wallet based on ISO/IEC 25022.

   De acuerdo con un estudio realizado por (Pradanita & Rochimah, 2020) propuso una evaluación de la calidad de uso de los productos de billetera digital basada en la norma ISO/IEC 25022. La calidad de uso se relaciona con la interacción del usuario con el software cuando se usa el producto. El cuestionario evalúa la efectividad midiendo la precisión y exhaustividad con la que los usuarios logran objetivos específicos. Evalúa la eficiencia midiendo los recursos gastados con relación a la precisión y la exhaustividad de lograr metas. Mide la satisfacción como

el grado en que se satisfacen las necesidades del usuario. Y mide la libertad de riesgo como el grado en que la billetera digital mitiga riesgos potenciales. Los resultados indicaron que la efectividad (85%), eficiencia (84%) y satisfacción (83%) obtuvieron resultados en la categoría “muy bueno”, mientras que la libertad de riesgo obtuvo un resultado en la categoría “bueno”.

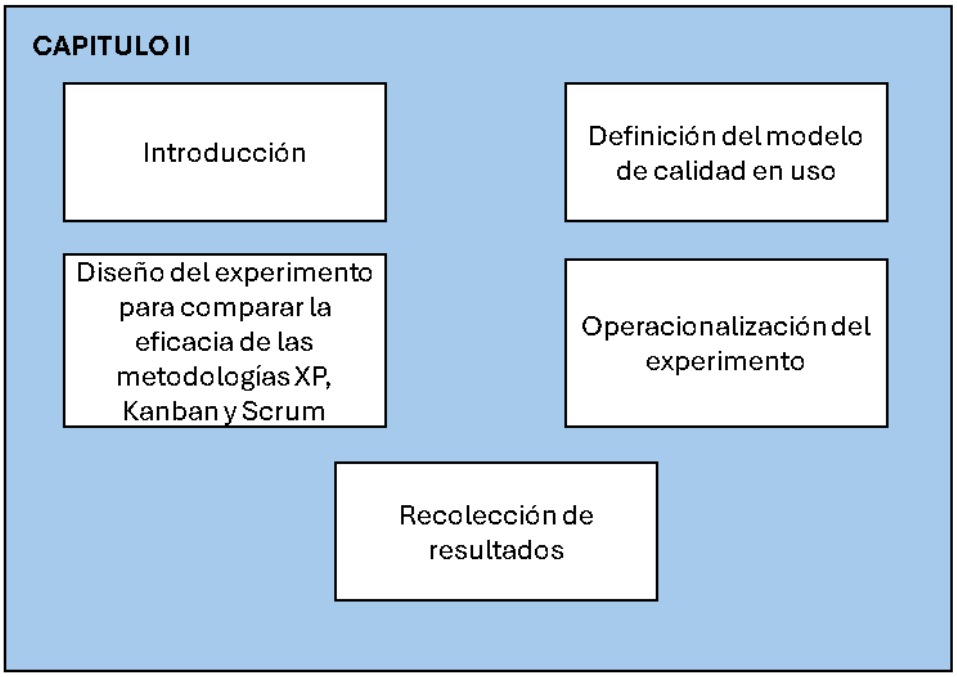
CAPÍTULO 2

Desarrollo del proyecto

2.1 Introducción

En este capítulo, se procede a definir y desarrollar un experimento controlado con el propósito de comparar la eficacia entre las metodologías ágiles Scrum, XP y Kanban. Este experimento se basa en un modelo de calidad en uso que hace uso de las normas ISO/IEC 25000 específicamente la norma ISO/IEC 25022.

Figura 11
Estructura Capítulo II



Elaboración: Propia.

2.2 Definición del Modelo de Calidad en Uso

En conjunto con los investigadores del experimento, el Ing. Mauricio Rea, PhD. Antonio Quiña y Juan Iza, se definió el modelo de calidad en uso en el marco del desarrollo del proyecto de investigación interno de la carrera de Software en la asignatura de Construcción de Software. El MSc. Mauricio Rea desempeñó el papel de definir la calidad y el PhD. Antonio Quiña definió el diseño del experimento.

Se utilizó el modelo de medición de calidad utilizado de acuerdo con la ISO/IEC 25022 y se eligió la característica de eficacia de acuerdo con las necesidades encontradas. La medida de calidad interna utilizada, según la (ISO/IEC 25023, 2020), mide el conjunto de atributos estáticos de un producto de software y si satisface las necesidades establecidas e implícitas del producto o sistema. La estructura manejada se identifica en la Tabla 7.

Tabla 7
Definición Modelo de Calidad en Uso

Definición Modelo de Calidad en Uso			
Característica	**Sub característica**	**Porcentaje Sub Característica**	**Porcentaje Característica**
Eficacia	Tareas completas	28%	60%
	Objetivos logrados	16%	
	Los errores en una tarea	16%	
Satisfacción	Utilidad	15%	40%
	Confianza	15%	
	Comodidad	10%	

Elaboración: Propia.

La medición identificada para las sub características de eficacia y satisfacción de la ISO/IEC 25022 se muestra en la Tabla 8.

Tabla 8
Medidas de eficacia y satisfacción

Característica	**Sub característica**	**Nombre del método**	**Función de medición**
Eficacia	Tareas completas	Medir el rendimiento del usuario	X = A / B

			A = Número de tareas únicas completado B = Número total de tareas únicas intentadas X = A
	Los errores en una tarea	El número de errores del usuario durante una tarea	A = número de errores cometidos por el usuario durante una tarea
	Tareas con errores	Proporción de tareas en las que errores fueron hechas por el usuario	X = A / B A = Número de tareas con errores B = Número total de tareas
	Utilidad	Determinar la cantidad de usuarios satisfechos con el uso del sistema de gestión documental	X = S/E S = Número de usuarios satisfechos E = Número de usuarios encuestados
Satisfacción	Confianza	Determinar la confianza de los usuarios, mediante la medición de las quejas presentadas por alguna falla del sistema de gestión documental	X = A / T, C=1-X X = % reclamos, % Confianza A = Número de quejas presentadas T = Total de encuestados

			X = (A+B+C+D+F) /E A = Muy de acuerdo; B=Algo de acuerdo; C=Ni de acuerdo ni en desacuerdo; D: Algo en desacuerdo; F=Muy en desacuerdo. E = Número de usuarios encuestados
	Comodidad	Determinar la facilidad y poco esfuerzo en el uso del sistema de gestión documental	

Elaboración: Propia.

En base a la ISO/IEC 25022 se tomó en cuenta las notas proporcionadas por la norma las cuales describen lo siguiente: Esta medida se puede medir para un usuario o un grupo de usuarios; Si las tareas se pueden completar en parte, la medida de los objetivos alcanzados es más apropiado; NOTA 3 Si las tareas son de diferente complejidad, las tareas ponderados podría ser utilizado en la fórmula: $X = \Sigma (i = 1..n) Wi \times Ai / B$, donde i es el número de la tarea y Wi representa la dificultad de que la tarea donde suma total de Wi = 1,0.; Este podría ser aplicada a las tareas identificadas en los requisitos o las tareas.

2.3 Diseño del experimento controlado para comparar la eficacia entre las metodologías ágiles SCRUM, XP y Kanban

El diseño del experimento se basó en la Guía de Experimentación en Ingeniería de Software de Wohlin (2012) y se llevó a cabo en colaboración con el MSc. Antonio Quiña y el Ing. Mauricio Rea, docentes de la Universidad Técnica del Norte. El experimento se realizó utilizando estrategias empíricas de ingeniería de software.

2.3.1 Objetivo

El objetivo principal del experimento controlado es comparar la calidad interna entre las metodologías ágiles Scrum, XP y Kanban con respecto a las tareas completadas, la cual

es una sub característica de la Eficacia. El experimento se consolidó mediante los siguientes pasos:

1. Capacitar a un grupo de estudiantes de Ingeniería de Software sobre metodologías ágiles: Scrum, Kanban y Extreme Programming.
2. Brindar recursos didácticos que faciliten la ejecución de cada metodología.
3. Realizar un taller con los estudiantes
4. Aplicar un cuestionario SUS, realizar la evaluación y analizar los resultados obtenidos con el experimento

2.3.2 Variables

Como se observa en la Tabla 9, la variable independiente fue identificada con las metodologías ágiles Scrum, Kanban y XP. De igual manera la variable dependiente se identificó a la calidad en uso (sub característica eficacia y satisfacción)

Tabla 9
Variable dependiente e independiente

Variable Independiente	Construcción de software con marco de trabajo Scrum
	Construcción de software con marco de trabajo Kanban
	Construcción de software con marco de trabajo XP
Variable Dependiente	Mejorar la calidad en uso (eficacia y satisfacción) del producto del software

A continuación, se detalla el uso de las métricas que se utilizará para evaluar el modelo de calidad en uso con su respectiva descripción, función de medición y método en función de la guía de usuario establecida en la (ISO/IEC 25022, 2016).

La métrica de tareas completas con respecto a la completitud de la guía de usuario se define como La proporción de tareas que se han completado sin ayuda y su función de medición es:

$$X = A / B$$

Dónde A es el número de tareas únicas completado y B es Número total de tareas únicas intentadas.

Por otro lado, también se encuentra la métrica de Objetivos logrados en una tarea, en la cual la norma establece como la proporción de los objetivos de la tarea que se logra correctamente sin ayuda, su función de medición es:

$$\{X = 1 – \Sigma Ai \mid X \geq 0\}$$

Dónde A i = valor proporcional de cada objetivo que falta u objetivo incorrecto en la salida de la tarea (valor máximo = 1)

Finalmente, la métrica de Los errores en una tarea dónde la norma indica que corresponde a el número de errores del usuario durante una tarea, su función de medición es:

$$X = A$$

Dónde A = número de errores cometidos por el usuario durante una tarea.

En cuanto a la característica de satisfacción, las métricas seleccionadas corresponden a la utilidad, dónde la norma indica que corresponde a determinar la cantidad de usuarios satisfechos con el uso del sistema de gestión documental, su función medición es

$$X = S/E$$

Dónde S = Número de usuarios satisfechos y E = Número de usuarios encuestados.

Otra métrica es la confianza y la norma indica que concierne a determinar la confianza de los usuarios, mediante la medición de las quejas presentadas por alguna falla del sistema de gestión documental, su función de medición es:

$$X = A / T, C=1-X$$

Dónde X = % reclamos, % Confianza, A = Número de quejas presentadas y T = Total de encuestados.

Finalmente, la comodidad que determina la facilidad y poco esfuerzo en el uso del sistema de gestión documental, su función de medición es:

$$X = (A+B+C+D+F) /E$$

Dónde A = Muy de acuerdo; B=Algo de acuerdo; C=Ni de acuerdo ni en desacuerdo; D: Algo en desacuerdo; F=Muy en desacuerdo y E = Número de usuarios encuestados.

2.3.3 Población

Los sujetos del experimento fueron 20 estudiantes de la carrera de Ingeniería en Software de la Universidad Técnica del Norte (Ibarra - Ecuador) los cuales eran estudiantes de la materia de Construcción de Software del período abril – julio 2024.

2.3.4 Diseño

El experimento se llevó a cabo utilizando un diseño factorial cruzado único para cada tratamiento, lo que permitió comparar los efectos de cada tratamiento entre los grupos (Vegas et al., 2016).

Las tres formas en que se asignan sujetos a diferentes tipos de tratamientos son explicadas por Vegas y sus colaboradores (2016), en el experimento controlado se utilizó el diseño factorial cruzado, en el cual las diferentes condiciones o variables que se están estudiando en un experimento se denominan tratamientos. Un diseño factorial cruzado con

tres tratamientos podría llamarse A, B y C. Cada tratamiento es una condición experimental única que se aplica a los sujetos en un orden determinado para evaluar sus efectos individuales y sus posibles interacciones. Además, cada tratamiento consta de un nivel, la cual tiene objetos diferentes en cada tratamiento, se realizó con 3 objetos, y estos 3 objetos hacen referencia a las metodologías que van a aplicar en cada proyecto de aula (Scrum, XP y Kanban), tal como se observa en la Tabla 10.

Los sujetos de prueba realizaron grupos de 4 estudiantes por afinidad para desarrollar el proyecto de aula asignado, sin embargo, el uso de cada metodología fue de manera aleatoria, el experimento fue realizado a partir del 02 de mayo del 2024 hasta el 23 de mayo del 2024, dando un tiempo de duración de 11 días durante 6 clases de la asignatura.

Tabla 10
Diseño factorial cruzado

Secuencia	Período	Tratamiento	Objeto
Grupo 1	11 días	Extreme Programming	Taller, Uso Metodología
Grupo 2	11 días	Kanban	Taller, Uso Metodología
Grupo 3	11 días	Scrum	Taller, Uso Metodología
Grupo 4	11 días	Kanban	Taller, Uso Metodología
Grupo 5	11 días	Extreme Programming	Taller, Uso Metodología

2.3.5 Tareas

Fase de preparación

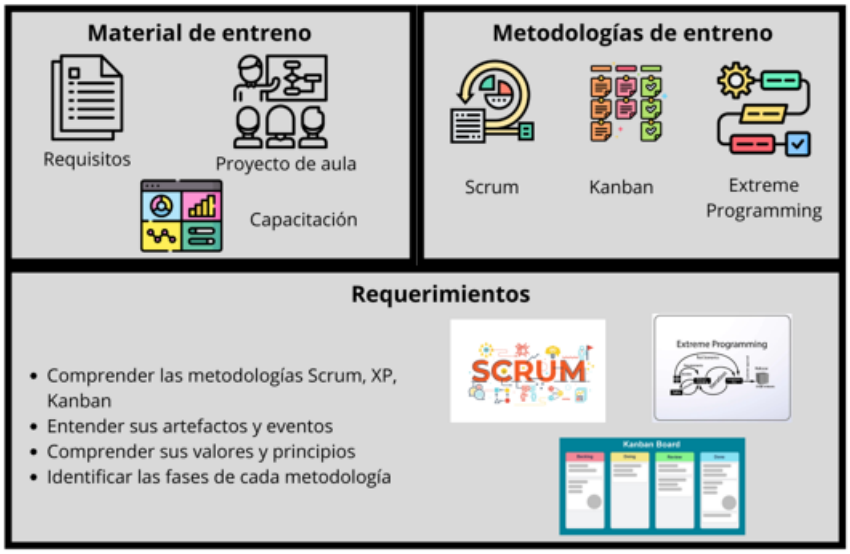
La fase de preparación contó con una breve introducción de las metodologías Scrum, XP y Kanban, además de una guía de cómo aplicar los recursos didácticos para facilitar la aplicación de cada metodología.

Primera Fase

En la primera fase se realizó una capacitación que constaba con la explicación conceptual y los fundamentos de cada metodología, de igual manera, utilizando herramientas de ofimática se explicó el proceso de cada marco de trabajo en ejercicios prácticos y el uso de artefactos que se van a utilizar durante el ejercicio. Para complementar los sujetos de prueba tuvieron acceso a recursos de pregrado en el que se usaban dichas metodologías de una manera más empresarial y procedan a realizar el taller sin ninguna dificultad durante las otras fases, todos los recursos explicados fueron entregados a los estudiantes.

En la Figura 12 se identifica la estructura de cómo se trabajó la primera fase con los sujetos experimentales:

Figura 12
Estructura primera fase

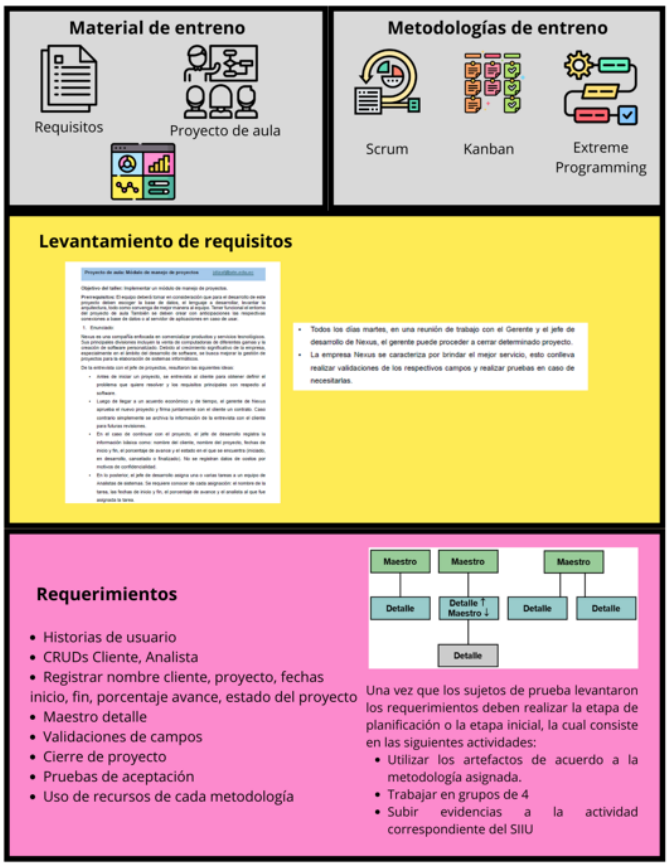


Fuente: Propia

Segunda Fase

La segunda fase consistió en que los sujetos de prueba se organizaron en grupos de cuatro personas y se les procedió a explicar el taller a realizar durante un lapso de tres semanas, el proyecto de aula consistía en realizar un ejercicio con la temática de gestión de proyectos, levantamiento de requerimientos, desarrollo de requisitos funcionales, aplicación de marco de trabajo de acuerdo al grupo, tal como se explicó en el diseño del experimento, todo esto basado en el documento de requisitos que se compartió durante el transcurso de la segunda fase. En la Figura 13 se observa la estructura más detallada de la segunda fase.

Figura 13
Estructura segunda fase



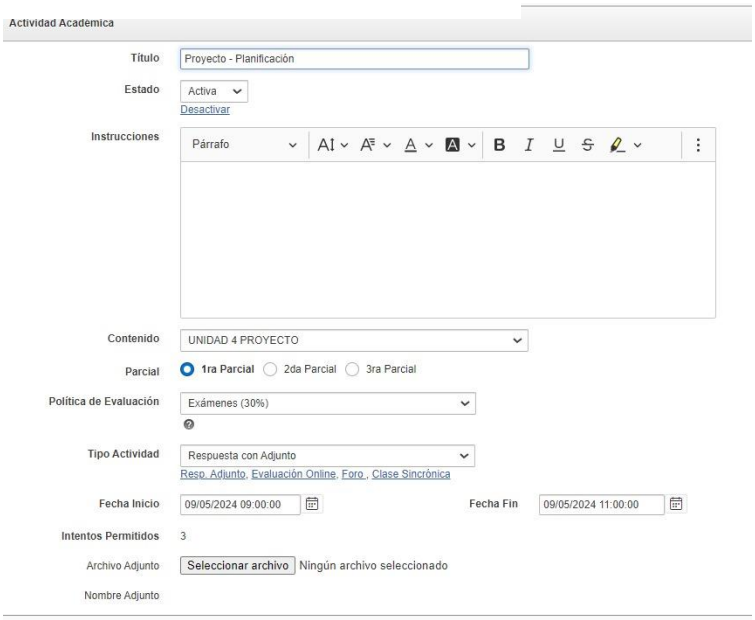
Fuente: Propia

Durante el transcurso de la segunda fase, al igual que, en la primera fase, los sujetos de prueba tuvieron acceso a los recursos didácticos por lo que se procedió a dividir el taller en tres etapas en los que los estudiantes deben identificar las fases de planificación, desarrollo y postdesarrollo e implementar en cada una la documentación solicitada. Una vez los

estudiantes realizaron los respectivos grupos, a cada grupo se le asigno una metodología a trabajar en base a esto la asignación quedó de la siguiente manera, se dispuso que dos grupos trabajen con XP, dos grupos con Kanban y un grupo con Scrum.

El taller solicitaba levantar requerimientos por lo que cada metodología constaba con realizar historias de usuario para comprender la explicación del usuario final o el cliente. el desarrollo de esta fase se procedió a generar una actividad en el Sistema Integral de Información Universitaria (SIIU) en la que los estudiantes debían adjuntar evidencia de la primera etapa que consistía en identificar y levantar los requerimientos solicitados por el taller que se les compartió, cada grupo debía realizarlo utilizando los recursos solicitados por cada metodología. En la Figura 14 se observa la actividad a realizar por los sujetos experimentales.

Figura 14
Actividad SIIU de la segunda fase



Nota. Adaptado del SIIU.

- **Scrum**

En el transcurso de esta fase los sujetos de prueba debían identificar los artefactos necesarios y eventos a utilizar durante la etapa de planificación correspondiente a cada metodología asignada, en este caso Scrum, fue trabajado por un equipo conformado por 4 personas. En la Figura 15 se puede apreciar la estructura del Scrum Team.

Figura 15
Scrum Team

Rol	Nombre	Responsabilidad
Product Owner	Juan Diego Iza	Cliente
Scrum Máster	Jordan Puruncajas	Dirigir al equipo
Equipo de desarrollo	Kevin Guacanes Tatiana Quilca Masciel Sevilla	Desarrollar el proyecto

Fuente: Equipo Scrum.

El equipo debía realizar una planificación en base a las horas autónomas y horas de clase establecidas por el syllabus. En la Figura 16 se puede identificar la planificación del Scrum Team.

Figura 16
Planificación de Sprints

N°. Sprint	Fecha de inicio	Fecha fin	Duración(horas)
Sprint 1	07/05/2024	11/05/2024	40
Sprint 2	12/05/2024	16/05/2024	40

Fuente: Equipo Scrum.

Para el levantamiento de requisitos los estudiantes debían analizar el documento compartido en el que se comparte el ejercicio detallando los requerimientos funcionales para el desarrollo del taller durante el transcurso de las sesiones, para el levantamiento el equipo usó historias de usuario para detallar los requisitos en lenguaje

informal o común. En la Figura 17 y Figura 18 se puede observar el proceso algunas de las historias de usuario levantadas por el grupo Scrum.

Figura 17
Historia de usuario 1

Historias de usuario	
Número: 1	**Usuario: Scrum Team**
Nombre Historia: Levantamiento de requisitos.	
Prioridad: Alta	
Descripción: Como scrum team, necesito poder recolectar los requisitos del cliente.	
Criterios de aceptación: • Se planifica una reunión con el producto owner. • El scrum team se reúne en una entrevista con el product owner para recolectar las necesidades que requiere el sistema. • Se organiza la información recolectada y se trabaja con los artefactos de la metodología scrum.	

Fuente: Equipo Scrum.

Figura 18
Historia de usuario 4

Historias de usuario	
Número: 4	**Usuario:** Jefe del desarrollo
Nombre Historia: Registro de información.	
Prioridad: Alta	
Descripción: Como jefe de Desarrollo, quiero registrar la información básica del proyecto (nombre del cliente, nombre del proyecto, fechas de inicio y fin, porcentaje de avance y estado) para tener un control y seguimiento de los proyectos.	
Criterios de aceptación: • Todos los campos de información básica del proyecto deben ser obligatorios y validados por el sistema. • La información registrada debe ser completa y precisa. • El sistema debe permitir editar la información básica del proyecto en caso de cambios o actualizaciones: • La información debe estar disponible para consulta en cualquier momento. • El sistema debe garantizar la confidencialidad de los datos de costos del proyecto.	

Fuente: Equipo Scrum.

La Figura 20 indica el producto backlog realizado por los sujetos de prueba del grupo Scrum, en el que se evidencia la prioridad y estimación de esfuerzo en horas a las historias de usuario identificadas en el proceso.

Figura 20
Product Backlog

Orden	ID	Descripción	Prioridad A(alta), M(media), B(baja).	Estimación (horas)
1	HU-01	Levantamiento de requisitos.	A	7
2	HU-02	Ingresar al sistema.	A	9
3	HU-03	Aprobar y firmar nuevo proyecto	A	12
4	HU-04	Registro de información.	A	12
5	HU-05	Asignación de Tareas a Analistas de Sistemas.	M	15
6	HU-06	Listar de proyectos.	M	15
7	HU-07	Cerrar proyectos completados.	B	10

Nota. Elaborado por equipo Scrum.

En la Figura 19 se puede observar el desarrollo del Sprint Planning en el que se identifica las historias de usuario, el número de desarrolladores, el producto owner, fecha de inicio, fecha fin, scrum máster, scrum team, total de números trabajadas por sprint, responsabilidad de cada desarrollador, fase de desarrollo, tarea, tiempo estimado, tiempo real, estado y comentario.

Figura 19
Sprint Backlog

SPRINT BACKLOG

Proyecto:							
Sprint	**1**		**Team**	H. autónomas	H. clase	Horas total	Horas total
Product Owner:	Juan Diego Iza		Guacanes Diaz Kevin Andre	5	5	10	20
Scrum Máster:	Puruncajas Jordan		Puruncajas Castillo Jordan Steven				
Nro. Desarrolladores	4		Quilca Burgos Tatiana Maricela				
Total horas	**20**		Sevilla Erazo Melany Masciel				
Fecha Inicio SP1:	05/07/2024						
Fecha Final SP1:	05/11/2024						

PLANIFICACIÓN					EJECUCIÓN		
Actividad / Historia de Usuario	Desarrollador	Fase Desarrollo	TAREA	TIEMPO ESTIMADO (Horas)	TIEMPO REAL (Horas)	ESTADO	COMENTARIO
Levantamiento de requisitos.	Scrum team	Analisis	Asistir a la reunion con el product owner.	2	1	Hecho	
	Scrum team	planificación	Crear las historias de usuario y planificar los sprint.	5	6	Hecho	
Ingresar al sistema.	Jordan Puruncajas	Diseño	Diseñar la interfaz web de inicio de sesión con campos para usuario y contraseña	2	2	En proceso	
	Tatiana Quilca	Validacion	Implementar la validación de los campos de usuario y contraseña	3	2	Pendiente	
	Kevin Guacanes	Diseño	Mostrar la interfaz correspondiente al rol del usuario autenticado	4	5	Pendiente	
Aprobar y firmar nuevo proyecto	Kevin Guacanes	Diseño	Diseñar la interfaz web para aprobación de proyectos	3	2	Pendiente	
	Jordan Puruncajas	Validación	Validar campos correctos.	2	2	Pendiente	
	Masciel Sevilla	Diseño	Guardar la información del contrato.	4	4	Pendiente	
	Tatiana Quilca	Diseño	Archivar la información del cliente.	3	4	Pendiente	
Registro de información.	Masciel Sevilla	Diseño	Diseñar la interfaz web para ingresa la informacion de proyectos	4	5	Pendiente	
	Tatiana Quilca	Validación	Implementar la validación de todos los campos obligatorios del proyecto para garantizar que estén completos antes de enviarlos al sistema.	4	3	Pendiente	
	Kevin Guacanes	Diseño	Desarrollar la funcionalidad para almacenar la información ingresada en la base de datos del sistema.	4	4	Pendiente	
Tareas no planificadas			**TOTAL**	**40**	**40**		

Nota. Elaborado por equipo Scrum.

- Extreme Programming

De acuerdo con la planificación de la segunda fase en esta metodología trabajaron dos grupos, cada grupo conformado por cuatro personas las cuales debían identificar la etapa de planificación y el uso adecuado de eventos y de sus correspondientes artefactos. En la Figura 21 se puede observar la identificación de roles y color de esferos respectivos para que haya una diferenciación de participantes.

Figura 21
Roles de Extreme Programming

Nombre	Rol	Color esfero
Juan Diego Iza	Cliente	
PhD. Antonio Quiña	Experto en la temática	Rojo
Francisco Alban	Desarrollador en Software	Negro obscuro
Francis Guevara	Desarrollador en Software	Naranja
Lizbeth Quilumba	Desarrollador en Software	Celeste
Verónica Saransig	Desarrollador en Software	Verde

Nota. Identificación de roles elaborada por un grupo experimental de la metodología XP.

Al igual que el grupo de metodología Scrum, los grupos de XP debían realizar levantamiento de requisitos para esto debían analizar el documento compartido en el que se comparte el ejercicio detallando los requerimientos funcionales para el desarrollo del taller durante el transcurso de las sesiones, para el levantamiento el equipo usó historias de usuario para detallar los requisitos en lenguaje informal o común. Cabe recalcar que cada grupo identificó 9 historias de usuario. En la Figura 22 y Figura 23 se puede observar el proceso de algunas de las historias de usuario levantadas por el grupo XP.

Figura 22
Historia de usuario 006

Historias de usuario	
Número: H-006	**Usuario:** Analista de Sistemas
Nombre Historia: Asignación de Tareas a Analistas de Sistemas	
Prioridad: Alta	
Descripción: Como analista de sistemas, quiero poder asignar una o varias tareas a un equipo de Analistas de Sistemas, incluyendo el nombre de la tarea, las fechas de inicio y fin, el porcentaje de avance y el analista al que fue asignada la tarea, para distribuir eficientemente el trabajo y monitorear su progreso.	
Criterios de aceptación: - El sistema debe permitir al usuario asignar tareas a los analistas de sistemas una vez que se haya registrado la información básica del proyecto. - Se deben proporcionar campos para registrar el nombre de la tarea, las fechas de inicio y fin, el porcentaje de avance y el analista al que fue asignada la tarea. - El usuario debe poder guardar la información de la asignación de tareas. - Debe ser posible editar o eliminar las asignaciones de tareas en cualquier momento.	

Nota. Historia de usuario elaborada por el grupo experimental XP.

Figura 23
Historia de usuario 009

Historias de usuario	
Número: H-009	**Usuario:** Analista de Sistemas
Nombre Historia: Validación de campos y pruebas	
Prioridad: Alta	
Descripción: Como Analista de Sistemas, quiero que el sistema realice validaciones de los campos ingresados y realice pruebas cuando sea necesario, para garantizar la calidad del servicio que ofrecemos en Nexus y evitar errores en la gestión de proyectos.	
Criterios de aceptación: • El sistema debe realizar validaciones de los campos ingresados para asegurar la integridad de los datos. • Debe ser posible configurar reglas de validación específicas para cada campo. • Se deben realizar pruebas automáticas cuando sea necesario, como parte del proceso de gestión de proyectos. • El usuario debe recibir notificaciones sobre los resultados de las pruebas realizadas.	

Nota. Historia de usuario elaborada por el grupo experimental XP.

Los sujetos de prueba debían realizar la fase de *Release Planning* en la que debían realizar una planificación dónde los desarrolladores y el grupo de trabajo establecen los tiempos de implementación de acuerdo con la estimación de esfuerzo de cada uno y de acuerdo con las historias de usuario. A continuación, en la Figura 24 se puede observar el Release planning ejecutado por un grupo.

Figura 24
Release Planning

Iteración	N° Historia	Prioridad	Tiempo estimado(horas)	Historia de usuario
1	H-001	Alta	10H (2:30H x C/U)	Recopilación de Requisitos del Cliente
1	H-002	Alta	16H (4:00H x C/U)	Planificación de Proyecto
1	H-003	Alta	14H (3:30H x C/U)	Creación de Prototipo del Proyecto
2	H-004	Alta	4H (2:00H x C/U) (Pair Programing)	Recolección de Requisitos del Cliente
2	H-005	Alta	8H (4:00H x C/U) (Pair Programing)	Registro de información Básica del Proyecto
2	H-006	Alta	8H (4:00H x C/U) (Pair Programing)	Asignación de Tareas a Analistas del Sistema
2	H-007	Alta	8H (4:00H x C/U) (Pair Programing)	Consulta de estado de Proyectos
2	H-008	Alta	8H (4:00H x C/U) (Pair Programing)	Cierre de Proyecto en Reuniones
2	H-009	Alta	4H (1:00H x C/U)	Validación de Campos y Pruebas

Nota. Release Planning elaborado por un grupo experimental XP

Para complementar la etapa los sujetos experimentales debían realizar una planificación estimando las horas de trabajo, horas semanales de trabajo, semanas laboradas y horas estimadas trabajas en el proyecto de acuerdo con los tiempos establecidos por la asignatura. En la Figura 25 se puede observar los cálculos para cumplir el plazo establecido.

Figura 25
Planificación horas de trabajo

Horas de trabajo	Horas semanales de trabajo	Semanas laboradas	Horas trabajas en el proyecto
80	40	2	80

Nota. Planificación de horas de trabajo destinadas en el proyecto elaborado por grupo experimental XP.

- Kanban

De acuerdo con la planificación de la segunda fase en esta metodología trabajaron dos grupos, cada grupo conformado por cuatro personas las cuales debían identificar la etapa de planificación y el uso adecuado de eventos y de sus correspondientes artefactos. En la Figura 26 Se puede observar la identificación de responsabilidades, ya que en esta metodología no existen roles específicos.

Figura 26
Designación de responsabilidades Kanban

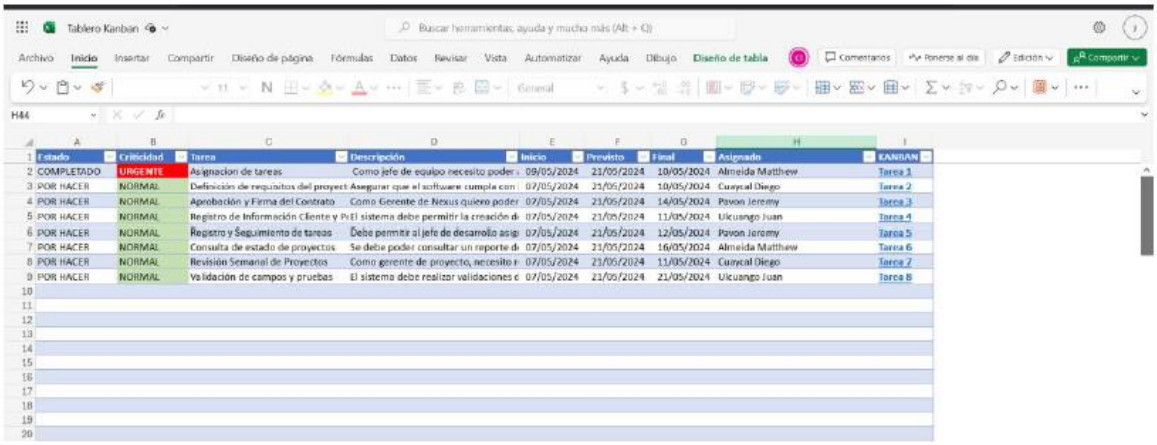
Nombre	Responsabilidad
Joan Pastillo	Gerente del Proyecto - Responsable de la planificación general, aprobación de proyectos y seguimiento de progreso.
Alexander Paspuel	Analista de Sistemas - Desarrolla y prueba las tareas asignadas, asegurando la calidad y funcionalidad.
Omar Simbaña	Técnico de Base de Datos - Encargado de las conexiones a bases de datos y mantenimiento del servidor de aplicaciones.
Bixmarck Rodríguez	Jefe de Desarrollo - Maneja el registro y actualización de proyectos, asignaciones de tareas y reportes de estado.

Nota. Elaborado por el grupo experimental Kanban.

Para trabajar con la metodología Kanban se trabajó con un tablero Kanban realizado en Excel, por lo que los sujetos de prueba debían levantar las columnas necesarias con sus respectivas tarjetas Kanban para la implementación de esta metodología. A continuación, en la Figura 27 y Figura 28 se observa el levantamiento de tablero Kanban realizado por un grupo. Cabe recalcar que al igual que las otras metodologías los sujetos experimentales debían examinar el documento que indicaba los requerimientos funcionales solicitados en el taller y en la etapa de planificación por lo que los grupos designados también debían realizar historias de usuario, en base a

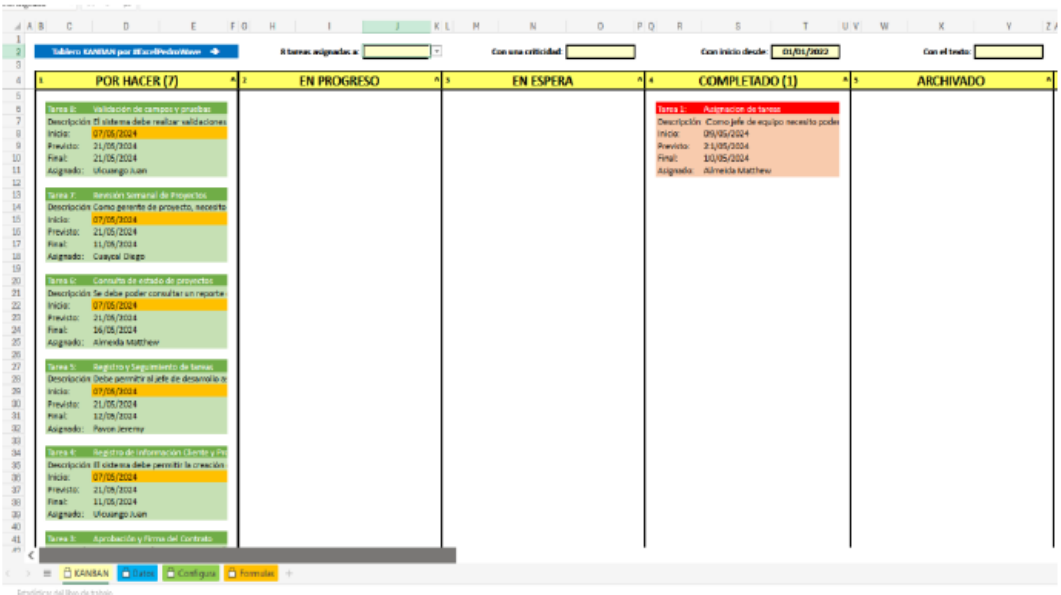
esto un grupo determinó que eran necesarias seis historias de usuario mientras que el otro grupo identificó ocho.

Figura 27
Creación de Tarjetas Kanban



Nota. Elaborado por grupo experimental Kanban.

Figura 28
Levantamiento de Tablero Kanban



Nota. Elaborado por grupo experimental Kanban.

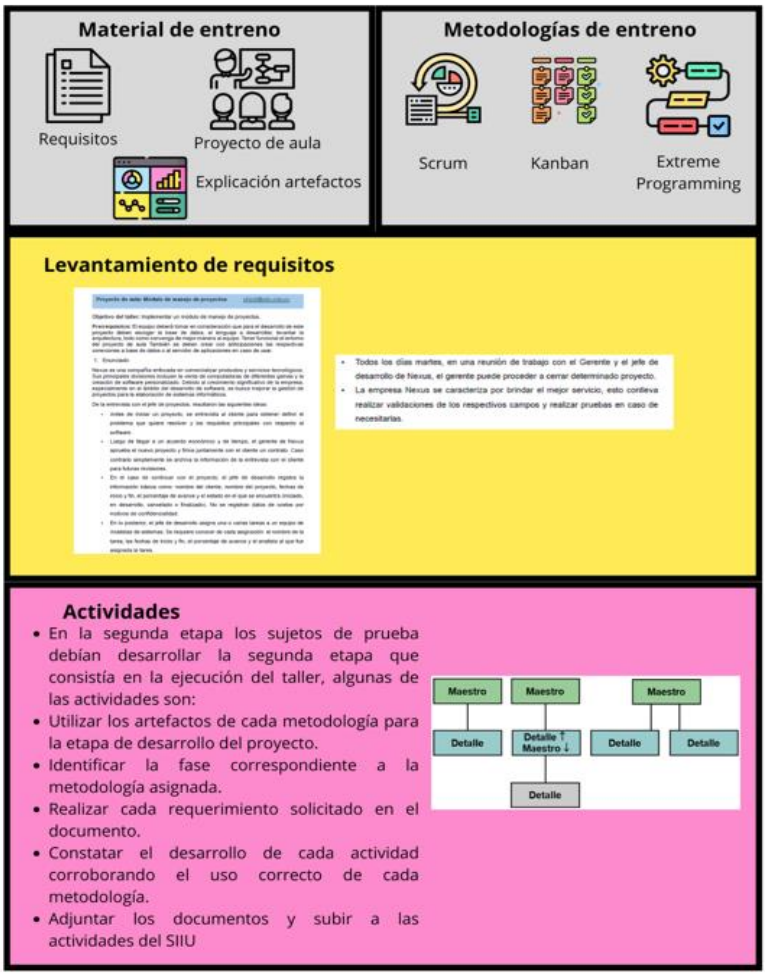
Tercera Fase

En el transcurso de esta fase, los estudiantes debían proceder con el progreso de la segunda etapa que corresponde al desarrollo del ejercicio aplicando el uso de los artefactos correspondientes a cada metodología, los sujetos debían identificar que requerimientos

solicitaban más prioridad, analizar que herramientas iban a usar, lenguaje, base de datos, entre otros.

La actividad constaba con evidenciar el uso de las metodologías asignadas por cada grupo mediante capturas de pantalla, el uso de artefactos, analizando sus eventos y completando los requerimientos solicitados. Así mismo, se asignó una actividad en la que los sujetos de prueba debían anexar y subir al SIIU. En la Figura 29 se observa la estructura de la tercera fase.

Figura 29
Estructura de la tercera fase



Fuente: Elaboración Propia.

Durante el desarrollo de esta fase se utilizaron varias prácticas correspondientes a cada metodología, por ejemplo, en la Figura 30 se observa como el grupo correspondiente a la metodología de Extreme Programming utilizó la programación en parejas.

Figura 30
Pair Programming

Nota. Elaborado por grupo experimental XP.

Con respecto al grupo experimental que utilizó la metodología Scrum se pudo observar que una de las prácticas más destacadas fue el uso del Sprint Retrospective tal como se logra identificar en la Figura 31.

Figura 31
Sprint Retrospective

Metodología SCRUM

Formulario de reunión retrospectiva

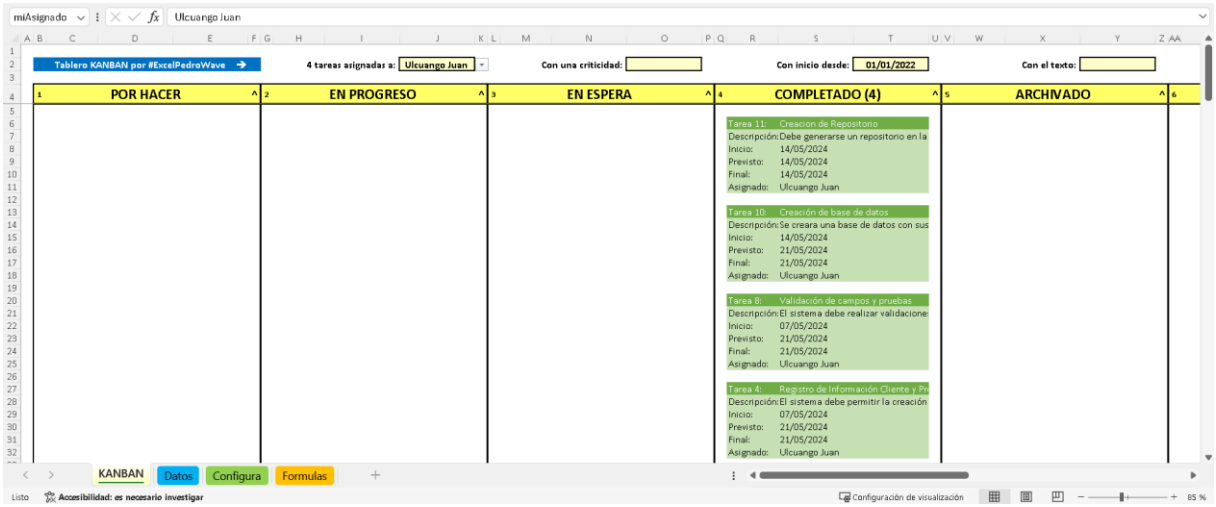
Masciel Sevilla

¿Qué salió bien en la iteración? (aciertos)	¿Qué no salió bien en la iteración? (errores)	¿Qué mejoras vamos a implementar en la próxima iteración? (recomendaciones de mejora continua)
• Conocí acerca de los comando de git y como se los ocupa para el trabajo colaborativo. • Las pruebas de aceptación fueron exitosas y cumplieron con los criterios de aceptación que establecimos en las historias de usuario. • Aprendí más acerca de los componentes react y también algunas cosas nuevas de JavaScript.	• Mala organización de tiempo en mis tareas . • Durante la realización de las tareas surgieron errores no previstos. • Debido al corto tiempo el diseño de las interfaces no es muy vistoso.	• Implementar una planificación más detallada y precisa • Establecer un sistema de seguimiento de errores más riguroso, como la implementación de pruebas unitarias • Dedicaremos más tiempo y recursos al diseño de interfaces. • Continuar aprendiendo sobre componentes React y JavaScript avanzado

Nota. Elaborado por grupo experimental Scrum.

Finalmente, en el grupo que trabajó con la metodología Kanban se pudo apreciar que una de las prácticas más relevantes fue limitar el trabajo en proceso (WIP) a un desarrollador, tal como se aprecia en la Figura 32.

Figura 32
Limitar el trabajo



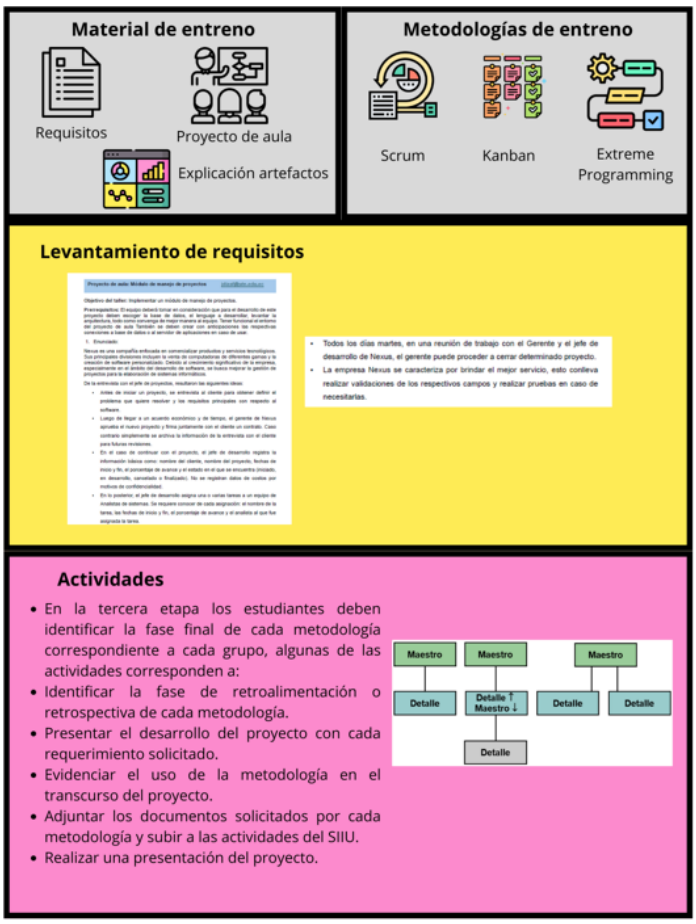
Nota. Elaborado por grupo experimental Kanban.

Cuarta Fase

En la cuarta fase, la fase de retroalimentación, los sujetos realizaron de manera grupal un ejercicio de retrospectiva correspondiente al uso de la metodología asignada en cada grupo. Algunas de las preguntas correspondientes al formulario de retrospectiva fueron las siguientes: ¿Qué salió bien en la iteración? (aciertos), ¿Qué no salió bien en la iteración? (errores) y ¿Qué mejoras vamos a implementar en la próxima iteración? (recomendaciones de mejora continua).

La intención de realizar la retroalimentación era tener idea de cómo se estaba efectuando y ejecutando la metodología en cada grupo para identificar los éxitos y fracasos, siendo este importante para los grupos de tal forma que puedan analizar su propio desempeño e identificar estrategias para mejorar sus procesos. La revisión de esta retrospectiva se realizó en conjunto con el director de trabajo de investigación.

Figura 33
Estructura de la cuarta fase



Fuente: Elaboración Propia.

2.3.6 Instrumentos

Para el desarrollo de las fases y capacitación se utilizaron varias herramientas de ofimática, sin embargo, para el ejercicio los sujetos de prueba fueron libres de utilizar cualquier herramienta de programación, patrones de diseño o arquitectura de software. En base a la recopilación de datos obtenidos de la documentación se evidenció que se utilizaron algunas herramientas, las más comunes fueron:

- Visual Studio Code: es un editor de código fuente ligero el cual es rico en extensiones para varios lenguajes(code.visualstudio, 2024)
- PostgreSQL: es una base de datos relacional open source(Postgresql.org, 2024)

- PgAdmin4: es un administrador de base de datos para PostgreSQL la cual es open source(Postgresql.org, 2024)
- JavaScript: es un lenguaje de programación web ligero(Developer.mozilla.org, 2024)

Para medir cada una de las características del modelo de calidad definido se estableció dos instrumentos de levantamiento de datos: (i) taller práctico para levantar datos de eficacia y (ii) encuesta SUS para medir la satisfacción del usuario.

2.4 Operacionalización del experimento

2.4.1 Muestra

El experimento se llevó a cabo de manera presencial durante 19 días. Los sujetos de la prueba fueron estudiantes de la materia de Construcción de Software de la carrera de Ingeniería en Software de la Universidad Técnica del Norte. Se llevó a cabo durante 6 clases desde el 2 de mayo hasta el 21 de mayo de 2024.

2.4.2 Preparación

En su mayoría, los sujetos del experimento llegaron a tiempo. Al principio se les dieron instrucciones sobre las tareas a realizar y cómo preparar los materiales para el ejercicio y el uso de artefactos correspondiente a cada metodología.

Más de la mitad de las personas no tenían experiencia previa con el manejo de las metodologías, que incluyen artefactos, eventos, roles, prácticas, principios y valores a excepción de Scrum. Los estudiantes conocían la teoría de algunas metodologías, pero no las habían aplicado, a diferencia de Scrum.

Realizamos el experimento según el cronograma. Los participantes del experimento pidieron explicaciones sobre la actividad. Los sujetos experimentales desarrollaron el ejercicio compartido de manera completamente autónoma. Se estimó que tardaría tres semanas en

resolver el taller. A continuación, en la Tabla 11 se detalla el proceso de la ejecución del experimento.

Tabla 11
Ejecución del experimento

Fecha	Hora inicio	Hora fin	Actividad
02/05/2024	9:00	10:30	Inicio del taller, Explicar instrucciones iniciales, capacitar sobre las metodologías
07/05/2024	7:00	10:00	Explicar ejercicios prácticos de las metodologías
09/05/2024	9:00	11:00	Asistir como dueño del producto, Aclarar dudas
14/05/2024	7:00	10:00	Aclarar dudas, Explicar ejercicio
16/05/2024	9:00	11:00	Evidenciar el uso de las metodologías
21/05/2024	7:00	10:00	Evidenciar la etapa de desarrollo del taller

23/05/2024	9:00	11:00	Realizar retroalimentación del ejercicio, Aplicar encuesta SUS, Fin del taller

Elaboración: Propia.

Al empezar el taller se procedió a entregar todo el material necesario para el uso de las metodologías incluyendo requisitos, artefactos, material de capacitación, actividades, cronograma, todo esto de acuerdo a los requisitos solicitado en las actividades, por lo que los sujetos constaban con la información y material pre cargado para realizar el experimento, al momento de la entrega de requisitos los sujetos de prueba procedieron a reunirse en grupos de 4 dando un total de 5 grupos, durante el transcurso de las clases debían realizar las actividades solicitadas, además, se precautelo que todos entendieran los requisitos para un mejor manejo de metodologías y comprensión del ejercicio.

2.5 Recolección de resultados

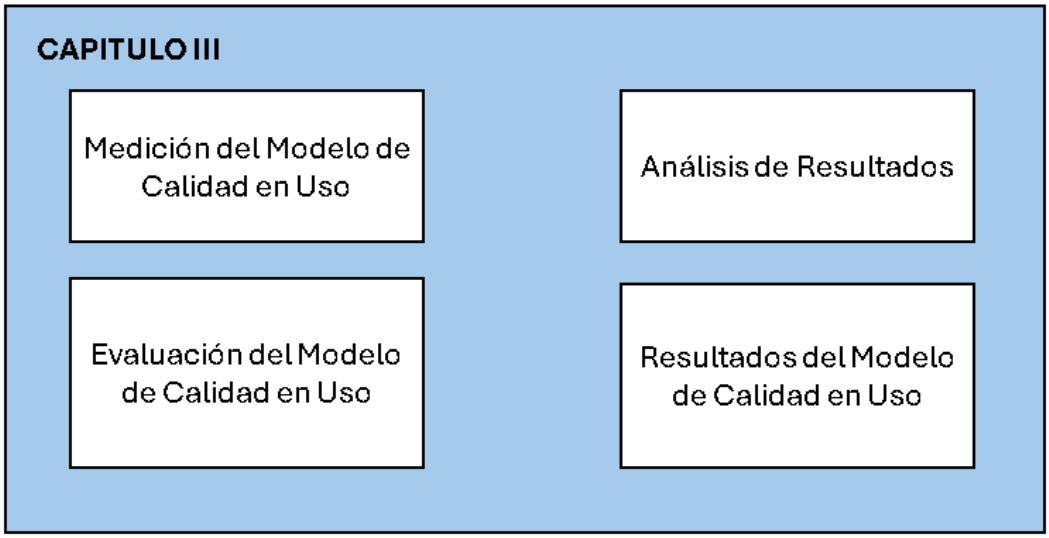
Al terminar cada sesión se procedió a dar un tiempo aproximado de 15 minutos para que suban las actividades correspondientes, esto en base a los resultados del proceso de ejecución del experimento explicado en la Tabla 11. Se procedió a adjuntar y corroborar la evidencia de cada grupo para comprobar si no hay errores.

CAPÍTULO 3

Validación de resultados

Este capítulo presenta los hallazgos del experimento controlado realizado para comparar la eficacia de las metodologías Scrum, Extreme Programming y Kanban utilizando un modelo de calidad en uso basado en las normas ISO/IEC 25022. La estructura del contenido del capítulo se muestra en la Figura 34.

Figura 34
Estructura del tercer capítulo



Fuente: Elaboración propia.

3.1 Medición del Modelo de Calidad en Uso

Para realizar la medición de la calidad en uso, se basó en la ISO/IEC 25022 (ISO/IEC 25022, 2016), que muestra la manera de como medir cada una de las subcaracterísticas del modelo de calidad definido en la Tabla 7. Para obtener los elementos de las métricas se estableció dos instrumentos de levantamiento de datos: (i) proyecto de aula para levantar datos de eficacia; (ii) encuesta sus para medir la satisfacción del usuario.

3.2 Análisis de resultados

Los resultados experimentales brindan datos de gran relevancia para proseguir con la fase de evaluación del modelo de calidad en uso basado en la ISO/IEC 25022, y comparar la eficacia de las metodologías ágiles Scrum, Kanban y Extreme Programming, para realizar el análisis se utilizó los resultados obtenidos por los sujetos experimentales de acuerdo con el proceso mencionado en el Capítulo 2.4 y 2.5.

3.2.1 Encuesta SUS

La encuesta se la realizó después ejecutar el taller posterior a la retroalimentación, la cual tuvo una duración aproximada de 20 minutos, a continuación, se detalla el diseño de la encuesta.

- Diseño de la Encuesta SUS

  SUS (System Usability Scale) – Escala de Usabilidad del Sistema) propone 10 preguntas para medir la usabilidad de un sistema, o en el caso del proyecto la metodología. Para el sistema de medición se va a usar la escala Likert, dónde 1 significa *Totalmente en desacuerdo* y 5 *Totalmente de acuerdo.* Estas preguntas permitieron levantar datos para medir las subcaracterísticas de utilidad, confianza y comodidad, dónde las preguntas P1, P2 y P3 fueron seleccionadas para la utilidad, las preguntas P4, P5, P6, P7, P8 y P9 fueron apartadas para la comodidad y las preguntas P10 y P11 para la confianza. A continuación, se detalla el enunciado de cada una de ellas:

  P1: ¿Considera usted que usaría esta metodología frecuentemente?

  P2: ¿Considera usted que la metodología es consistente?

  P3: ¿Se siente satisfecho con el uso de la metodología en su proyecto de aula?

  P4: ¿Considera usted que la metodología es fácil de usar?

  P5: ¿Considera usted qué los artefactos y eventos de la metodología son fáciles de usar?

  P6: ¿Considera usted que necesitaría ayuda de una persona con conocimientos técnicos para usar esta metodología?

P7: Determine el esfuerzo de uso de la metodología de acuerdo con la escala. 1) Muy Ligero 2) Ligero 3) Normal 4) Duro 5) Muy duro

P8: ¿Considera usted que necesita aprender muchas cosas antes de usar esta metodología?

P9: ¿Considera usted que la mayoría de la gente aprendería a usar esta metodología en forma rápida?

P10: ¿Se siente confiado al usar esta metodología?

- Ejecución de la encuesta

  a) La encuesta se diseñó en Google Forms.
  b) Después de realizar el proyecto de aula los sujetos ejecutaron la encuesta.

- Observaciones

  El 100% de los encuestados respondieron la encuesta que corresponde a los 20 sujetos experimentales de la asignatura de Construcción de Software.

En coordinación con el director PhD. Antonio Quiña, para el análisis de resultados se estandarizó cuatro etapas (planificación, ejecución, verificación y retrospectiva) en los que cada uno de los grupos debía analizar y adaptar acorde al uso de artefactos y prácticas sugerida por cada metodología asignada, en base a esto se logró una normalización de los procesos que debía seguir cada grupo y por ende facilitó la evaluación de cada marco de trabajo.

3.2.2 Análisis de resultados Scrum

En la Tabla 12 se puede identificar los resultados conseguidos del experimento, en donde, se utilizó una escala de tres parámetros: passed (completo la tarea), failed (fallo en completar la tarea) y por último not done (no completó la tarea), de esta forma se facilitó la tabulación de datos para aplicar las métricas de evaluación en los resultados del experimento.

Tabla 12
Resultados Scrum

Scrum										
Planeación						Ejecución		Verificación		Retrospectiva
Nro. Estudiante	Equipo Scrum	Planificación	Historias de usuarios	Product Backlog	Sprint Planning	Codificación	Criterios de aceptación	Reuniones	Evidencia codificación/Sprints	Retroalimentación
1	passed	passed	failed	passed	failed	passed	passed	passed	passed	passed
2	passed	passed	failed	passed	failed	passed	passed	passed	passed	passed
3	passed	passed	failed	passed	failed	passed	passed	passed	passed	passed
4	passed	passed	failed	passed	failed	passed	passed	passed	passed	passed

Fuente: Elaboración Propia.

Para una mejor visualización de los resultados se ha evaluado las actividades señaladas como “failed” o “falló en completar la tarea”, de una mejor manera en la que se indica los porcentajes realizados por los grupos y que reflejan cuál fue el nivel alcanzado durante la actividad, tal como se observa en la Tabla 13.

Tabla 13
Porcentajes de actividad - Scrum

	Actividad		Porcentaje obtenido
Scrum	Planeación	Equipo Scrum	100%
		Planificación	100%
		Historias de usuarios	90%
		Product Backlog	100%
		Sprint Planning	85%
	Ejecución	Codificación	100%
		Criterios de aceptación	100%
	Verificación	Reuniones	100%
		Evidencia codificación/Sprints	100%
	Retrospectiva	Retroalimentación	100%
	Total		97,5%

Elaboración: Propia.

En la Tabla 13 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de planificación (Equipo Scrum), teniendo como resultados que el 100% de las evaluaciones de la fase de planificación centrada en la identificación del equipo Scrum fue calificada como "passed", indicando que los sujetos si lograron realizar con éxito el requerimiento.

En la Tabla 13 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de planificación (Planificación), teniendo como resultados que el 100% de las evaluaciones de la fase de planificación centrada en la organización utilizando Scrum fue calificada como "passed"", indicando que los sujetos si lograron realizar con éxito el requerimiento.

En la Tabla 13 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de planificación (Historias de usuario), teniendo como resultados que las evaluaciones de la fase de planificación centrada en las historias de usuario fueron calificadas como "failed", recalcando que la actividad fue completada con un 90%, indicando que los sujetos fallaron en intentar realizar el requerimiento solicitado.

En la Tabla 13 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de planificación (Product Backlog), teniendo como resultados que el 100% de las evaluaciones de la fase de planificación centrada en el Product Backlog fue calificada como "passed", indicando que los sujetos si lograron realizar con éxito el requerimiento.

En la Tabla 13 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de planificación (Sprint Planning), teniendo como resultados las evaluaciones de la fase de planificación centrada en el Sprint Planning fue calificada como "failed", señalando que la actividad fue completada en un 85%, indicando que los sujetos fallaron en intentar realizar el requerimiento solicitado.

En la Tabla 13 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de ejecución (codificación), teniendo como resultados que el 100% de las evaluaciones de la fase de ejecución centrada en la codificación fue calificada como "passed", indicando que los sujetos si lograron realizar con éxito el requerimiento.

En la Tabla 13 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de ejecución (criterios de aceptación), teniendo como resultados que el 100% de las evaluaciones de la fase de ejecución centrada en los criterios de aceptación fue calificada como "passed"," indicando que los sujetos si lograron realizar con éxito el requerimiento.

En la Tabla 13 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de revisión (reuniones), teniendo como resultados que el 100% de las evaluaciones de la fase de revisión centrada en las reuniones fue calificada como "passed", indicando que los sujetos si lograron realizar con éxito el requerimiento.

En la Tabla 13 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de revisión (código / sprints), teniendo como resultados que el 100% de las evaluaciones de la fase de revisión centrada en el código y los sprints fue calificada como "passed", indicando que los sujetos si lograron realizar con éxito el requerimiento.

En la Tabla 13 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de retrospectiva (retroalimentación), teniendo como resultados que el 100% de las evaluaciones de la fase de retrospectiva centrada en la retroalimentación fue calificada como "passed", indicando que los sujetos si lograron realizar con éxito el requerimiento.

En definitiva, se puede observar que un 97,5% de actividades fue completada por parte de los sujetos experimentales que trabajaron con la metodología Scrum, señalando que las actividades que tuvieron más dificultades fueron las historias de usuarios y el Sprint Planning.

3.2.3 Análisis de resultados Extreme Programming

En la Tabla 14 se puede identificar los resultados conseguidos del experimento, en donde, se utilizó una escala de tres parámetros: passed (completo la tarea), failed (fallo en completar la tarea) y por último not done (no completó la tarea), de esta forma se facilitó la tabulación de datos para aplicar las métricas de evaluación en los resultados del experimento.

Tabla 14
Resultados Extreme Programming

Extreme Programming											
Planificación					Ejecución		Verificación				Retrospectiva
Nro. Estudiante	Roles	Planificación	Historias de usuarios	Iteración - historias de usuario	Herramientas de programación	Mockups - avances	Codificación	Pruebas de aceptación	Pair Programming	Iteración	Retroalimentación
1	passed	passed	failed	passed	passed	passed	passed	passed	failed	passed	passed
2	passed	passed	failed	passed	passed	passed	passed	passed	failed	passed	passed
3	passed	passed	failed	passed	passed	passed	passed	passed	failed	passed	passed
4	passed	passed	failed	passed	passed	passed	passed	passed	failed	passed	passed
5	passed	passed	failed	passed	passed	failed	passed	passed	not done	failed	passed
6	passed	passed	failed	passed	passed	failed	passed	passed	not done	failed	passed
7	passed	passed	failed	passed	passed	failed	passed	passed	not done	failed	passed
8	passed	passed	failed	passed	passed	failed	passed	passed	not done	failed	passed

Elaboración: Propia.

Para una mejor visualización de los resultados se ha evaluado las actividades señaladas como “failed” o “falló en completar la tarea”, de una mejor manera en la que se indica los porcentajes realizados por los grupos y que reflejan cuál fue el nivel alcanzado durante la actividad, tal como se observa en la **Tabla *15***.

Tabla 15
Porcentajes de actividad - Extreme Programming

		Porcentaje obtenido			Promedio total
		Actividad	Grupo 1	Grupo 2	
Extreme Programming	Planificación	Roles	100%	100%	100%
		Planificación	100%	100%	100%
		Historias de usuarios	75%	80%	78%
		Iteración - historias de usuario	100%	100%	100%
	Ejecución	Herramientas de programación	100%	100%	100%
		Mockups - avances	100%	85%	93%
	Verificación	Codificación	100%	100%	100%
		Pruebas de aceptación	100%	100%	100%
		Pair Programming	85%	0%	43%
		Iteración	100%	90%	95%
	Retrospectiva	Retroalimentación	100%	100%	100%
	Total		96,36%	86,82%	91,6%

Elaboración: Propia.

En la Tabla 15 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de planificación (roles) teniendo como resultados que el 100% de las evaluaciones de la fase de planificación centrada en la identificación de los roles en XP fue calificada como "passed", indicando que los sujetos de ambos grupos de trabajo si lograron realizar con éxito el requerimiento, teniendo como promedio un total de 100% en esta actividad.

En la Tabla 15 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de planificación (planificación) teniendo como resultados que el 100% de las evaluaciones de la fase de planificación centrada en la organización utilizando XP fue calificada como "passed", por otro lado," indicando que los sujetos de ambos grupos de trabajo si lograron realizar con éxito el requerimiento, teniendo como promedio un total de 100% en esta actividad.

En la Tabla 15 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de planificación (historias de usuario) teniendo como resultados que las evaluaciones de la fase de planificación centrada en realización de historias de usuario en XP fueron calificadas como "failed", recalcando que el grupo uno completó el 75% de la actividad y el segundo grupo el 80%, dando como promedio 78% en esta actividad.

En la Tabla 15 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de planificación (planificación de iteración) teniendo como resultados que el 100% de las evaluaciones de la fase de planificación centrada en la planificación de la iteración en XP fue calificada como "passed", indicando que los sujetos de ambos grupos de trabajo si lograron realizar con éxito el requerimiento, teniendo como promedio un total de 100% en esta actividad.

En la Tabla 15 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de ejecución (herramientas de programación) teniendo como resultados que el 100% de las evaluaciones de la fase de planificación centrado en el diseño de herramientas de programación en XP fue calificada como "passed", indicando que los sujetos de ambos grupos

de trabajo si lograron realizar con éxito el requerimiento, teniendo como promedio un total de 100% en esta actividad.

En la Tabla 15 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de ejecución (Mockups) teniendo como resultados que el grupo uno finalizó en un 100% y pasó con éxito el requerimiento la actividad mientras que el segundo grupo un 85% indicando que falló en realizar el requerimiento, dando como promedio un 93% de completar con éxito esta actividad.

En la Tabla 15 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de verificación (codificación) teniendo como resultados que el 100% de las evaluaciones de la fase de verificación centrado la codificación de requerimientos en XP fue calificada como "passed", indicando que los sujetos de ambos grupos de trabajo si lograron realizar con éxito el requerimiento, teniendo como promedio un total de 100% en esta actividad.

En la Tabla 15 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de verificación (pruebas de aceptación) teniendo como resultados que el 100% de las evaluaciones de la fase de verificación enfocado en realizar las pruebas de aceptación en XP fue calificada como "passed", indicando que los sujetos de ambos grupos de trabajo si lograron realizar con éxito el requerimiento, teniendo como promedio un total de 100% en esta actividad.

En la Tabla 15 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de verificación (Pair Programming) teniendo como resultados que el grupo uno falló en completar con éxito el requerimiento finalizando con un 85% mientras que el segundo grupo no realizó la actividad, obteniendo un promedio total de 43% completado en esta actividad.

En la Tabla 15 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de verificación (Iteración) teniendo como resultados que el grupo uno obtuvo “passed” completado con un 100%, mientras que el segundo grupo falló en completar la actividad con un 90% hecho, teniendo como promedio total de 95% completado en esta actividad.

En la Tabla 15 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de retroalimentación (retrospectiva) teniendo como resultados que el 100% de las evaluaciones de la fase de retroalimentación enfocado en realizar la retrospectiva en fue calificada como "passed", indicando que los sujetos de ambos grupos de trabajo si lograron realizar con éxito el requerimiento, teniendo como promedio un total de 100% en esta actividad.

En definitiva, se puede observar que un 91,6% de actividades fue completada por parte de los sujetos experimentales que trabajaron con la metodología XP, señalando que las actividades que tuvieron más dificultades fueron las historias de usuarios con un porcentaje de 75% y 80% respectivamente con los grupos y el uso del Pair Programming durante el uso de esta metodología obteniendo un 85% y 0% respectivamente con los grupos.

3.2.4 Análisis de resultados Kanban

En la Tabla 16 se puede identificar los resultados conseguidos del proyecto, en donde, se utilizó una escala de tres parámetros: passed (completo la tarea), failed (fallo en completar la tarea) y por último not done (no completó la tarea), de esta forma se facilitó la tabulación de datos para aplicar las métricas de evaluación en los resultados del taller.

Tabla 16
Resultados Kanban

Kanban												
Planificación						Ejecución				Verificación		Retroalimentación
Nro. Estudiante	Equipo Kanban	Levantamiento de tablero	Herramientas de desarrollo	Historias de usuarios	Tarjetas Kanban	Actualización del tablero Kanban	Codificación	Límites del trabajo en proceso	Evidencia Prácticas Kanban	Reuniones	Pruebas de aceptación	Retrospectiva
1	passed	passed	failed	failed	passed	failed	passed	failed	passed	passed	failed	passed
2	passed	passed	failed	failed	passed	failed	passed	failed	passed	passed	failed	passed
3	passed	passed	failed	failed	passed	failed	passed	failed	passed	passed	failed	passed
4	passed	passed	failed	failed	passed	failed	passed	failed	passed	passed	failed	passed
5	passed	passed	failed	failed	passed	passed	passed	not done	passed	passed	passed	passed
6	passed	passed	failed	failed	passed	passed	passed	not done	passed	passed	passed	passed
7	passed	passed	failed	failed	passed	passed	passed	not done	passed	passed	passed	passed
8	passed	passed	failed	failed	passed	passed	passed	not done	passed	passed	passed	passed

Fuente: Elaboración Propia.

Para una mejor visualización de los resultados se ha evaluado las actividades señaladas como “failed” o “falló en completar la tarea”, de una mejor manera en la que se indica los porcentajes realizados por los grupos y que reflejan cuál fue el nivel alcanzado durante la actividad, tal como se observa en la Tabla 17.

Tabla 17
Porcentajes de actividades - Extreme Programming

		Porcentaje obtenido			Promedio total
		Actividad	Grupo 1	Grupo 2	
Kanban	Planificación	Equipo Kanban	100%	100%	100%
		Levantamiento de tablero	100%	100%	100%
		Herramientas de desarrollo	80%	85%	83%
		Historias de usuarios	80%	85%	83%
		Tarjetas Kanban	100%	100%	100%
	Ejecución	Actualización del tablero Kanban	60%	70%	65%
		Codificación	100%	100%	100%
		Límites del trabajo en proceso	70%	0%	35%
		Evidencia Prácticas Kanban	100%	100%	100%
	Verificación	Reuniones	100%	100%	100%
		Pruebas de aceptación	80%	100%	90%
	Retroalimentación	Retrospectiva	100%	100%	100%
	Total		89,17%	86,67%	87,9%

Elaboración: Propia.

En la Tabla 17 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de planeación (Equipo Kanban) teniendo como resultados que el 100% de las evaluaciones de la fase de planificación enfocado asignar e identificar al equipo Kanban fue calificada como "passed", indicando que los sujetos de ambos grupos de trabajo si lograron realizar con éxito el requerimiento, teniendo como promedio un total de 100% en esta actividad.

En la Tabla 17 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de planeación (Levantamiento de tablero) teniendo como resultados que el 100% de las evaluaciones de la fase de planificación enfocado en levantar el tablero Kanban fue calificada como "passed", indicando que los sujetos de ambos grupos de trabajo si lograron realizar con éxito el requerimiento, teniendo como promedio un total de 100% en esta actividad.

En la Tabla 17 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de planeación (herramientas de desarrollo) teniendo como resultados que el grupo uno falló en completar la actividad con un 80% y el segundo grupo de igual manera con un 85%, obteniendo así un promedio total de 83% en esta actividad.

En la Tabla 17 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de planeación (historias de usuario) teniendo como resultados que el grupo uno falló en completar la actividad con un 80% y el segundo grupo de igual manera con un 85%, obteniendo así un promedio total de 83% en esta actividad.

En la Tabla 17 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de planeación (Levantamiento de tablero) teniendo como resultados que el 100% de las evaluaciones de la fase de planificación enfocado en levantar el tablero Kanban fue calificada como "passed", indicando que los sujetos de ambos grupos de trabajo si lograron realizar con éxito el requerimiento, teniendo como promedio un total de 100% en esta actividad.

En la Tabla 17 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de desarrollo (actualización de tablero) teniendo como resultados que el grupo uno falló en completar el requerimiento con un 60% y el segundo grupo de igual manera con un 70%, obteniendo así un promedio total de 65% en esta actividad.

En la Tabla 17 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de desarrollo (codificación) teniendo como resultados que el 100% de las evaluaciones de la fase de ejecución enfocado en la codificación de requisitos, fue calificada como "passed", indicando que los sujetos de ambos grupos de trabajo si lograron realizar con éxito el requerimiento, teniendo como promedio un total de 100% en esta actividad.

En la Tabla 17 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de desarrollo (Límites WIP) teniendo como resultados que el grupo uno falló en completar el requerimiento con un 70% y el segundo grupo de igual manera con un 0%, obteniendo así un promedio total de 35% en esta actividad.

En la Tabla 17 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de desarrollo (evidenciar prácticas Kanban) teniendo como resultados que el 100% de las evaluaciones de la fase de ejecución enfocado en justificar las prácticas de la metodología Kanban, fue calificada como "passed", indicando que los sujetos de ambos grupos de trabajo si lograron realizar con éxito el requerimiento, teniendo como promedio un total de 100% en esta actividad.

En la Tabla 17 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de post desarrollo (reuniones) teniendo como resultados que el 100% de las evaluaciones de la fase de verificación enfocado en justificar las reuniones, fue calificada como "passed”, indicando que los sujetos de ambos grupos de trabajo si lograron realizar con éxito el requerimiento, teniendo como promedio un total de 100% en esta actividad.

En la Tabla 17 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de post desarrollo (pruebas de aceptación) teniendo como resultados que el grupo uno falló en completar el requerimiento con un 80% y el segundo grupo completó con éxito el requerimiento logrando “passed” con un 100%, obteniendo así un promedio total de 90% en esta actividad.

En la Tabla 17 se puede observar los valores porcentuales de respuesta en cuánto a la etapa de post desarrollo (retrospectiva) teniendo como resultados que el 100% de las evaluaciones de la fase de retroalimentación enfocado en la retrospectiva, fue calificada como "passed", indicando que los sujetos de ambos grupos de trabajo si lograron realizar con éxito el requerimiento, teniendo como promedio un total de 100% en esta actividad.

En definitiva, se puede observar que un 91,6% de actividades fue completada por parte de los sujetos experimentales que trabajaron con la metodología Kanban, señalando que las actividades que tuvieron más dificultades fueron las herramientas de desarrollo obteniendo un promedio de 83%, las historias de usuarios con un porcentaje de 80% y 85% respectivamente con los grupos dando como promedio un 83%, la actualización del tablero Kanban con un promedio de 65% y el uso de los límites de trabajo en proceso (WIP) con un promedio de 35%.

3.3 Evaluación del modelo de calidad en uso

Para evaluar el modelo de calidad, primero se tabuló los datos obtenidos de la ejecución del taller práctico (Anexo 1) y encuesta SUS (Anexo 5), luego se aplicó las métricas establecidas en la Tabla 1. A continuación, se detalla las mediciones del modelo establecido.

3.3.1 Evaluación de la característica Eficacia

- Subcaracterística Tareas Completas Scrum

  La subcaracterística de Tareas Completas evalúa la proporción de las tareas que se han completado correctamente sin una asistencia o ayuda técnica. Para obtener los resultados se tuvo que contabilizar el número de tareas únicas completadas y el número total de tareas únicas intentadas.

  Dónde,

  A= número de tareas únicas completas = 32 (sumatoria de todas las tareas completadas por los 4 sujetos)

  B= número de tareas únicas intentadas = 40 (sumatoria de todas las tareas intentadas por los 4 sujetos).

  La métrica de “tareas completas” describe la proporción de las tareas que se han completado correctamente sin asistencia, y en la Tabla 18 se observa la aplicación de la métrica.

Tabla 18
Métrica medir el rendimiento del usuario – Scrum

Métrica	**Medir el rendimiento del usuario**	
Elementos de datos	**Descripción**	**Valor**
A	Número de tareas únicas completado	32
B	Número total de tareas únicas intentadas	40

Elaboración: Propia.

Fórmula: X = A / B

Reemplazando en la fórmula se obtiene $X = \frac{32}{40} = 0{,}8$. Lo cual quiere decir que el 80% de las tareas fueron completadas por los sujetos.

- Subcaracterística Tareas Completas Extreme Programming

La subcaracterística de Tareas Completas evalúa la proporción de las tareas que se han completado correctamente sin una asistencia o ayuda técnica. Para obtener los resultados se tuvo que contabilizar el número de tareas únicas completadas y el número total de tareas únicas intentadas.

Dónde,

A= número de tareas únicas completas = 64 (sumatoria de todas las tareas completadas por los 8 sujetos)

B= número de tareas únicas intentadas = 84 (sumatoria de todas las tareas intentadas por los 8 sujetos)

La métrica de “tareas completas” describe la proporción de las tareas que se han completado correctamente sin asistencia, y en la Tabla 19 se observa la aplicación de la métrica.

Tabla 19
Métrica medir el rendimiento del usuario – Extreme Programming

Métrica	**Medir el rendimiento del usuario**	
Elementos de datos	**Descripción**	**Valor**
A	Número de tareas únicas completado	64
B	Número total de tareas únicas intentadas	84

Elaboración: Propia.

Fórmula: X = A / B

Reemplazando en la fórmula se obtiene $X = \frac{64}{84} = 0{,}76$. Lo cual quiere decir que el 76% de las tareas fueron completadas por los sujetos.

- Subcaracterística Tareas Completas Kanban

La subcaracterística de Tareas Completas evalúa la proporción de las tareas que se han completado correctamente sin una asistencia o ayuda técnica. Para obtener los resultados se tuvo que contabilizar el número de tareas únicas completadas y el número total de tareas únicas intentadas.

Dónde,

A= número de tareas únicas completas = 64 (sumatoria de todas las tareas completadas por los 8 sujetos)

B= número de tareas únicas intentadas = 92 (sumatoria de todas las tareas intentadas por los 8 sujetos)

La métrica de “tareas completas” describe la proporción de las tareas que se han completado correctamente sin asistencia, y en la Tabla 20 se observa la aplicación de la métrica.

Tabla 20
Métrica medir el rendimiento del usuario – Kanban

Métrica	**Medir el rendimiento del usuario**	
Elementos de datos	**Descripción**	**Valor**
A	Número de tareas únicas completado	64
B	Número total de tareas únicas intentadas	92

Fórmula: X = A / B

Reemplazando en la fórmula se obtiene $X = \frac{64}{92} = 0{,}69$. Lo cual quiere decir que el 69% de las tareas fueron completadas por los sujetos.

- Subcaracterística Objetivos logrados Scrum

  La subcaracterística de Objetivos logrados evalúa la proporción de los objetivos de la tarea que se logra correctamente sin ayuda. Para obtener los resultados se tuvo que contabilizar el valor proporcional de cada objetivo y el número de tareas que había en cada objetivo, tomando en cuenta si la tarea se completó o no se completó.

  Cada sujeto tenía 4 objetivos y cada objetivo tiene varias tareas para completar, ver tabla Tabla 11. Si una tarea no fue completada, el objetivo correspondiente a esa tarea también se lo consideró no completo. Por lo cual en la tabulación del taller se obtuvo 32 objetivos totales alcanzados por los sujetos, de los 40 (10*4) objetivos esperados en la ejecución del taller práctico. Por lo cual se obtuvo un 80% (0,8) de los objetivos alcanzados.

- Subcaracterística Objetivos logrados Kanban

  La subcaracterística de Objetivos logrados evalúa la proporción de los objetivos de la tarea que se logra correctamente sin ayuda. Para obtener los resultados se tuvo

que contabilizar el valor proporcional de cada objetivo y el número de tareas que había en cada objetivo, tomando en cuenta si la tarea se completó o no se completó.

Cada sujeto tenía 4 objetivos y cada objetivo tiene varias tareas para completar, ver tabla Tabla 11. Si una tarea no fue completada, el objetivo correspondiente a esa tarea también se lo consideró no completo. Por lo cual en la tabulación del taller se obtuvo 64 objetivos totales alcanzados por los sujetos, de los 96 (12*8) objetivos esperados en la ejecución del taller práctico. Por lo cual se obtuvo un 66% (0,66) de los objetivos alcanzados.

- Subcaracterística Objetivos logrados XP

  La subcaracterística de Objetivos logrados evalúa la proporción de los objetivos de la tarea que se logra correctamente sin ayuda. Para obtener los resultados se tuvo que contabilizar el valor proporcional de cada objetivo y el número de tareas que había en cada objetivo, tomando en cuenta si la tarea se completó o no se completó.

  Cada sujeto tenía 4 objetivos y cada objetivo tiene varias tareas para completar, ver tabla Tabla 11. Si una tarea no fue completada, el objetivo correspondiente a esa tarea también se lo consideró no completo. Por lo cual en la tabulación del taller se obtuvo 64 objetivos totales alcanzados por los sujetos, de los 88 (11*8) objetivos esperados en la ejecución del taller práctico. Por lo cual se obtuvo un 72% (0,72) de los objetivos alcanzados.

- Subcaracterística Los errores en una tarea Scrum

  La métrica de “los errores en una tarea” evalúa la proporción de los errores del usuario durante una tarea, tal como se observa en la Tabla 21.

  Dónde,

A= errores en las tareas completadas = 8 (sumatoria de todos los errores en tareas completadas, los errores cometidos en tareas no completadas no se tomaron en cuenta)

B= número de tareas únicas intentadas = 40

Tabla 21
Métrica de rendimiento del usuario - Scrum

Métrica	**Medir el rendimiento del usuario**	
Elementos de datos	Descripción	Valor
A	Número de tareas con errores	8
B	Número total de tareas	40

Fórmula: $X = 1 - \frac{A}{B}$

Reemplazando en la fórmula se obtiene que $X = 1 - \frac{8}{40} = 0{,}8$. Con este resultado se puede decir que el 80% de las tareas intentadas no tienen errores.

- Subcaracterística Los errores en una tarea Kanban

La métrica de “los errores en una tarea” evalúa la proporción de los errores del usuario durante una tarea, tal como se observa en la Tabla 22.

Dónde,

A= errores en las tareas completadas = 28 (sumatoria de todos los errores en tareas completadas, los errores cometidos en tareas no completadas no se tomaron en cuenta)

B= número de tareas únicas intentadas = 92

Tabla 22
Métrica de rendimiento del usuario - Kanban

Métrica	**Medir el rendimiento del usuario**	
Elementos de datos	Descripción	Valor
A	Número de tareas con errores	28
B	Número total de tareas	92

Fórmula: $X = 1 - \frac{A}{B}$

Reemplazando en la fórmula se obtiene que $X = 1 - \frac{28}{92} = 0{,}7$. Con este resultado se puede decir que el 70% de las tareas intentadas no tienen errores.

- Subcaracterística Los errores en una tarea XP

  La métrica de “los errores en una tarea” evalúa la proporción de los errores del usuario durante una tarea, tal como se observa en la Tabla 23.

  Dónde,

  A= errores en las tareas completadas = 20 (sumatoria de todos los errores en tareas completadas, los errores cometidos en tareas no completadas no se tomaron en cuenta)

  B= número de tareas únicas intentadas = 88

Tabla 23
Métrica de rendimiento del usuario - XP

Métrica	**Medir el rendimiento del usuario**	
Elementos de datos	Descripción	Valor
A	Número de tareas con errores	20
B	Número total de tareas	84

Fórmula: $X = 1 - \frac{A}{B}$

Reemplazando en la fórmula se obtiene que $X = 1 - \frac{20}{84} = 0{,}76$. Con este resultado se puede decir que el 76% de las tareas intentadas no tienen errores.

3.3.2 Evaluación de la característica Satisfacción

- Subcaracterística: Utilidad Scrum

  Las preguntas de la encuesta SUS seleccionadas para recolectar datos de medición de la "Utilidad" fueron: P1, P2 Y P3. Utilizando la escala de Likert se estableció pesos a las respuestas dadas por los sujetos, ver Tabla 24.

Tabla 24
Peso respuestas

Escala	**Respuesta**	**Peso**
1	Totalmente en desacuerdo	0,2
2	En desacuerdo	0,4
3	Neutral	0,6
4	De acuerdo	0,8
5	Totalmente de acuerdo	1

Fuente: Elaboración Propia.

Luego se calculó las respuestas conforme a la escala establecida y para el resultado se sumó el total de cada pregunta, ver Tabla 25.

Tabla 25
Resultados SUS Utilidad - Scrum

Pregunta	Sujetos satisfechos
P1	3,6 de 4
P2	3,4 de 4
P3	3,8 de 4

Fuente: Elaboración Propia.

Para obtener el valor total de los sujetos satisfechos se calculó el promedio de las tres preguntas.

Fórmula: $X = \frac{A}{B}$

Dónde,

A= usuarios satisfechos = 3,6 (ver Tabla 25)

B= usuarios encuestados = 4

Reemplazando en la fórmula se obtiene $X = \frac{3,6}{4} = 0,9$. Lo que quiere decir que el 90% de los sujetos estuvieron satisfechos al usar esta metodología.

- Subcaracterística: Utilidad Kanban

Las preguntas de la encuesta SUS seleccionadas para recolectar datos de medición de la “Utilidad” fueron: P1, P2 Y P3. Utilizando la escala de Likert se estableció pesos a las respuestas dadas por los sujetos, ver Tabla 24.

Luego se calculó las respuestas conforme a la escala establecida y para el resultado se sumó el total de cada pregunta, ver Tabla 26.

Tabla 26
Resultados SUS Utilidad - Kanban

Pregunta	Sujetos satisfechos
P1	5,6 de 8
P2	5,8 de 8
P3	6,2 de 8

Fuente: Elaboración Propia.

Para obtener el valor total de los sujetos satisfechos se calculó el promedio de las tres preguntas.

Fórmula: $X = \frac{A}{B}$

Dónde,

A= usuarios satisfechos = 5,87 (ver Tabla 26)

B= usuarios encuestados = 8

Reemplazando en la fórmula se obtiene $X = \frac{5,87}{8} = 0,73$. Lo que quiere decir que el 73% de los sujetos estuvieron satisfechos al usar esta metodología.

- Subcaracterística: Utilidad XP

Las preguntas de la encuesta SUS seleccionadas para recolectar datos de medición de la “Utilidad” fueron: P1, P2 Y P3. Utilizando la escala de Likert se estableció pesos a las respuestas dadas por los sujetos, ver Tabla 24.

Luego se calculó las respuestas conforme a la escala establecida y para el resultado se sumó el total de cada pregunta, ver Tabla 27.

Tabla 27
Resultados SUS Utilidad - XP

Pregunta	Sujetos satisfechos
P1	5,2 de 8
P2	5,6 de 8
P3	5,4 de 8

Fuente: Elaboración Propia.

Para obtener el valor total de los sujetos satisfechos se calculó el promedio de las tres preguntas.

Fórmula: $X = \frac{A}{B}$

Dónde,

A= usuarios satisfechos = 5,4 (ver Tabla 27)

B= usuarios encuestados = 8

Reemplazando en la fórmula se obtiene $X = \frac{5,4}{8} = 0,68$. Lo que quiere decir que el 68% de los sujetos estuvieron satisfechos al usar esta metodología.

- Subcaracterística: Comodidad Scrum

Las preguntas de la encuesta SUS seleccionadas para recolectar datos de medición de la "Comodidad" fueron desde la P4 hasta la P9. En la **Tabla *28*** se muestra el resultado de la ejecución de la encuesta.

Tabla 28
Resultado SUS Comodidad – Scrum

Respuesta	Peso	Usuarios	Satisfacción
Totalmente en desacuerdo	0,2	0	0,00
En desacuerdo	0,4	0	0,00
Neutral	0,6	1	0,15
De acuerdo	0,8	2	0,4
Totalmente de acuerdo	1	1	0,25

Fuente: Elaboración Propia.

Fórmula: $X = \mathrm{A} + \mathrm{B} + \mathrm{C} + \mathrm{D} + \mathrm{E}$

Dónde,

A= Muy de acuerdo =0,25

B= Algo de acuerdo= 0,40

C= Ni de acuerdo ni en desacuerdo= 0,15

D= Algo en desacuerdo= 0,00

E= Muy en desacuerdo = 0,00

Reemplazando en la fórmula se obtiene $X = 0{,}25 + 0{,}40 + 0{,}15 + 0{,}00 + 0{,}00 = 0{,}8$. Lo que quiere decir que el 80% de usuarios están cómodos de usar la metodología.

- Subcaracterística: Comodidad Kanban

Las preguntas de la encuesta SUS seleccionadas para recolectar datos de medición de la “Comodidad” fueron desde la P4 hasta la P9. En la Tabla 29 se muestra el resultado de la ejecución de la encuesta.

Tabla 29
Resultado SUS Comodidad – Kanban

Respuesta	Peso	Usuarios	Satisfacción
Totalmente en desacuerdo	0,2	0	0,00
En desacuerdo	0,4	1	0,05
Neutral	0,6	1	0,075
De acuerdo	0,8	4	0,4
Totalmente de acuerdo	1	2	0,25

Fuente: Elaboración Propia.

Fórmula: $X = \mathrm{A} + \mathrm{B} + \mathrm{C} + \mathrm{D} + \mathrm{E}$

Dónde,

A= Muy de acuerdo =0,25

B= Algo de acuerdo= 0,40

C= Ni de acuerdo ni en desacuerdo= 0,075

D= Algo en desacuerdo= 0,05

E= Muy en desacuerdo = 0,00

Reemplazando en la fórmula se obtiene $X = 0{,}25 + 0{,}40 + 0{,}075 + 0{,}05 + 0{,}00 = 0{,}78$. Lo que quiere decir que el 78% de usuarios están cómodos de usar la metodología.

- Subcaracterística: Comodidad XP

Las preguntas de la encuesta SUS seleccionadas para recolectar datos de medición de la “Comodidad” fueron desde la P4 hasta la P9. En la Tabla 30 se muestra el resultado de la ejecución de la encuesta.

Tabla 30
Resultado SUS Comodidad – XP

Respuesta	Peso	Usuarios	Satisfacción
Totalmente en desacuerdo	0,2	0	0,00
En desacuerdo	0,4	1	0,05
Neutral	0,6	3	0,23
De acuerdo	0,8	3	0,30
Totalmente de acuerdo	1	1	0,13

Fuente: Elaboración Propia.

Fórmula: $X = \mathrm{A} + \mathrm{B} + \mathrm{C} + \mathrm{D} + \mathrm{E}$

Dónde,

A= Muy de acuerdo =0,13

B= Algo de acuerdo= 0,30

C= Ni de acuerdo ni en desacuerdo= 0,23

D= Algo en desacuerdo= 0,05

E= Muy en desacuerdo = 0,00

Reemplazando en la fórmula se obtiene $X = 0{,}13 + 0{,}30 + 0{,}23 + 0{,}05 + 0{,}00 = 0{,}71$. Lo que quiere decir que el 71% de usuarios están cómodos de usar la metodología.

- Subcaracterística: Confianza Scrum

Fórmula $X = 1 - \%C, C = \frac{A}{B}$

Dónde,

A= número de reclamos = 2

B = número de usuarios = 4

C = % reclamos

Reemplazando en la fórmula se obtiene $C = \frac{2}{4}, X = 1 - 0{,}50 = 0{,}50$. Lo que quiere decir que el 50% de usuarios están cómodos de usar la metodología.

- Subcaracterística: Confianza Kanban

  Fórmula $X = 1 - \%C, C = \frac{A}{B}$

Dónde,

A= número de reclamos = 3

B = número de usuarios = 8

C = % reclamos

Reemplazando en la fórmula se obtiene $C = \frac{3}{8}, X = 1 - 0{,}38 = 0{,}62$. Lo que quiere decir que el 62% de usuarios están cómodos de usar la metodología.

- Subcaracterística: Confianza XP

  Fórmula $X = 1 - \%C, C = \frac{A}{B}$

Dónde,

A= número de reclamos = 1

B = número de usuarios = 8

C = % reclamos

Reemplazando en la fórmula se obtiene $C = \frac{1}{8}, X = 1 - 0{,}13 = 0{,}87$. Lo que quiere decir que el 87% de usuarios están cómodos de usar la metodología.

3.4 Resultados del modelo de calidad en uso

En esta sección se muestra los resultados obtenidos de la evaluación de la eficacia entre las metodologías ágiles Scrum, Kanban y Extreme Programming en base al modelo de calidad en uso presentado en las tablas, a continuación, se indica característica (C), subcaracterística (SC), porcentaje de cada una y de sus respectivos resultados que indican el grado de eficacia de la metodología.

3.4.1 Resultados Scrum

Como se observa en la Tabla 31, el porcentaje utilizado de la característica para el modelo de calidad en uso de la investigación fue del 100%, y se utilizó la subcaracterística de la eficacia con un 60% y de la satisfacción con un 40%.

Tabla 31
Resultado evaluación eficacia y satisfacción – Scrum

Característica	Sub-característica	Porcentaje Sub-característica (SC)	Porcentaje característica (C)	Medición	Resultado (SC)	Resultado(C)
Eficacia	Tareas Completas	28%	60%	0,80	22,4%	48%
	Objetivos alcanzados	16%		0,80	12,8%	
	Los errores en una tarea	16&		0,80	12,8%	
Satisfacción	Utilidad	15%	40%	0,90	13,5%	30,50%
	Confianza	15%		0,50	7,5%	
	Comodidad	10%		0,80	8%	
	Total	100%				78,50%

Fuente: Elaboración Propia.

3.4.2 Resultados Kanban

Como se observa en la Tabla 32, el porcentaje utilizado de la característica para el modelo de calidad en uso de la investigación fue del 100%, y se utilizó la subcaracterística de la eficacia con un 60% y de la satisfacción con un 40%.

Tabla 32
Resultado evaluación eficacia y satisfacción – Kanban

Característica	Sub-característica	Porcentaje Sub-característica (SC)	Porcentaje característica (C)	Medición	Resultado (SC)	Resultado(C)
Eficacia	Tareas Completas	28%	60%	0,69	19,32%	41,08%
	Objetivos alcanzados	16%		0,66	10,56%	
	Los errores en una tarea	16%		0,70	11,20%	
Satisfacción	Utilidad	15%	40%	0,73%	10,50%	27,60%
	Confianza	15%		0,62%	9,30%	
	Comodidad	10%		0,78%	7,8%	
	Total	100%				68,68%

Fuente: Elaboración Propia.

3.4.3 Resultados Extreme Programming

Como se observa en la Tabla 33, el porcentaje utilizado de la característica para el modelo de calidad en uso de la investigación fue del 100%, y se utilizó la subcaracterística de la eficacia con un 60% y de la satisfacción con un 40%.

Tabla 33
Resultado evaluación eficacia y satisfacción – XP

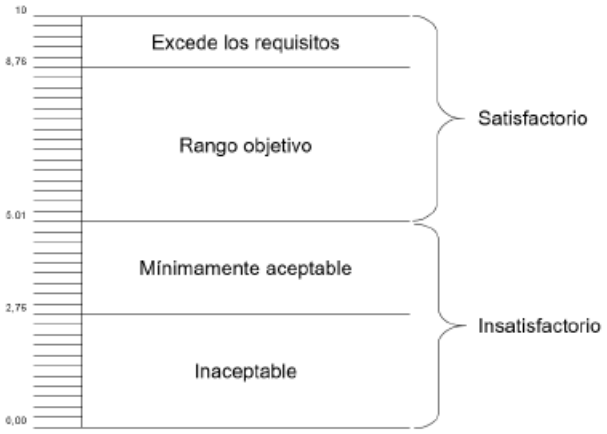
Característica	Sub-característica	Porcentaje Sub-característica (SC)	Porcentaje característica (C)	Medición	Resultado (SC)	Resultado(C)
Eficacia	Tareas Completas	28%	60%	0,76	21,28%	44,96%
	Objetivos alcanzados	16%		0,72	11,52%	
	Los errores en una tarea	16&		0,76	12,16%	
Satisfacción	Utilidad	15%	40%	0,68%	10,20%	30,35%
	Confianza	15%		0,87%	13,05%	
	Comodidad	10%		0,71%	7,10%	
	Total	100%				75,31%

Fuente: Elaboración Propia.

3.4.4 Análisis de resultados del modelo de calidad en uso

Para llevar a cabo el análisis de resultados, se utilizó una escala de medición de resultados que constaba de rangos que van desde inaceptable hasta exceder los requisitos, como se muestra en la **¡Error! No se encuentra el origen de la referencia.**. Según l a(ISO/IEC 25000, 2015), se tomaron en cuenta las medidas de calidad en uso del producto de software para determinar si cumple con los requisitos de calidad mediante el uso de un grado adecuado de confianza o métricas.

Figura 35
Resultados en la escala de medición



Fuente: Basada en (ISO/IEC 25000, 2015).

La evaluación del modelo de calidad en uso de la eficacia y satisfacción entre Scrum, Kanban y XP, como se puede observar en la Tabla 34 indica que en Scrum 78,50% (7,86) está dentro del rango objetivo y es satisfactorio; así mismo, la metodología XP arrojó 75,31% (7,53) que se encuentra en el rango objetivo indicando que es satisfactorio y superado mínimamente por la metodología Scrum. Finalmente, la metodología Kanban arrojó 68,68% (6,87) y se encuentra en el rango objetivo (satisfactorio) superado levemente por las metodologías XP y Scrum.

Tabla 34
Resultados Scrum, Kanban y XP

Tabla de Resultados				
Característica	Sub característica	Resultado Scrum	Resultado Kanban	Resultado XP
Eficacia	Tareas completas	7,86	6,87	7,53
	Objetivos logrados			
	Los errores en una tarea			
Satisfacción	Utilidad			
	Confianza			
	Comodidad			

Fuente: Elaboración Propia.

CONCLUSIONES

- El marcó teórico que se desarrolló en el Capítulo I, permitió fundamentar y caracterizar la base conceptual de cada metodología ágil empleada durante la investigación y permitió concebir el conocimiento para el diseño y ejecución de un experimento controlado, el cual se utiliza para medir el modelo de calidad en uso y lograr la comparación de la eficacia entre Scrum, Kanban y Extreme Programming.
- Mediante el uso de la norma ISO/IEC 25022 permitió definir un modelo de calidad en uso con base a un experimento controlado de una manera sencilla y didáctica (ver capítulo II, Tabla 7), en dónde, se seleccionó la característica de la eficacia y satisfacción, la cual se ajustó a los requerimientos de la investigación. Con la utilización de la norma se logró definir las métricas “tareas completas”,” objetivos logrados” y “los errores en una tarea” por parte de la eficacia y “utilidad”, “confianza” y “comodidad” para la satisfacción (Tabla 8), las cuales se usaron para medir de manera ordenada y cuantitativa los efectos en el modelo de calidad en uso.
- El experimento controlado (ver CAPITULO II) facilitó el control directo sobre los tratamientos experimentales (Scrum, Kanban y XP) además del diseño de experimentación factorial cruzado (Tabla 10), permitiendo una mayor flexibilidad y variabilidad en cuanto a la asignación de objetos (artefactos) a los sujetos de prueba, se les dio la oportunidad de trabajar en grupos acorde al gusto de cada individuo para las sesiones y tratamientos, logrando una distribución uniforme, cumpliendo con las expectativas de la investigación.
- La aplicación de la norma ISO/IEC 25022, facilitó evaluar los resultados obtenidos y permitió determinar que Scrum (7,86) tuvo un rango satisfactorio levemente mayor que XP (7,53), igualmente satisfactorio, y Kanban (6,87) que entra en un rango objetivo de satisfactorio, pero por debajo de las dos metodologías antes mencionadas. Lo que quiere que la metodología Scrum permitió mayor flexibilidad y autogestión al momento de construir software desde cero, que, con las metodologías XP y Kanban. Además, se podría decir que el uso de la metodología Scrum es más cómodo que las otras metodologías.

RECOMENDACIONES

- Para redactar el marco teórico indagar en las bases de datos de citas y resúmenes bibliográficos de centros académicos como por ejemplo de universidades o de investigación como Scopus, Science Direct o Web of Science para poseer una buena base de fundamentos teóricos y técnicos, además de apoyarse en las herramientas de gestión de referencias bibliográficas para llevar una mejor organización, una de ellas es Mendeley o EndNote que cuentan con aplicaciones web.
- Realizar la replicación del experimento controlado en diferentes entornos como, por ejemplo, en otros centros académicos, como también en el entorno industrial o empresarial. Con lo cual, se podrá aporta nuevos resultados que corroboren o refuten los resultados obtenidos en la presente investigación.
- Ejecutar el experimento controlado en entornos en los que se pueda aplicar en grupos de trabajo similares en cantidad de sujetos para mayor homogeneidad y normalización al momento de evaluar y recopilar datos.

- No realizar dos sesiones de prueba de la experimentación el mismo día, debido al estrés acumulado en los sujetos de prueba, ya que existe la posibilidad de que los resultados estén sesgados.

- Evaluar un producto software con las ISO/IEC 25000 en calidad en uso para conocer el nivel de calidad que tiene el software y así darle un valor adicional al producto. Las normas ISO/IEC 25000 ofrecen una guía para el diseño, medición y evaluación de la calidad en uso, aunque no son tan descriptivas y no cuentan con ejemplos prácticos, por lo cual al momento de evaluar con las normas ISO se recomienda apoyarse en trabajos donde han utilizado estás normas y así tener una guía y un ejemplo más claro.

BIBLIOGRAFÍA

Adnan, M., & Afzal, M. (2017). Ontology based multiagent effort estimation system for scrum agile method. *IEEE Access*, *5*, 25993–26005. https://doi.org/10.1109/ACCESS.2017.2771257

Alarcón-Aldana, A. C., Díaz, E. L., & Callejas-Cuervo, M. (2014). Guía para la evaluación de la usabilidad en los entornos virtuales de aprendizaje (EVA). *Informacion Tecnologica*, *25*(3), 135–144. https://doi.org/10.4067/S0718-07642014000300016

Anand, R. V., & Dinakaran, M. (2016). Popular Agile Methods in Software Development:Review and Analysis. *International Journal of Scientific and Technical Advancements*, *2*(4), 147–150.

Anderson, D. J., & Carmichael, A. (2016). *Essential Kanban condensed*.

Ballesteros, P., & Silva, P. (2008). A practical form to apply the System Kanban in the Colombian Mypimes. *Scientia et Technica Año XIV*, *39*.

Bowes, Jim. (2015). *Kanban vs Scrum vs XP – an Agile comparison - Manifesto*. https://manifesto.co.uk/kanban-vs-scrum-vs-xp-an-agile-comparison/

Cadavid, A., Fernández, J., & Morales, J. (2013). *A review of agile methodologies for software development*.

Cadavid, A. N., Fernández Juan, & Vélez, J. (2013). *Revisión de metodologías ágiles para el desarrollo de software*.

Cargua, J. (2023). *DESARROLLO DE UN PROTOTIPO DE SISTEMA WEB INFORMATIVO PARA CÁLCULO DE MATRÍCULA Y ARANCELES*.

Carrera de Software - UTN. (2023). *Misión y Visión*. https://software.utn.edu.ec/informacion/mision-y-vision/

Chasqui, P. (2020). *DESARROLLO DE UN SOFTWARE EDUCATIVO WEB GAMIFICADO PARA LA ENSEÑANZA DE INFORMÁTICA BÁSICA*.

Chaudhari, A. R., & Joshi, S. D. (2021). Study of effect of Agile software development Methodology on Software Development Process. *Proceedings of the 2nd International Conference on Electronics and Sustainable Communication Systems, ICESC 2021*, 1848–1851. https://doi.org/10.1109/ICESC51422.2021.9532842

Chauhan, V. (2014). *Role of Software Metrics to Improve Software Quality*. www.ijcsit.com

Chulde, F. (2022). *COMPARACIÓN DE LA FACILIDAD DE APRENDIZAJE ENTRE GRAPHQL Y REST MEDIANTE UN MODELO DE CALIDAD INTERNA BASADA EN LAS ISO/IEC 25000*.

code.visualstudio. (2024). *Documentation for Visual Studio Code*. https://code.visualstudio.com/docs

Computer Society, I., & Engineering Standards Committee, S. C. (2017). *ISO/IEC/IEEE 24748-3:2020, Systems and Software Engineering-Life Cycle Management-Part 3: Guidelines for the Application of ISO/IEC/IEEE 12207 (Software Life Cycle Processes)*.

CSOFT - UTN. (2023). *Encuesta sobre metodologías de proyectos de software*.

Cusumano, M. (2007). *Extreme Programming Compared with Microsoft-Style Iterative Development*. Cusumano, M. A. (2007). Extreme programming compared with Microsoft-style iterative development. Communications of the ACM, 50(10), 15. doi:10.1145/1290958.1290979

De La Cruz, Z. (2022). *CREACIÓN DE UNA APLICACIÓN BACKEND DEL MÓDULO DE TRANSFERENCIAS (CUENTAS Y MONTOS), CON MICROSERVICIOS GRAPHQL Y REST EN CONTENEDORES DOCKER, PARA FOMENTAR EL LEVANTAMIENTO DE UNA ARQUITECTURA DEVOPS*.

Developer.mozilla.org. (2024). *Introducción a Express/Node - Aprende desarrollo web | MDN*. https://developer.mozilla.org/es/docs/Learn/Server-side/Express_Nodejs/Introduction

Donoso, S. (2023). *Desarrollo de un sistema web turístico para la parroquia de Nono*.

Fernández González, J. (2013). *Introducción a las metodologías ágiles Otras formas de analizar y desarrollar*.

Flores, F., Sanhueza, V., Valdés, H., & Reyes, L. (2021). Metodologías ágiles: un análisis de los desafíos organizacionales para su implementación. *Revista Científica*, *43*(1), 38–49. https://doi.org/10.14483/23448350.18332

Fonseca, B., Matamoros, L., Hernández, ´Ángel, & Cornelio, O. (2019). *La estructura de desglose del trabajo como mecanismo viable para la generación de proyectos exitosos*.

Gaete, J., Villarroel, R., Figueroa, I., Cornide-Reyes, H., & Muñoz, R. (2021). Enfoque de aplicación ágil con Scrum, Lean y Kanban. In *Revista chilena de ingeniería* (Vol. 29, Issue 1).

Gong, H., Liu, B., & Shao, D. (2018a). A Simulation Model of Kanban Software Process. *Proceedings - Asia-Pacific Software Engineering Conference, APSEC*, *2017-December*, 745–746. https://doi.org/10.1109/APSEC.2017.96

Gong, H., Liu, B., & Shao, D. (2018b). A Simulation Model of Kanban Software Process. *Proceedings - Asia-Pacific Software Engineering Conference, APSEC*, *2017-December*, 745–746. https://doi.org/10.1109/APSEC.2017.96

Goto, T., Tsuchida, K., & Nishino, T. (2014). EPISODE: An extreme programming method for innovative software based on systems design. *Proceedings - 2014 IIAI 3rd International Conference on Advanced Applied Informatics, IIAI-AAI 2014*, 780–784. https://doi.org/10.1109/IIAI-AAI.2014.157

Gupta, A., Poels, G., & Bera, P. (2022). Using Conceptual Models in Agile Software Development: A Possible Solution to Requirements Engineering Challenges in Agile Projects. *IEEE Access*, *10*, 119745–119766. https://doi.org/10.1109/ACCESS.2022.3221428

Gupta, N., Sharma, H., Kumar, S., Kumar, A., & Kumar, R. (2022). A Comparative Study of Implementing Agile Methodology and Scrum Framework for Software Development. *Proceedings of the 2022 11th International Conference on System Modeling and Advancement in Research Trends, SMART 2022*, 1088–1092. https://doi.org/10.1109/SMART55829.2022.10047477

Hamid, M., Zeshan, F., Ahmad, A., Ahmad, F., Hamza, M. A., Khan, Z. A., Munawar, S., & Aljuaid, H. (2020). An Intelligent Recommender and Decision Support System (IRDSS) for Effective Management of Software Projects. *IEEE Access*, *8*, 140752–140766. https://doi.org/10.1109/ACCESS.2020.3010968

Hayat, F., Rehman, A., Sarmad, K., Wahab, K., & Abbas, M. (2019). *The Influence of Agile Methodology (Scrum) on Software Project Management*.

Hohl, P., Klünder, J., van Bennekum, A., Lockard, R., Gifford, J., Münch, J., Stupperich, M., & Schneider, K. (2018). Back to the future: origins and directions of the “Agile Manifesto” – views of the originators. *Journal of Software Engineering Research and Development*, *6*(1). https://doi.org/10.1186/s40411-018-0059-z

ISO. (2016). *ISO/IEC 25022:2016 - Systems and software engineering — Systems and software quality requirements and evaluation (SQuaRE) — Measurement of quality in use*. https://www.iso.org/standard/35746.html

ISO/IEC 25000. (2015). *NORMAS ISO 25000*. https://iso25000.com/index.php/normas-iso-25000

ISO/IEC 25022. (2016). *ISO/IEC 25022:2016(en), Systems and software engineering — Systems and software quality requirements and evaluation (SQuaRE) — Measurement of quality in use*. https://www.iso.org/obp/ui/en/#iso:std:iso-iec:25022:ed-1:v1:en

ISO/IEC 25023. (2020). *INTE/ISO/IEC 25023:2020 Ingeniería de sistemas y de software - Requisitos y evaluación de la calidad de sistemas y del software (SQuaRE) – Medición de la calidad de sistemas y productos de software. 506.* https://erp.inteco.org/shop/inte-iso-iec-25023-2020-ingenieria-de-sistemas-y-de-software-requisitos-y-evaluacion-de-la-calidad-de-sistemas-y-del-software-square-medicion-de-la-calidad-de-sistemas-y-productos-de-software-5949#attr=

Jaramillo, A. (2021). *Análisis y comparación de las metodologías de SCRUM y según PMI gestión de proyectos*.

Joskowicz, J. (2008). *Reglas y Prácticas en eXtreme Programming*.

Kaur, K., Khurana, M., & Manisha. (2021). Impact of Agile Scrum Methodology on Time to Market and Code Quality - A Case Study. *Proceedings - 2021 3rd*

International Conference on Advances in Computing, Communication Control and Networking, ICAC3N 2021, 1673–1678. https://doi.org/10.1109/ICAC3N53548.2021.9725375

Kitchenham, B., Pearl Brereton, O., Budgen, D., Turner, M., Bailey, J., & Linkman, S. (2009). Systematic literature reviews in software engineering - A systematic literature review. In *Information and Software Technology* (Vol. 51, Issue 1, pp. 7–15). https://doi.org/10.1016/j.infsof.2008.09.009

Legowo, M., Indiarto, B., & Prayitno, D. (2019). *Agile Software Methodology with Scrum for Developing Quality Assurance System*.

Letelier, P., & Penadés, C. (2015). *Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP)*. www.agileuniverse.com.

Letelier, P., Penadés, C., Canós, J., & Sánchez, E. (2012). *Metodologías Ágiles en el Desarrollo de Software*.

Maida, E. G., & Pacienzia, J. (2015). *Metodologías de desarrollo de software*. https://repositorio.uca.edu.ar/bitstream/123456789/522/1/metodologias-desarrollo-software.pdf

Maida, G., & Pacienzia, J. (2015). *Metodologías de desarrollo de software*. http://bibliotecadigital.uca.edu.ar/repositorio/tesis/metodologias-desarrollo-software.pdf

Maldonado, J. (2020). *APLICACIÓN DE UN FRAMEWORK PARA LA EVALUACIÓN DE LA CALIDAD EN USO DEL SII-ACADÉMICO DE LA ESCUELA POLITÉCNICA NACIONAL*.

Manisha, Khurana, M., & Kaur, K. (2021). Impact of Agile Scrum Methodology on Team's Productivity and Client Satisfaction - A Case Study. *Proceedings - 2021 3rd International Conference on Advances in Computing, Communication Control and Networking, ICAC3N 2021*, 1686–1691. https://doi.org/10.1109/ICAC3N53548.2021.9725505

Matthies, C. (2018). Scrum2kanban: Integrating kanban and scrum in a university software engineering capstone course. *Proceedings - International Conference on Software Engineering*, 48–55. https://doi.org/10.1145/3194779.3194784

Mazzanti, G. (2012). Agile in the bathtub: Developing and producing bathtubs the agile way. *Proceedings - 2012 Agile Conference, Agile 2012*, 197–203. https://doi.org/10.1109/Agile.2012.27

Meléndez, S., Gaitan, M., & Pérez, N. (2016). *METODOLOGIA ÁGIL DE DESARROLLO DE SOFTWARE PROGRAMACION EXTREMA.*

Mera Paz, J. (2016). Análisis del proceso de pruebas de calidad de software. *Ingeniería Solidaria*, *12*(20), 163–176. https://doi.org/10.16925/in.v12i20.1482

Miswar, Suhardi, & Kurniawan. (2018). *A Systematic Literature Review on Survey Data Collection System*.

Mohammad, A. H., Alwadan, T., Mohammad, J., & Ababneh, A. (2013). Agile Software Methodologies: Strength and Weakness. In *Article in International Journal of Engineering Science and Technology*. https://www.researchgate.net/publication/257207655

Mohammed, A., & Abushama, H. (2013). *Popular Agile Approaches in Software Development: Review and Analysis*.

Naciones Unidas. (2016). *Infraestructura - Desarrollo Sostenible*. https://www.un.org/sustainabledevelopment/es/infrastructure/

Nakai, H., Tsuda, N., Honda, K., Washizaki, H., & Fukazawa, Y. (2016). Initial Framework for Software Quality Evaluation Based on ISO/IEC 25022 and ISO/IEC 25023. *Proceedings - 2016 IEEE International Conference on Software Quality, Reliability and Security-Companion, QRS-C 2016*, 410–411. https://doi.org/10.1109/QRS-C.2016.66

Narváez, D., & Vaca, V. (2022). *DESARROLLO DE UN PROTOTIPO DE SISTEMA WEB PARA LA GESTIÓN DE INVENTARIO PARA UN PROVEEDOR DE SERVICIO DE INTERNET*.

Navarrete, L., & Cárdenas, M. (2016). *Diseño y desarrollo de un CMS(Content Managment System) para código abierto utilizando Node.js y Mongo.db como base de datos documental*.

Palacio, M. (2022). *Scrum Master (3rd ed.)*. https://www.scrummanager.com/files/scrum_master.pdf

Papadopoulos, G. (2015). Moving from Traditional to Agile Software Development Methodologies Also on Large, Distributed Projects. *Procedia - Social and Behavioral Sciences*, *175*, 455–463. https://doi.org/10.1016/j.sbspro.2015.01.1223

Patilla, H. J., Gómez Enciso, E., Carlos, J., Pulache, J., Lozano Rodríguez, J. L., Solórzano Huallanca, E., & Meneses Conislla, Y. (2021). *Modelo de Gestión de Desarrollo de Software Ágil mediante Scrum y Kanban sobre la Programación Extrema*.

Pérez. (2012). *Guía Comparativa de Metodologías Ágiles*.

Pérez, O. (2011). *Cuatro enfoques metodológicos para el desarrollo de Software RUP – MSF – XP - SCRUM*.

Postgresql.org. (2024). *PostgreSQL: About*. https://www.postgresql.org/about/

Pradanita, W., & Rochimah, S. (2020). *Quality in Use of Digital Wallet based on ISO/IEC 25022*.

Project Management Institute. (2021). *The standard for project management and a guide to the project management body of knowledge (PMBOK guide).* (Seventh).

Puentes, R., Parada, J., & Pabón, L. (2023). *WORK BREAKDOWN STRUCTURES (WBS) IN THE SCOPE MANAGEMENT OF SOFTWARE DEVELOPMENT PROJECTS*. https://orcid.org/0000-0001-8430-2660

Quiña-Mera, A., Fernandez, P., García, J. M., & Ruiz-Cortés, A. (2023). GraphQL: A Systematic Mapping Study. *ACM Computing Surveys*, *55*(10). https://doi.org/10.1145/3561818

Quiña-Mera, J. A., Guevara-Vega, C. P., Guzmán-Chamorro, E. D., Guevara-Vega, V. A., & Andrade, A. V. B. (2019). Functional Requirement Management Automation and the Impact on Software Projects: Case Study in Ecuador. *Advances in Intelligent Systems and Computing*, *918*, 317–324. https://doi.org/10.1007/978-3-030-11890-7_31

Rahmat, A., & Hanifiah, N. A. M. (2020, December 14). Usability Testing in Kanban Agile Process for Club Management System. *6th International Conference on Interactive Digital Media, ICIDM 2020*. https://doi.org/10.1109/ICIDM51048.2020.9339668

Rajpal, M. (2018). Effective distributed pair programming. *Proceedings - International Conference on Software Engineering*, 6–10. https://doi.org/10.1145/3196369.3196388

Randall, R. (2014). Agile at IBM: Software developers teach a new dance step to management. *Strategy and Leadership*, *42*(2), 26–29. https://doi.org/10.1108/SL-01-2014-0003

Renata, R., & Centeno, J. (2015). *Guía práctica para la selección de una metodología ágil respecto al tipo de proyecto de desarrollo de software entre SCRUM, XP y KANBAN*. http://www.ulacit.ac.cr

Roa, P., Morales, C., & Gutiérrez, P. (2015). *Norma ISO / IEC 25000*. https://revistas.udistrital.edu.co/index.php/tia

Ruk, A., Khan, M., & Khan Sehar. (2019). *A survey on Adopting Agile Software Development: Issues & Its impact on Software Quality*.

Salvay, J. E. (2017). *Kanban y Scrumban orientados a Proyectos de Tecnología de la Información*.

Santos, E., Lima, T. M., & Gaspar, P. D. (2023). Optimization of the Production Management of an Upholstery Manufacturing Process Using Lean Tools: A Case Study. *Applied Sciences (Switzerland)*, *13*(17). https://doi.org/10.3390/app13179974

Scrum.org, Vacanti, D., & Yeret, Y. (2021). *La Guía Kanban para Scrum Teams*.

Shamshurin, I., & Saltz, J. S. (2019). Using a coach to improve team performance when the team uses a kanban process methodology. *International Journal of Information Systems and Project Management*, *7*(2), 61–77. https://doi.org/10.12821/ijispm070204

Singh, A. (2018). *INTEGRATING THE EXTREME PROGRAMING MODEL WITH SECURE PROCESS FOR REQUIREMENT SELECTION*.

Telemaco, U., Oliveira, T., Alencar, P., & Cowan, D. (2020). A catalogue of agile smells for agility assessment. *IEEE Access*, *8*, 79239–79259. https://doi.org/10.1109/ACCESS.2020.2989106

Tulia, V., & Jácome, A. (2018). *Calidad de software del módulo de talento humano del sistema informático de la Universidad Técnica del Norte bajo la norma ISO/IEC 25000*.

Universidad de Ingeniería y Tecnología. (2022). *La diferencia entre Kanban Coach, Agile Coach y Scrum Master en un Equipo de Gestión Ágil.* https://educacion-ejecutiva.utec.edu.pe/blog/diferencia-entre-kanban-coach-agile-coach-scrum-master-equipo-gestion-agil

Vaca, T., & Jácome, A. (2018). *Calidad de software del módulo de talento humano del sistema informático de la Universidad Técnica del Norte bajo la norma ISO/IEC 25000*. https://www.researchgate.net/publication/325022337_Calidad_de_software_del_modulo_de_talento_humano_del_sistema_informatico_de_la_Universidad_Tecnica_del_Norte_bajo_la_norma_ISOIEC_25000

Vega, V., Quelopana, A., Flores, C., & Munizaga, A. (2018). Guía de Aplicación del Modelo de DeLone y McLean para la Evaluación de Productos de Software. *RISTI - Revista Iberica de Sistemas e Tecnologias de Informacao*, *29*, 14–29. https://doi.org/10.17013/risti.29.14-29

Vegas, S., Apa, C., & Juristo, N. (2016). Crossover Designs in Software Engineering Experiments: Benefits and Perils. *IEEE Transactions on Software Engineering*, *42*(2), 120–135. https://doi.org/10.1109/TSE.2015.2467378

Verret, J. (2018). *Presented to the Interdisciplinary Studies Program: Implementing Agile Methodology: Challenges and Best Practices*.

Villarreal, Darwin. (2013). *Adopción de una metodología ágil para proyectos de software (Tésis de Maestría)*. https://bit.ly/3OBC4CW

Wohlin, C., Höst, M., & Henningsson, K. (2003). LNCS 2765 - Empirical Research Methods in Software Engineering. In *LNCS* (Vol. 2765).

WQR. (2022). *World Quality Report 2022-23: El 72% de las organizaciones cree que la ingeniería de calidad puede contribuir al aspecto medioambiental de la TI sostenible*. Capgemini. https://www.capgemini.com/es-es/noticias/notas-de-prensa/world-quality-report-2022-23-el-72-de-las-organizaciones-cree-que-la-ingenieria-de-calidad-puede-contribuir-al-aspecto-medioambiental-de-la-ti-sostenible/

Yasvi, M. A., & Yadav, K. S. (2019). *Review On Extreme Programming-XP*. https://www.researchgate.net/publication/332465869

Zumba, J., & León, C. (2018). Evolution of the Methodologies and Models used in Software Development. *INNOVA Research Journal*, *3*(10), 20–33.

Zumba, J. P. (2018). Evolución de las Metodologías y Modelos utilizados en el Desarrollo de Software. *INNOVA Research Journal*, *3*(10), 20–33. https://doi.org/10.33890/innova.v3.n10.2018.651

ANEXOS

Anexo 1
Requerimientos del Proyecto de Aula

Proyecto de aula: Módulo de manejo de proyectos jdizaf@utn.edu.ec

Objetivo del taller: Implementar un módulo de manejo de proyectos.

Prerrequisitos: El equipo deberá tomar en consideración que para el desarrollo de este proyecto deben escoger la base de datos, el lenguaje a desarrollar, levantar la arquitectura, todo como convenga de mejor manera al equipo. Tener funcional el entorno del proyecto de aula También se deben crear con anticipaciones las respectivas conexiones a base de datos o al servidor de aplicaciones en caso de usar.

1. Enunciado:

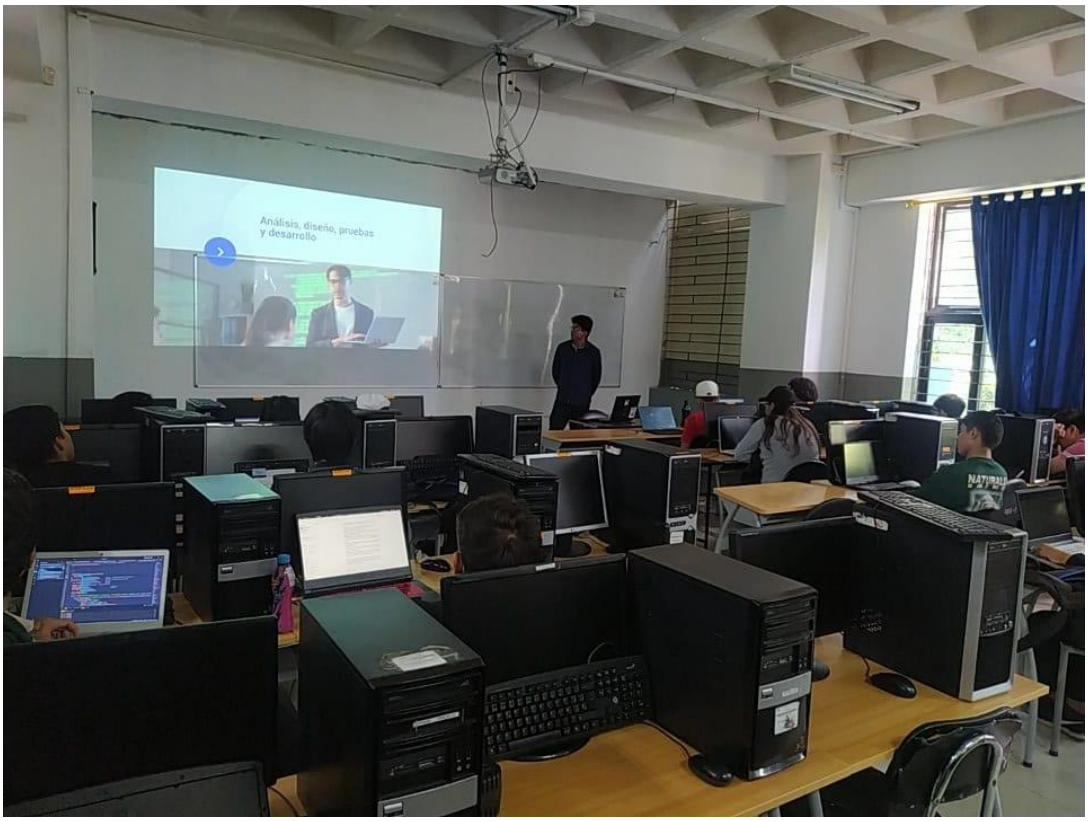
Nexus es una compañía enfocada en comercializar productos y servicios tecnológicos. Sus principales divisiones incluyen la venta de computadoras de diferentes gamas y la creación de software personalizado. Debido al crecimiento significativo de la empresa, especialmente en el ámbito del desarrollo de software, se busca mejorar la gestión de proyectos para la elaboración de sistemas informáticos.

De la entrevista con el jefe de proyectos, resultaron las siguientes ideas:

- Antes de iniciar un proyecto, se entrevista al cliente para obtener definir el problema que quiere resolver y los requisitos principales con respecto al software.
- Luego de llegar a un acuerdo económico y de tiempo, el gerente de Nexus aprueba el nuevo proyecto y firma juntamente con el cliente un contrato. Caso contrario simplemente se archiva la información de la entrevista con el cliente para futuras revisiones.
- En el caso de continuar con el proyecto, el jefe de desarrollo registra la información básica como: nombre del cliente, nombre del proyecto, fechas de inicio y fin, el porcentaje de avance y el estado en el que se encuentra (iniciado, en desarrollo, cancelado o finalizado). No se registran datos de costos por motivos de confidencialidad.
- En lo posterior, el jefe de desarrollo asigna una o varias tareas a un equipo de Analistas de sistemas. Se requiere conocer de cada asignación: el nombre de la tarea, las fechas de inicio y fin, el porcentaje de avance y el analista al que fue asignada la tarea.
- En cualquier momento el jefe de desarrollo puede consultar un reporte de todos los proyectos y conocer su estado.
- Todos los días martes, en una reunión de trabajo con el Gerente y el jefe de desarrollo de Nexus, el gerente puede proceder a cerrar determinado proyecto.
- La empresa Nexus se caracteriza por brindar el mejor servicio, esto conlleva realizar validaciones de los respectivos campos y realizar pruebas en caso de necesitarlas.

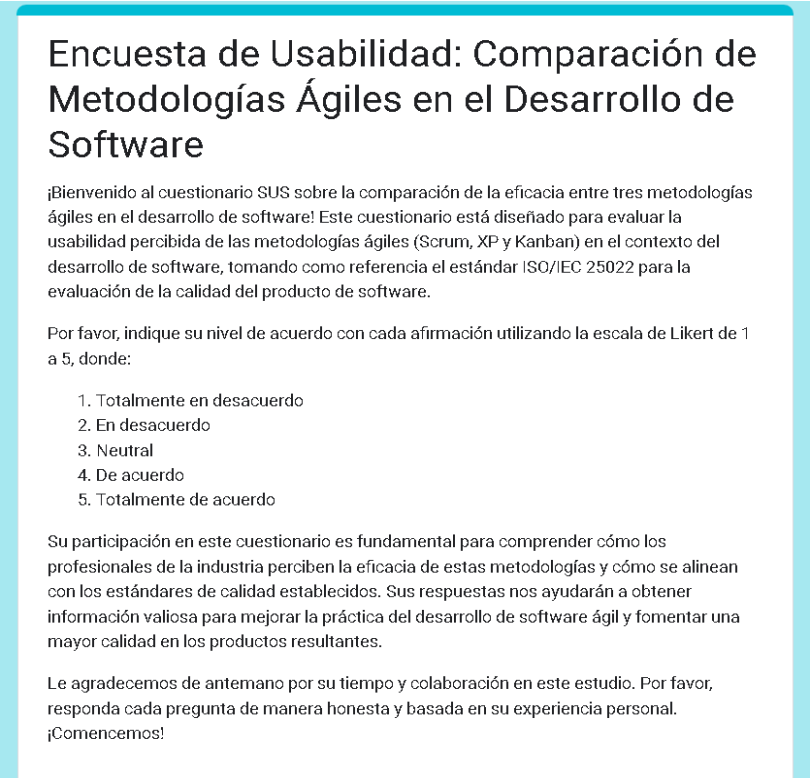
Anexo 2
Búsqueda de artículos

Anexo 3
Capacitación a estudiantes de la asignatura de Construcción de Software



Anexo 4
Revisión de avances

Anexo 5
Encuesta de Usabilidad I



Encuesta de Usabilidad: Comparación de Metodologías Ágiles en el Desarrollo de Software

¡Bienvenido al cuestionario SUS sobre la comparación de la eficacia entre tres metodologías ágiles en el desarrollo de software! Este cuestionario está diseñado para evaluar la usabilidad percibida de las metodologías ágiles (Scrum, XP y Kanban) en el contexto del desarrollo de software, tomando como referencia el estándar ISO/IEC 25022 para la evaluación de la calidad del producto de software.

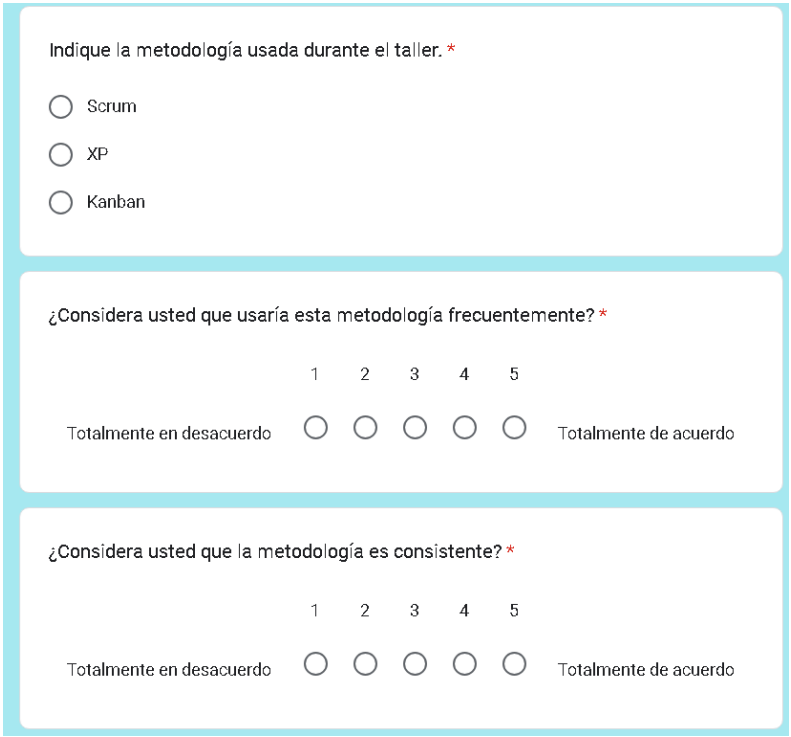
Por favor, indique su nivel de acuerdo con cada afirmación utilizando la escala de Likert de 1 a 5, donde:

1. Totalmente en desacuerdo
2. En desacuerdo
3. Neutral
4. De acuerdo
5. Totalmente de acuerdo

Su participación en este cuestionario es fundamental para comprender cómo los profesionales de la industria perciben la eficacia de estas metodologías y cómo se alinean con los estándares de calidad establecidos. Sus respuestas nos ayudarán a obtener información valiosa para mejorar la práctica del desarrollo de software ágil y fomentar una mayor calidad en los productos resultantes.

Le agradecemos de antemano por su tiempo y colaboración en este estudio. Por favor, responda cada pregunta de manera honesta y basada en su experiencia personal. ¡Comencemos!

Anexo 6
Encuesta de Usabilidad II



Indique la metodología usada durante el taller. *

- Scrum
- XP
- Kanban

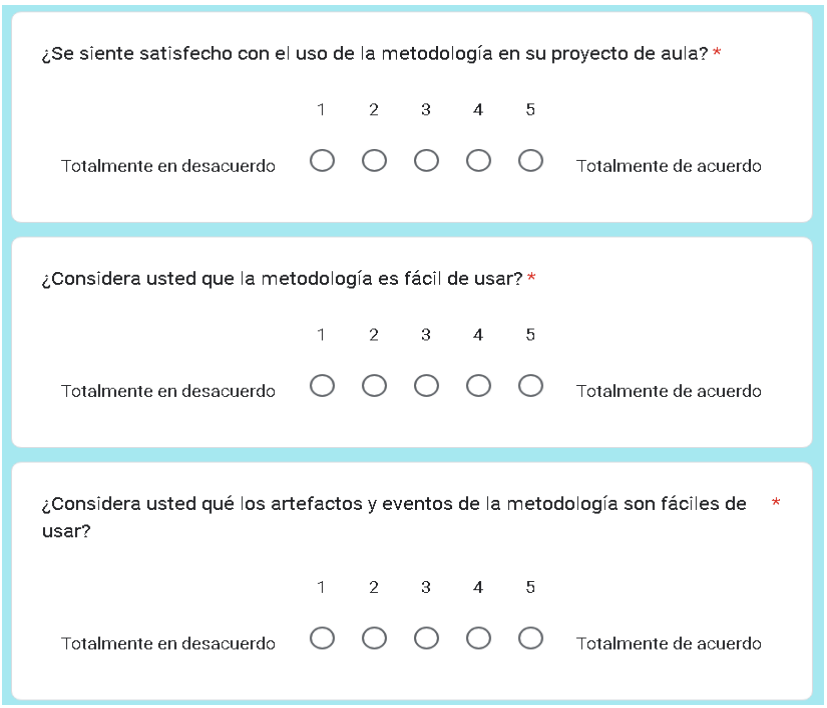
¿Considera usted que usaría esta metodología frecuentemente? *

	1	2	3	4	5	
Totalmente en desacuerdo	○	○	○	○	○	Totalmente de acuerdo

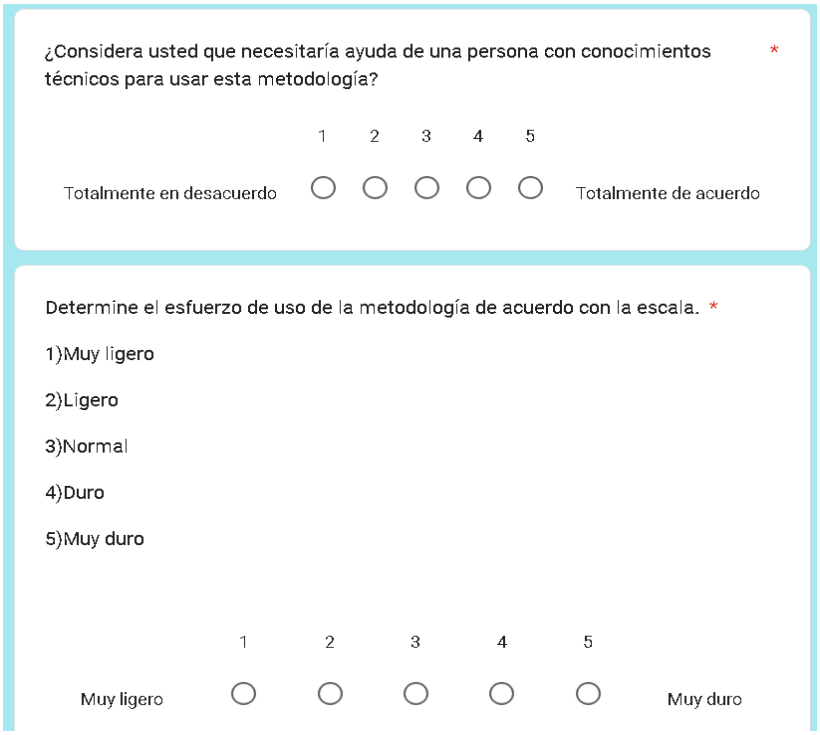
¿Considera usted que la metodología es consistente? *

	1	2	3	4	5	
Totalmente en desacuerdo	○	○	○	○	○	Totalmente de acuerdo

Anexo 7
Encuesta de Usabilidad III

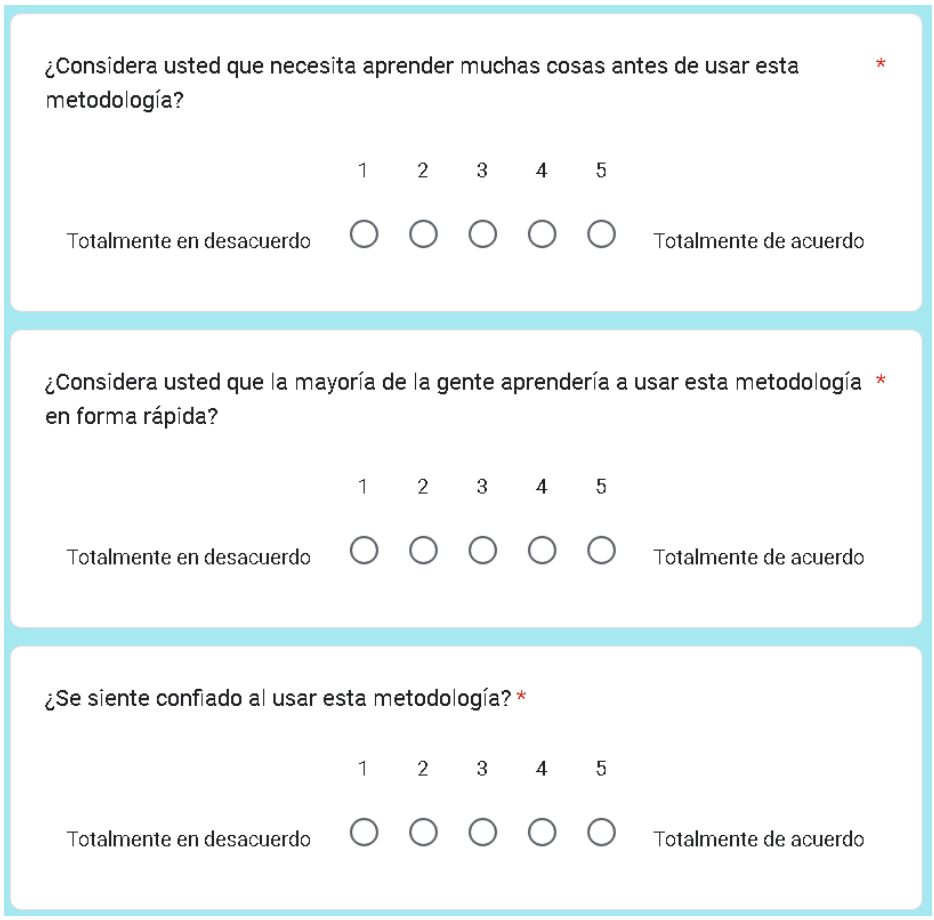


Anexo 8
Encuesta de Usabilidad IV

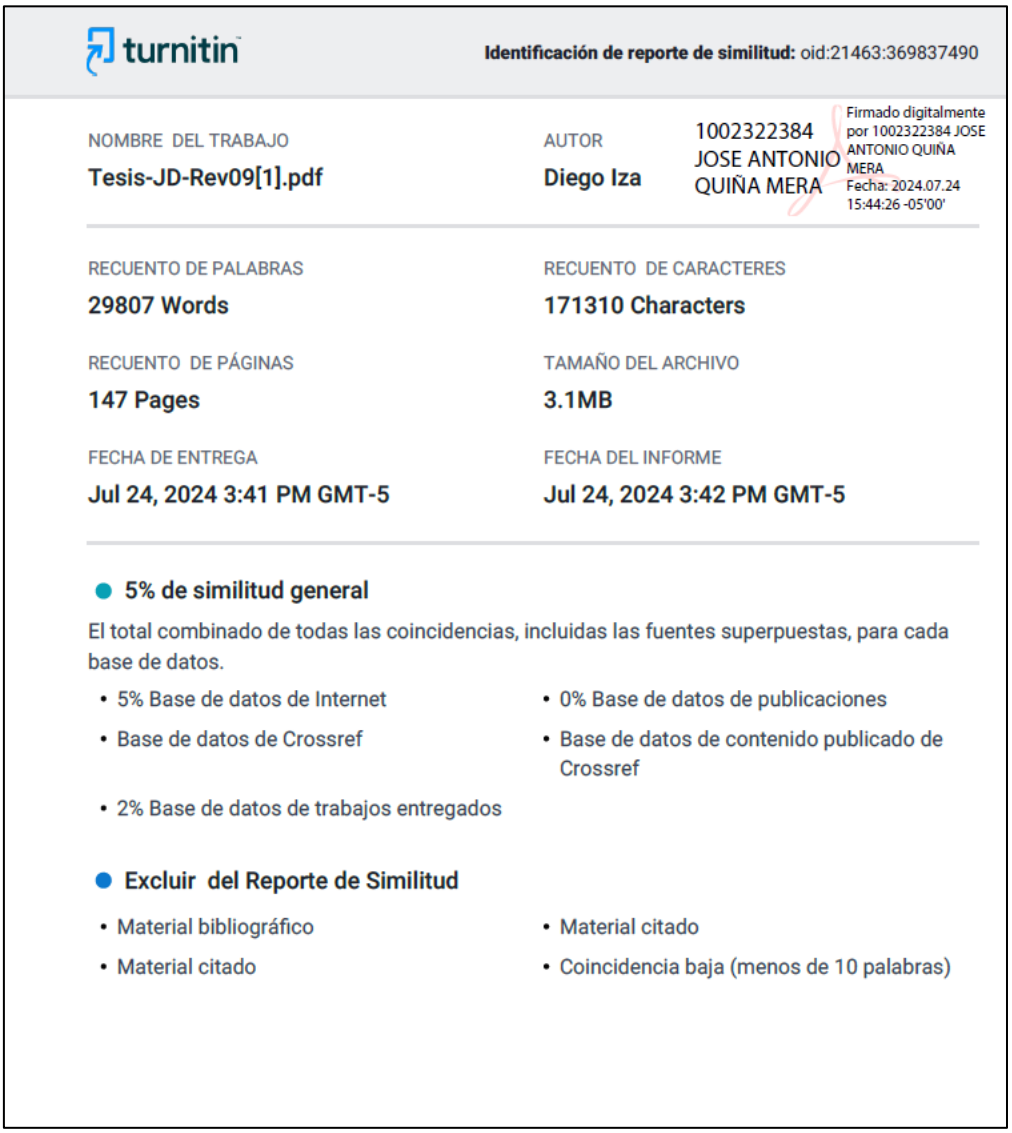


Anexo 9

Encuesta de Usabilidad V



Anexo 10
Identificación de reporte de similitud



turnitin

Identificación de reporte de similitud: oid:21463:369837490

NOMBRE DEL TRABAJO
Tesis-JD-Rev09[1].pdf

AUTOR
Diego Iza

1002322384 JOSE ANTONIO QUIÑA MERA
Firmado digitalmente por 1002322384 JOSE ANTONIO QUIÑA MERA
Fecha: 2024.07.24 15:44:26 -05'00'

RECUENTO DE PALABRAS
29807 Words

RECUENTO DE CARACTERES
171310 Characters

RECUENTO DE PÁGINAS
147 Pages

TAMAÑO DEL ARCHIVO
3.1MB

FECHA DE ENTREGA
Jul 24, 2024 3:41 PM GMT-5

FECHA DEL INFORME
Jul 24, 2024 3:42 PM GMT-5

5% de similitud general

El total combinado de todas las coincidencias, incluidas las fuentes superpuestas, para cada base de datos.

- 5% Base de datos de Internet
- 0% Base de datos de publicaciones
- Base de datos de Crossref
- Base de datos de contenido publicado de Crossref
- 2% Base de datos de trabajos entregados

Excluir del Reporte de Similitud

- Material bibliográfico
- Material citado
- Material citado
- Coincidencia baja (menos de 10 palabras)